

Contents

S. AL-DEEN ALMOUSA, M. TELEK, Enhanced optimization of high order concentrated matrix-exponential distributions	5
N. BÁTFAI, R. BESENCZI, P. JESZENSZKY, M. SZABÓ, M. ISPÁNY, Markov modeling of traffic flow in Smart Cities	21
I. K. BODA, E. TÓTH, English language learning by visualizing the literary content of a knowledge base in the three-dimensional space	45
A. BODONYI, GY. KURUCZ, G. HOLLÓ, R. KUNKLI, A barycentric coordinates-based visualization framework for movement of microscopic organisms	61
G. BOGACSOVICS, A. HAJDU, R. LAKATOS, M. BEREGI-KOVÁCS, A. TIBA, H. TOMÁN, Replacing the SIR epidemic model with a neural network and training it further to increase prediction accuracy	73
M. DÍAZ, O. NICOLIS, J. C. MARÍN, S. BARAN, Post-processing methods for calibrating the wind speed forecasts in central regions of Chile	93
D. EFROSININ, I. KOCHETKOVA, N. STEPANOVA, A. YAROVSLAVTSEV, K. SAMOUYLOV, R. VALENTINI, Trees classification based on Fourier coefficients of the sapflow density flux	109
I. FAZEKAS, A. BARTA, L. FÓRIÁN, Ensemble noisy label detection on MNIST	125
A. HIJAZY, A. ZEMPLÉNI, How well can screening sensitivity and sojourn time be estimated	139
T. KÁDEK, T. MIHÁLYDEÁK, Dealing with uncertainty: A rough-set-based approach with the background of classical logic	157
M. A. KORTEBY, Z. GÁL, P. POLGÁR, Multi dimensional analysis of sensor communication processes	169
G. KOVÁSZNAI, K. GAJDÁR, N. NARODYTSKA, Portfolio solver for verifying Binarized Neural Networks	183
A. KUKI, T. BÉRCZES, Á. TÓTH, J. SZTRIK, A contribution to scheduling jobs submitted by finite-sources in computational clusters	201
O. LANTANG, GY. TERDIK, A. HAJDU, A. TIBA, Investigation of the efficiency of an interconnected convolutional neural network by classifying medical images	219
T. MAJOROS, S. ONIGA, Y. XIE, Motor imagery EEG classification using feedforward neural network	235
D. PAPP, M. ZOMBOR, L. BUTTYÁN, TEE-based protection of cryptographic keys on embedded IoT devices	245
C. RIESER, P. FILZMOSER, Compositional trend filtering	257
K. SCHÄFFER, CS. I. SIDLÓ, Exploiting the structure of communication in actor systems	271
R. SCHEFZIK, SimBPDD: Simulating differential distributions in Beta-Poisson models, in particular for single-cell RNA sequencing data	283
Z. GY. YANG, Á. AGÓCS, G. KUSPER, T. VÁRADI, Abstractive text summarization for Hungarian	299

ANNALES MATHEMATICAE ET INFORMATICAЕ 53. (2021)

ANNALES MATHEMATICAE ET INFORMATICAЕ

TOMUS 53. (2021)



COMMISSIO REDACTORIUM

Sándor Bácsó (Debrecen), Sonja Gorjanc (Zagreb), Tibor Gyimóthy (Szeged),
Miklós Hoffmann (Eger), József Holovács (Eger), Tibor Juhász (Eger),
László Kovács (Miskolc), Gergely Kovásznai (Eger), László Kozma (Budapest),
Kálmán Liptai (Eger), Florian Luca (Mexico), Giuseppe Mastroianni (Potenza),
Ferenc Mátyás (Eger), Ákos Pintér (Debrecen), Miklós Rontó (Miskolc),
László Szalay (Sopron), János Sztrik (Debrecen), Gary Walsh (Ottawa)



HUNGARIA, EGER

ANNALES MATHEMATICAE ET INFORMATICAE

VOLUME 53. (2021)

EDITORIAL BOARD

Sándor Bácsó (Debrecen), Sonja Gorjanc (Zagreb), Tibor Gyimóthy (Szeged),
Miklós Hoffmann (Eger), József Holovács (Eger), Tibor Juhász (Eger),
László Kovács (Miskolc), Gergely Kovásznai (Eger), László Kozma (Budapest),
Kálmán Liptai (Eger), Florian Luca (Mexico), Giuseppe Mastroianni (Potenza),
Ferenc Mátyás (Eger), Ákos Pintér (Debrecen), Miklós Rontó (Miskolc),
László Szalay (Sopron), János Sztrik (Debrecen), Gary Walsh (Ottawa)

INSTITUTE OF MATHEMATICS AND INFORMATICS
ESZTERHÁZY KÁROLY UNIVERSITY
HUNGARY, EGER

Selected papers of the
1st Conference on
Information Technology
and Data Science

The conference was organized by
Faculty of Informatics, University of Debrecen, Hungary,
November 6–8, 2020

Conference Chairmen
István Fazekas and András Hajdu

*The conference was supported by the construction
EFOP-3.6.3-VEKOP-16-2017-00002.
The project was co-financed by the
Hungarian Government and the European Social Fund.*

HU ISSN 1787-6117 (Online)

A kiadásért felelős az
Eszterházy Károly Egyetem rektora
Megjelent a Líceum Kiadó gondozásában
Kiadóvezető: Dr. Nagy Andor
Felelős szerkesztő: Dr. Domonkosi Ágnes
Műszaki szerkesztő: Dr. Tómás Tibor
Megjelent: 2021. május

Enhanced optimization of high order concentrated matrix-exponential distributions*

Salah Al-Deen Almousa^a, Miklós Telek^{ab}

^aDepartment of Networked Systems and Services
Technical University of Budapest
Budapest, Hungary

^bMTA-BME Information Systems Research Group
Budapest, Hungary
almousa@hit.bme.hu
telek@hit.bme.hu

Submitted: November 23, 2020

Accepted: February 11, 2021

Published online: May 18, 2021

Abstract

This paper presents numerical methods for finding high order concentrated matrix-exponential (ME) distributions, whose squared coefficient of variation (SCV) is very low. Due to the absence of symbolic construction to obtain the most concentrated ME distributions, non-linear optimization problems are defined to obtain high order concentrated matrix-exponential (CME) distributions. The number of parameters to optimize increases with the order in the “full” version of the optimization problem. For orders, where “full” optimization is infeasible ($n > 184$), a “heuristic” optimization procedure, optimizing only 3 parameters independent of the order, was proposed in [6].

In this work we present an enhanced version of this heuristic optimization procedure, optimizing only 6 parameters independent of the order, which results in CME distributions with lower SCV than the existing 3-parameter method. The SCV gain of the new procedure compared to the old one is

*This work is partially supported by the OTKA K-123914 and the NKFIH BME NC TKP2020 projects.

approximately 1.66 and it is almost independent of the order. The range of the applicability of the heuristic optimization methods extends to order $n = 5000$.

To further extend the range of available CME distributions, we also propose a parameter extrapolation approach, which provides CME distributions until order $n = 20000$. The SCV of the obtained order 20000 CME distribution is $\approx 10^{-9}$.

Keywords: Squared coefficient of variation, optimization, concentrated matrix exponential distributions, extrapolation

AMS Subject Classification: 65K10, 90C31

1. Introduction

Highly concentrated matrix-exponential functions are useful in many research areas, for example, in numerical inverse Laplace transform (NILT) methods [5], as well in numerical inverse Z-transform (NIZT) methods [7]. Recently, Akar et al. [1], proposed the ME-fication technique, in which a concentrated matrix exponentiation distribution replaces the Erlang distribution for approximating deterministic time horizons.

Concentrated ME distributions of order N , with $N = 2n + 1$,¹ are abbreviated as CME(N). CME distributions successfully constructed in [6] in the range of $N = 369, \dots, 2001$ are based on a heuristic numerical optimization procedure optimizing 3 parameters independent of the order. This preliminary result indicated that the minimal SCV of CME(N) is less than $1/N^2$. The reasons for applying a heuristic approach are that there is no symbolic construction available to obtain the most concentrated ME distribution, and the full numerical optimization-based approaches (i.e., where the number of parameters to optimize is increasing with N) get to be prohibitively complex for $N > 369$ according to [6]. In this work we aim at improving the heuristic optimization procedure presented in [6], which we refer to as *3-parameter optimization*. The proposed enhanced optimization procedure optimizes 6 parameters (independent of the order) and we will refer to it as *6-parameter optimization method*.

The rest of the paper is organized as follows. In Section 2 we provide a brief introduction of ME distributions and discuss the definition of SCV and the optimization problem to obtain its minimum. In Section 3 we review the optimization methods proposed for SCV minimization in the literature and discuss their applicability. Section 4 introduces the proposed enhanced SCV optimization procedure with 6 parameters and Section 5 discusses its numerical properties. Section 6 presents the parameter extrapolation approach to extend the availability of CME distributions up to order $n = 20000$. Finally, Section 7 concludes the paper.

¹Both of these two order definitions are present in the related literature. N , “the cardinality of the describing matrix”, is more commonly used in phase type and matrix exponential distribution related literature, while n , “the number of complex conjugate eigenvalue pairs” is more commonly used in NILT related literature.

2. Matrix exponential distributions

Definition 2.1. Order N ME functions (referred to as $\text{ME}(N)$) are given by

$$f(t) = \underline{\alpha} e^{\mathbf{A}t} (-\mathbf{A}) \mathbf{1}, \quad (2.1)$$

where $\underline{\alpha}$ is a real row vector of size N , \mathbf{A} is a real matrix of size $N \times N$ and $\mathbf{1}$ is the column vector of ones of size N .

Definition 2.2. If $f(t) \geq 0, \forall t \geq 0$, and $\underline{\alpha}$ is such that $\underline{\alpha} \mathbf{1} = 1$ then $f(t)$ is the probability density function of a ME distribution of order N .

According to (2.1), vector $\underline{\alpha}$ and matrix \mathbf{A} define a matrix exponential function. We refer to the pair $(\underline{\alpha}, \mathbf{A})$ as *matrix representation* in the sequel.

An ME distribution is said to be concentrated when its squared coefficient of variation

$$\text{SCV}(f(t)) = \frac{\mu_0 \mu_2}{\mu_1^2} - 1, \quad (2.2)$$

is low. In (2.2), μ_i denotes the i th moment, defined by $\mu_i = \int_{t=0}^{\infty} t^i f(t) dt$. We note that the SCV according to (2.2) is insensitive to multiplication and scaling, i.e. $\text{SCV}(f(t)) = \text{SCV}(cf(\lambda t))$.

The optimization problem to obtain the minimal SCV of $\text{ME}(N)$ can be formulated as

$$\begin{aligned} & \min_{\underline{\alpha}, \mathbf{A}} \text{SCV}(f(t)) \\ & \text{subject to } f(t) \geq 0, \quad \forall t > 0. \end{aligned}$$

Although matrix-exponential functions have been used for many decades, there are still many questions open regarding their properties. Such an important question is how to decide efficiently if a matrix-exponential function is non-negative for $\forall t > 0$. In general, $f(t) \geq 0, \forall t > 0$ does not necessarily hold for given $(\underline{\alpha}, \mathbf{A})$ representation, and it is rather difficult to check. A potential numerical solution for checking this property is proposed in [9].

Due to the difficulty of checking the constraints of the above constrained optimization problem, its solution is an open problem currently.

3. Concentrated ME distributions

A possible way to simplify the constrained optimization problem is to search for the minimum in a special subset of $\text{ME}(N)$, which is non-negative by construction. Horváth et al. in [6] suggest such a subset which is characterized by

$$f(t) = cf^+(\lambda t), \quad (3.1)$$

where $f^+(t)$ is an exponential cosine-square function with order n defined as

$$f^+(t) = e^{-t} \prod_{j=1}^n \cos^2 \left(\frac{\omega t - \phi_j}{2} \right), \quad (3.2)$$

where $\omega \geq 0$ and $0 \leq \phi_j < 2\pi$ for $j \in \{1, \dots, n\}$.

In [6] the authors conjectured that the density function of the most concentrated ME distribution of order N belongs to this special class of $\text{ME}(N)$, but the validity of this conjecture is not proved even for the smallest non-obvious case, $N = 3$.

An exponential cosine-square function is a non-negative (due to its construction) matrix exponential function and [8, Appendix A] presents how to obtain the matrix representation of size $N = 2n + 1$ associated with $f^+(t)$ in (3.2). Consequently, the set of exponential cosine-square functions of order n is a special subset of $\text{ME}(N)$ (where $N = 2n + 1$).

In this paper, we make use the fact that exponential cosine-square functions can also be represented in the following hyper-exponential form [6]

$$f^+(t) = e^{-t} \prod_{j=1}^n \cos^2 \left(\frac{\omega t - \phi_j}{2} \right) = \sum_{k=0}^{2n} \eta_k e^{-\beta_k t}, \quad t \geq 0, \quad (3.3)$$

where the η_k, β_k ($k = 0, \dots, 2n$) coefficients contain complex conjugate pairs. Generally, calculating the μ_0, μ_1, μ_2 moments based on (3.2), is not an easy task due to computational complexity caused by the product of the cosine square terms. Instead calculating the μ_0, μ_1, μ_2 moments based on (3.3) is much easier since

$$\mu_i = \int_{t=0}^{\infty} t^i \sum_{k=0}^{2n} \eta_k e^{-\beta_k t} dt = \sum_{k=0}^{2n} \frac{i! \eta_k}{\beta_k^{i+1}}, \quad (3.4)$$

3.1. Full optimization of the $f^+(t)$ parameters

In the sequel, we utilize the fact that multiplication and scaling (with c and λ in (3.1)) does not effect the SCV and optimize the SCV of $f^+(t)$ instead of $f(t)$. $f^+(t)$ in (3.2) is defined by $n + 1$ parameters: the frequency ω and the zeros ϕ_j for $j = 1, \dots, n$. Unfortunately, $\text{SCV}(f^+(t))$ is not a simple function of the parameters. For a given n to find $f^+(t)$ with minimal SCV, i.e.

$$\min_{\omega, \phi_1, \dots, \phi_n} \text{SCV}(f^+(t))$$

is still a hard non-linear optimization problem, where the number of parameters to optimize is $n + 1$.

Numerical methods for the solution of this problem are discussed in [6]. The main findings reported there are that evolution strategy based optimization provided the best numerical results. The solution of the problem with the *CMA-ES*

method [4] is fast, but does not find the best optimum compared to the *BIPOP-CMA-ES* method [3], which is much slower. The applicability of the two methods are $n \leq 74$ ($N \leq 149$) in case of the BIPOP-CMA-ES method and $n \leq 184$ ($N \leq 369$) in case of the CMA-ES method. For these orders the respective the optimization procedures take several days to terminate on an average PC. The computational complexity of these procedures increases super linearly with the order n , which inhibits the application of these procedure for higher orders.

3.2. Heuristic optimization of the $f^+(t)$ parameters

To go beyond order $n = 184$, [6] proposed a sub-optimal, 3-parameter heuristic optimization procedure, that reduce the complexity of the optimization problem by reducing the number of parameters to optimize to three, independent of the order.

Figure 2 displays the location of the the ϕ_j parameters obtained by the full optimization method for $n = 74$. As it is visible in the figure, there is a gap between the ϕ_j parameters at around $p \approx 5.2$ and the size of that gap, which is the maximum value in Figure 1 is around $w \approx 0.28$. The heuristic optimization procedure proposed in [6] assumes that the ϕ_j parameters are equidistant below and above that gap. Figure 1 and 2 display how good this assumption is compared to the fully optimized ϕ_j parameters.

Since the ϕ_j parameters are located between 0 and 2π and the number of parameters are n , this assumption allows to determine the ϕ_j parameters based on p and w according to the following expression

$$\phi_j = \begin{cases} (j - 1/2)d & \text{if } j \leq i, \\ (j - 1/2)d + w & \text{if } j > i. \end{cases} \quad (3.5)$$

where

$$d = \frac{2\pi - w}{n}, \quad i = \left\lfloor \frac{p - w/2}{d} + \frac{1}{2} \right\rfloor. \quad (3.6)$$

With the use of (3.5), $f^+(t)$ is defined by the parameters ω , p and w and the related optimization problem is

$$\begin{aligned} & \min_{\omega, p, w} SCV(f^+(t)), \\ & \text{subject to: } 0 < p - w/2 < p + w/2 < 2\pi, \omega > 0. \end{aligned}$$

The solution of this optimization problem is computed by the CMA-ES method up to $n = 1000$ in [6]. Moreover, in this work we expand this solution up to $n = 5000$.

4. Enhanced heuristic optimization of $SCV(f^+(t))$ with 6 parameters

Figure 1 and 2 suggests that the equidistant location of the ϕ_j parameters according to (3.5) is not flexible enough to obtain similar low SCV as obtained with the full optimization method. Starting from this assumption, we try to locate the ϕ_j parameters in a more flexible way. To this end we introduce two different power functions below and above the gap of the ϕ_j parameters as follows

$$\phi_j(a_1, b_1, a_2, b_2, i, \gamma, \delta) = \begin{cases} a_1 + b_1 j^\gamma & \text{for } 1 \leq j \leq i, \\ a_2 + b_2 j^\delta & \text{for } i+1 \leq j \leq n. \end{cases} \quad (4.1)$$

The auxiliary parameters, a_1 , b_1 , a_2 , b_2 , can be transformed to a set of more expressive parameters based on the following relations

$$\phi_1 = 0, \quad \phi_i = p - w/2, \quad \phi_{i+1} = p + w/2, \quad \phi_{n+1} = 2\pi. \quad (4.2)$$

Substituting these relations into (4.1) results in the following function for the ϕ_j parameters

$$\phi_j(p, w, i, \gamma, \delta) = \begin{cases} \frac{(p-w/2)j^\gamma - p-w/2}{(i^\gamma-1)} & \text{for } 1 \leq j \leq i, \\ 2\pi - \frac{(j^\delta - (n+1)^\delta)(2\pi - p - w/2)}{((i+1)^\delta - (n+1)^\delta)} & \text{for } i+1 \leq j \leq n. \end{cases} \quad (4.3)$$

In (4.3) the parameters are constrained by

$$0 < p - w/2 < p + w/2 < 2\pi, \quad \gamma > 0, \quad \delta > 0.$$

The intuitive meaning of the parameters in (4.3) are as follows: the meaning of p , w , ω , and i are the same as in the 3-parameter optimization method, i.e.

- i : is the number of ϕ_j parameters left to the gap,
- p , w : are the midpoint of the gap and its width,

while γ and δ are shape parameters defining the power series of the ϕ_j parameters below and above the gap.

Based on (4.3), which defines the ϕ_j parameters based on 5 parameters, the optimization of $SCV(f^+(t))$ for a given order n is the following 6-parameter optimization problem

$$\begin{aligned} & \min_{\omega, p, w, i, \gamma, \delta} SCV(f^+(t)) \\ & \text{subject to: } 0 < p - w/2 < p + w/2 < 2\pi, \gamma > 0, \delta > 0, \omega > 0, \end{aligned}$$

where p , w , i , γ , δ define the ϕ_j parameters according to (4.3) and the ϕ_j parameters and ω define $f^+(t)$ according to (3.2).

To solve this optimization problem we propose to compute the SCV according to Algorithm 1 and obtain the optimum by the CMA-ES method using Algorithm

1 as the objective function. The procedure to obtain η_i , β_i and the required high precision arithmetic are detailed in [6]. Here we only recall that all computations can be performed with standard double precision arithmetic except the ones indicated to be “high precision”. In those cases, to obtain results in 16 digits precision, the required numerical precision is $0.647n + 17.478$ digits for $\text{ME}(2n + 1)$.

Algorithm 1 The objective function of the heuristic method.

- 1: **procedure** COMPUTESCVC($\omega, p, w, i, \gamma, \delta$)
 - 2: Obtain ϕ_j for $j \in \{1, \dots, n\}$ by (4.3)
 - 3: Compute η_i, β_i (high precision) by (3.3)
 - 4: Compute μ_i (high precision) by (3.4)
 - 5: Compute SCV by (2.2)
 - 6: **return** SCV
 - 7: **end procedure**
-

5. Numerical properties

The behaviour of the ϕ_j parameters obtained by the proposed 6-parameter heuristic method is also depicted in Figure 1 and 2. The figures suggest, that the ϕ_j parameters obtained by the 6-parameter optimization method better approximate the behaviour of the ϕ_j parameters obtained by full optimization than the ones of the 3-parameter method.

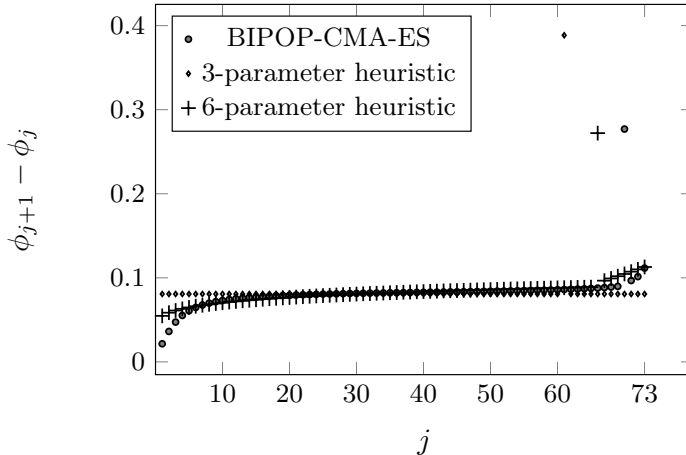


Figure 1. Difference of consecutive ϕ_j values obtained by full optimization, 3-parameter optimization and the proposed 6-parameter optimization for order $n = 74$.

Moreover, Figure 2 displays how the distribution of ϕ_j locations are influenced

by the shaping parameters γ , δ , and getting closer (compared to the 3-parameter case) to the fully optimized ones. This improvement in the positioning of the ϕ_j parameters leads to a significant SCV reduction compared to the 3-parameter method as illustrated in Figure 3.

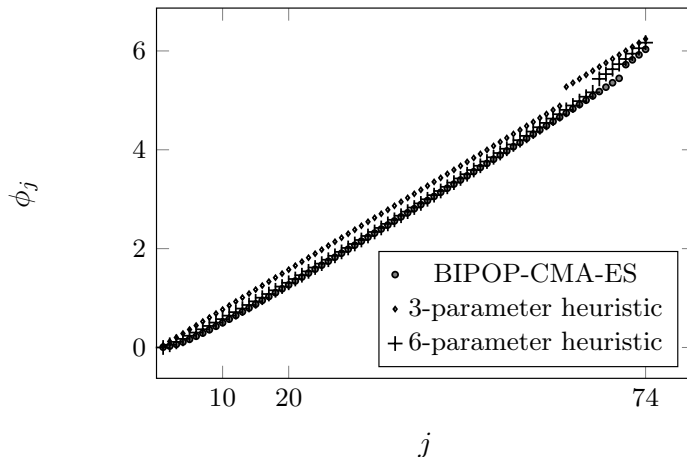


Figure 2. The location of the ϕ_j parameters obtained by full optimization, 3-parameter optimization and the proposed 6-parameter optimization for order $n = 74$.

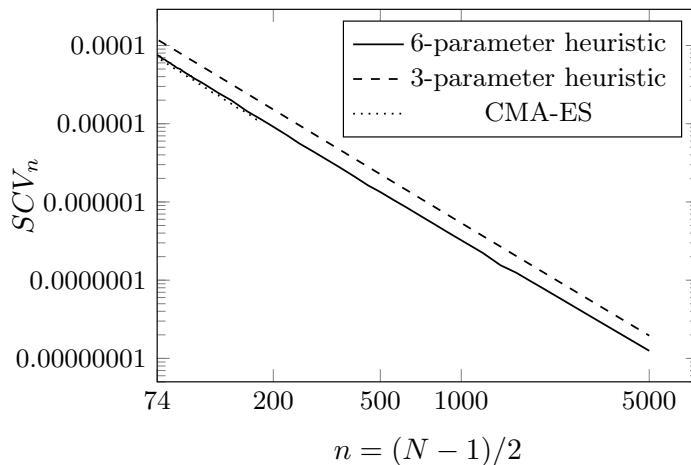


Figure 3. The minimal SCV values obtained by full optimization, 3-parameter optimization and the proposed 6-parameter optimization as a function of order n in log-log scale.

In the depicted range the gain (the ratio of the SCV obtained by the two

methods) is approximately 1.66 and it is almost independent of the order. The proposed heuristic optimization resulted in almost the same SCV values as the ones obtained by the full optimization method in the range where full optimization is feasible and beyond that order ($n > 184$) the 6-parameter optimization results seem to follow the same decay trend.

We believe with some confidence in the possibility of expanding the heuristic optimization for orders larger than $n = 5000$, using a more powerful computing device.

Figure 4 depicts the running time of the heuristic optimization procedure on an average PC clocked at 2.9 GHz as a function of the order n .

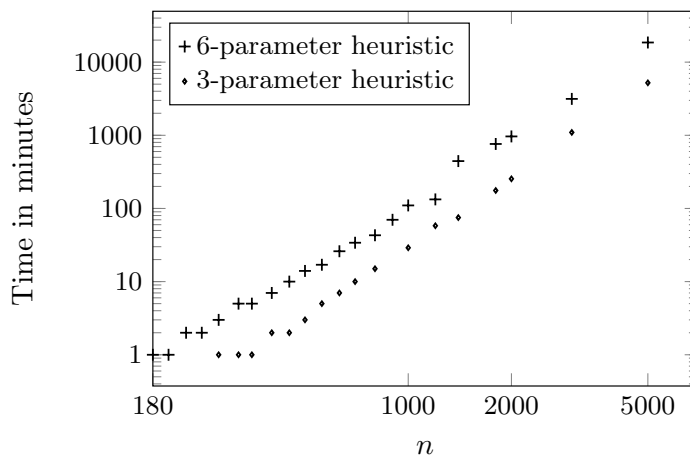


Figure 4. Running time of the heuristic parameter optimization procedures for different orders in log-log scale.

6. Extrapolation of the parameters of heuristic optimization

The high computational costs of the heuristic optimization methods, plotted in Figure 4, inhibits their application for orders higher than $n = 5000$. In this section, we intend to obtain CME distributions for orders $n > 5000$ by extrapolating parameters of the heuristic optimization procedures.

Let $\mathbf{v}(n)$ denote the parameter values obtained from the heuristic optimization method for order n , that is, for the 3-parameter method $\mathbf{v}(n) = \{\omega(n), p(n), w(n)\}$ and for the 6-parameter method $\mathbf{v}(n) = \{\omega(n), p(n), w(n), i(n), \gamma(n), \delta(n)\}$. Furthermore, let $\mathcal{N} = \{n_1, n_2, \dots, n_K\}$ be the set of K orders for which the parameter is available (the heuristic optimization is performed) $n_K = 5000$ in our case and the other evaluated orders are visible in Figure 4.

6.1. Extrapolation methods

To extrapolate the $\mathbf{v}(n)$ vector for $n > 5000$ we considered the following extrapolation approaches.

- Element-wise extrapolation of $\mathbf{v}(n)$

In this set of methods the elements of $\mathbf{v}(n)$ are extrapolated independent of each others.

- Polynomial extrapolation ($k + 1$ parameters):

$$\hat{v}_i(n) = a_i + b_i n + c_i n^2 + \dots + z_i n^k, \quad (6.1)$$

where $v_i(n)$ is the i th element of $\mathbf{v}(n)$ and $i \in \{1, 2, 3\}$ in case of the 3-parameter method and $i \in \{1, 2, \dots, 6\}$ in case of the 6-parameter method.

- Power function extrapolation (3 parameters):

$$\hat{v}_i(n) = a_i n^{b_i} + c_i. \quad (6.2)$$

- Exponential extrapolation (3 parameters):

$$\hat{v}_i(n) = a_i e^{b_i n} + c_i. \quad (6.3)$$

In the element-wise extrapolation, we apply the following distance measure for $\hat{v}_i(n)$

$$\mathcal{D}_i = \frac{1}{K} \sum_{n \in \mathcal{N}} |v_i(n) - \hat{v}_i(n)|. \quad (6.4)$$

That is, in power function and exponential extrapolation, the optimal extrapolation parameters are obtained as

$$\{a_i^*, b_i^*, c_i^*\} = \arg \min_{\{a_i, b_i, c_i\}} \mathcal{D}_i, \quad (6.5)$$

and in polynomial extrapolation, the $\{a_i, b_i, \dots, z_i\}$ parameters are obtained similarly.

- Vector-wise extrapolation of $\mathbf{v}(n)$

- Vector polynomial extrapolation ($(m + k)m$ parameters):

$$\hat{\mathbf{v}}(n) = (\mathbf{a} + \mathbf{b}n + \mathbf{c}n^2 + \dots + \mathbf{z}n^k) \mathbf{G}, \quad (6.6)$$

where each row sum of \mathbf{G} is one (and this way \mathbf{G} contains $(m - 1)m$ free parameters), m is the number of elements of \mathbf{v} . $m = 3$ or 6 depending on the applied heuristic optimization.

- Matrix power function extrapolation ($(m + 2)m$ parameters):

$$\hat{\mathbf{v}}(n) = \mathbf{a}\text{Diag}\langle n^{b_1}, \dots, n^{b_m} \rangle \mathbf{G} + \mathbf{c}. \quad (6.7)$$

- Matrix exponential extrapolation ($(m + 2)m$ parameters):

$$\hat{\mathbf{v}}(n) = \mathbf{a}\text{Diag}\langle e^{b_1 n}, \dots, e^{b_m n} \rangle \mathbf{G} + \mathbf{c}. \quad (6.8)$$

In case of vector-wise extrapolation we apply the L^2 vector norm as the distance measure

$$\mathcal{D} = \frac{1}{K} \sum_{n \in \mathcal{N}} \|\mathbf{v}(n) - \hat{\mathbf{v}}(n)\|_2 = \frac{1}{K} \sum_{n \in \mathcal{N}} \sqrt{\sum_{i=1}^m (\mathbf{v}_i(n) - \hat{\mathbf{v}}_i(n))^2}. \quad (6.9)$$

That is, in Matrix power and Matrix exponential extrapolation, the optimal parameters are obtained as

$$\{\mathbf{a}^*, \mathbf{b}^*, \mathbf{c}^*, \mathbf{G}^*\} = \arg \min_{\{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{G}\}} \mathcal{D} \quad (6.10)$$

and the Matrix polynomial case is optimized similarly according to its parameters.

We note that the results obtained by any of these methods are sensitive for \mathcal{N} , the set of orders which are considered in the parameter estimations. That is, different extrapolation parameters are obtained by the same extrapolation procedure for different \mathcal{N} sets. Generally, we used the optimization results between orders 400 and 5000, that is, $400 \leq n \leq 5000$ for $\forall n \in \mathcal{N}$.

The goodness of an extrapolation approach can be judged by computing the SCV obtained from the extrapolated parameters and checking if the trend of decay for the given order $n > 5000$ follows the trend obtained by the heuristic method for order $n \leq 5000$ and plotted in Figure 3. Based on this goodness measure, we found all extrapolation approaches inappropriate except the element-wise power function extrapolation for all parameters, whose results are presented in the following subsection.

6.2. Element-wise power function extrapolation

Below, we present the results of the element-wise power function extrapolation method which we obtained by the *CF tool* of Matlab Curve Fitting Toolbox [2].

6.2.1. Extrapolation for the 3-parameter heuristic method

As discussed in [6] and in subsection 3.2, the 3-parameter heuristic optimization procedure minimizes the SCV as a function of ω , p , w . The procedure can be applied with reasonable computation time (c.f. Figure 4) up to order $n = 5000$. Beyond this limit we apply the element-wise power function extrapolation according to (6.2), (6.4), and (6.5).

Table 1. The optimal extrapolation parameters for ω , p and w .

	a_i^*	b_i^*	c_i^*	\mathcal{D}_i
$\hat{w}(n)$	25.03	-1.017	0	2.045E-06
$\hat{p}(n)$	-2.691	-0.2467	6.029	3.876E-03
$\hat{\omega}(n)$	0.8919	-0.2399	0.1737	1.377E-05

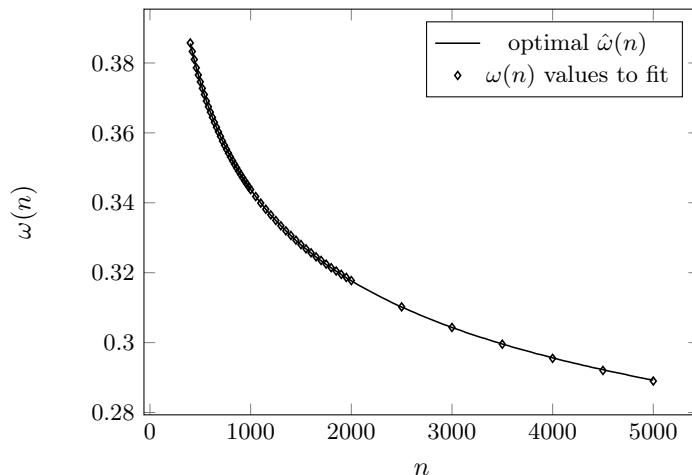
**Figure 5.** Curve fitting for $\omega(n)$ using a power function according to (6.2).

Table 1 summarizes the results for all the three parameters and Figure 5 demonstrate the quality of the obtained result for the ω parameter.

Based on the extrapolation parameters in Table 1 and the associated extrapolation model in (6.2) we can extrapolate the $\omega(n)$, $p(n)$, $w(n)$ for orders larger than 5000. Using those extrapolated $\hat{\omega}(n)$, $\hat{p}(n)$, $\hat{w}(n)$ values, Figure 6 and Table 2 present the associated SCV as a function of the order up to $n = 20000$. Figure 6 and Table 2 indicate that the SCV values obtained by the extrapolation method follow the same decay trend of the heuristic optimization. For orders less than 5000, Table 2 also compares the $\omega(n)$, $p(n)$, $w(n)$ values obtained from the 3-parameter heuristic method, and the $\hat{\omega}(n)$, $\hat{p}(n)$, $\hat{w}(n)$ values provided by the power function extrapolation method. For those orders the SCV value computed by the $\omega(n)$, $p(n)$, $w(n)$ and the $\hat{\omega}(n)$, $\hat{p}(n)$, $\hat{w}(n)$ parameters are identical in their first 3 digits.

Based on Figure 6 and Table 2 we conclude that the extrapolation of the $\hat{\omega}(n)$, $\hat{p}(n)$, $\hat{w}(n)$ parameters with the element-wise power function extrapolation provide fairly concentrated matrix exponential distributions up to order 20000, whose SCV follows the same decay trend for as the one of the 3-parameter heuristic method up to order 5000.

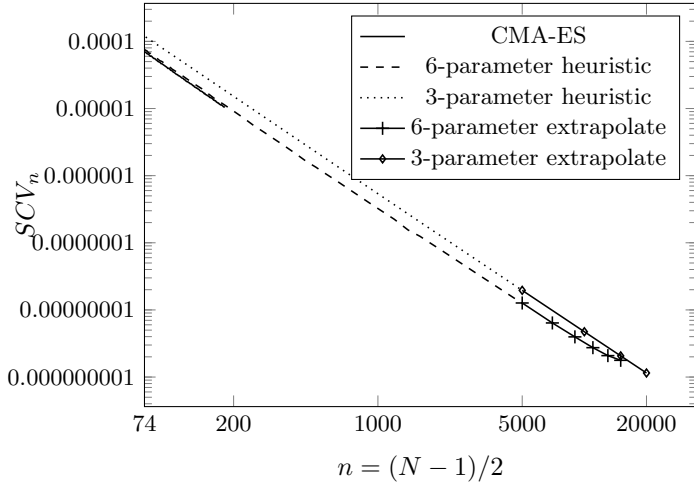


Figure 6. The approximated and the heuristics SCV as a function of order n in log-log scale.

Table 2. Original and extrapolated parameters and the associated SCV for the 3-parameter heuristic optimization.

n	Heuristic Optimization				Extrapolation			
	$w(n)$	$p(n)$	$\omega(n)$	SCV	$\hat{w}(n)$	$\hat{p}(n)$	$\hat{\omega}(n)$	SCV
400	0.0563612	5.4162	0.385725	3.5945E-06	0.0565153	5.41526	0.3855761	3.5947992E-06
800	0.0278559	5.51193	0.353088	8.53737E-07	0.0279266	5.51173	0.3531175	8.538201E-07
1200	.0184659	5.56361	0.336537	3.69091E-07	0.0184899	5.56096	0.3364873	3.691452E-07
1500	0.014703	5.58534	0.328039	2.32831E-07	0.014735	5.58603	0.328002	2.32849E-07
2000	0.0109708	5.61548	0.317745	1.28656E-07	0.010997	5.61638	0.317712	1.28668E-07
2500	0.00874565	5.63895	0.310221	8.12596E-08	0.008765	5.63848	0.310205	8.12665E-08
3000	0.0072661	5.65621	0.304338	5.58463E-08	0.007281	5.65565	0.304363	5.58511E-08
3500	0.00621258	5.66986	0.299541	4.06814E-08	0.006225	5.66958	0.299619	4.06848E-08
4000	0.00542217	5.68131	0.295500	3.09232E-08	0.005434	5.68123	0.295650	3.09269E-08
4500	0.0048099	5.69091	0.292034	2.42819E-08	0.004821	5.69119	0.292252	2.42852E-08
5000	0.00432189	5.69975	0.289008	1.95615E-08	0.004331	5.69986	0.289293	1.9564E-08
10000	-	-	-	-	0.002140	5.75159	0.271585	4.73132E-09
15000	-	-	-	-	0.001417	5.77800	0.262512	2.06641E-09
20000	-	-	-	-	0.001057	5.79519	0.256589	1.14904E-09

6.2.2. Extrapolation for the 6-parameter heuristic method

We applied the same extrapolation approach for the parameters of the 6-parameter heuristic method using the element-wise power function approximation according to (6.2), (6.4), and (6.5). The obtained optimal extrapolation parameter values are summarized in Table 3. Using the associated $\hat{w}(n)$, $\hat{p}(n)$, $\hat{\omega}(n)$, $\hat{i}(n)$, $\hat{\gamma}(n)$, $\hat{\delta}(n)$ functions, we also computed the SCV up to order 15000. The results are plotted in Figure 6. Unfortunately, the SCV values obtained by this 6-parameter

extrapolation methods do not follow the same decay as the one of the 6-parameter heuristic method up to $n = 5000$. At around, $n = 15000$ the SCV obtained from the 6-parameter extrapolation gets to be as high as the one obtained from the 3-parameter extrapolation. Most probable, the reason for this behaviour is the instability caused by the higher number of extrapolated parameters.

Table 3. Optimal extrapolation parameters based on the 6-parameter heuristic optimization method according to (6.2).

	a_i^*	b_i^*	c_i^*	\mathcal{D}_i
$\hat{w}(n)$	21.59	-1.017	0	1.958E-03
$\hat{p}(n)$	-26.35	-0.9762	5.653	1.963E-02
$\hat{\omega}(n)$	0.8952	-0.2458	0.1318	4.484E-03
$\hat{i}(n)$	0.9001	1	-0.7137	1.569
$\hat{\gamma}(n)$	3.631	-0.76579	0.9988	3.98E-03
$\hat{\delta}(n)$	10.48	-0.4475	0.8504	1.393E-01

Figure 7 plots the time to compute the SCV as a function of the order on a regular PC. The computation time is practically identical for both methods because the most expensive step of the computation is to transform the cosine-square form into the hyper-exponential form according to (3.3), which is need in both cases.

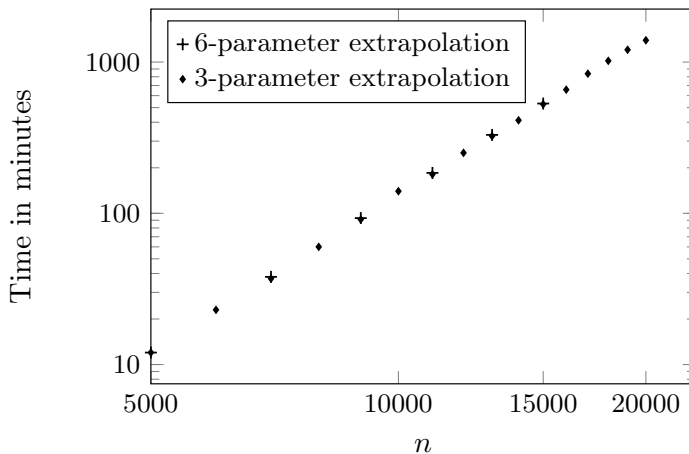


Figure 7. Running time of the parameter approximation procedures for different orders in log-log scale.

Our full C++ implementation for the 3 and 6-parameter heuristic optimization methods is reachable at webspn.hit.bme.hu/~almousa/tools/CME_heur_approx.zip. The procedure uses extended floating point arithmetic when needed and it also contain the CMA-ES method, which is the optimization engine applied in the 3 and 6-parameter heuristic optimization methods

7. Conclusion

We propose an efficient 6-parameter heuristic SCV optimization procedure for concentrated matrix exponential distributions. The SCV values resulted by this 6-parameter optimization procedure are rather close to the ones obtained by the full optimization methods when both methods are feasible to compute, and seem to follow the same SCV decay trend for larger orders. Due to the exponential increase of the computation time as a function of the order, the applicability of the proposed heuristic optimization method extends to order $n = 5000$.

For larger orders, we also propose a parameter extrapolation approach which allowed us to obtain CME distributions up to order 20000, such that the decay of the SCV follows the same trend as the one of the optimization procedures up to order 5000.

References

- [1] N. AKAR, O. GURSOY, G. HORVATH, M. TELEK: *Transient and First Passage Time Distributions of First-and Second-order Multi-regime Markov Fluid Queues via ME-fication*, Methodology and Computing in Applied Probability (2020), pp. 1–27, DOI: <https://doi.org/10.1007/s11009-020-09812-y>.
- [2] *Curve Fitting Toolbox*, <https://www.mathworks.com/help/curvefit>, MATLAB: User's Guide. MathWorks, 2001.
- [3] N. HANSEN: *Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed*, in: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, ACM, 2009, pp. 2389–2396, DOI: <https://doi.org/10.1145/1570256.1570333>.
- [4] N. HANSEN: *The CMA evolution strategy: a comparing review*, in: Towards a New Evolutionary Computation, Springer, 2006, pp. 75–102, DOI: https://doi.org/10.1007/3-540-32494-1_4.
- [5] G. HORVÁTH, I. HORVÁTH, S. A.-D. ALMOUSA, M. TELEK: *Numerical inverse Laplace transformation using concentrated matrix exponential distributions*, Performance Evaluation 137 (2020), p. 102067, DOI: <https://doi.org/10.1016/j.peva.2019.102067>.
- [6] G. HORVÁTH, I. HORVÁTH, M. TELEK: *High order concentrated matrix-exponential distributions*, Stochastic Models 36.2 (2020), pp. 176–192, DOI: <https://doi.org/10.1080/15326349.2019.1702058>.
- [7] I. HORVÁTH, A. MÉSZÁROS, M. TELEK: *Numerical Inverse Transformation Methods for Z-Transform*, Mathematics 8.4 (2020), p. 556, DOI: <https://doi.org/10.3390/math8040556>.
- [8] I. HORVÁTH, O. SÁFÁR, M. TELEK, B. ZÁMBÓ: *Concentrated matrix exponential distributions*, in: European Workshop on Performance Engineering, Springer, 2016, pp. 18–31, DOI: https://doi.org/10.1007/978-3-319-46433-6_2.
- [9] P. REINECKE, M. TELEK: *Does a given vector-matrix pair correspond to a PH distribution*, Performance Evaluation 80.0 (2014), pp. 40–51, DOI: <https://doi.org/10.1016/j.peva.2014.08.001>.

Markov modeling of traffic flow in Smart Cities*

Norbert Bátfai, Renátó Besenczi, Péter Jeszenszky,
Máté Szabó, Márton Ispány

Faculty of Informatics, University of Debrecen, Hungary

`besenczi.renato@inf.unideb.hu`

`jeszenszky.peter@inf.unideb.hu`

`szabo.mate@inf.unideb.hu`

`ispany.marton@inf.unideb.hu`

Submitted: January 15, 2021

Accepted: April 25, 2021

Published online: May 18, 2021

Dedicated to the kind memory of our colleague, Norbert Bátfai.

Abstract

Modeling and simulating the traffic flow in large urban road networks are important tasks. A mathematically rigorous stochastic model proposed in [8] is based on the synthesis of the graph and Markov chain theories. In this model, the transition probability matrix describes the traffic dynamics and its unique stationary distribution approximates the proportion of the vehicles at the segments of the road network. In this paper various Markov models are studied and a simulation method is presented for generating random traffic trajectories on a road network based on the two-dimensional stationary distribution of the models. In a case study we apply our method to the central region of the city of Debrecen by using the road network data from the OpenStreetMap project which is available publicly.

*Renátó Besenczi and Márton Ispány are supported by Project no. TKP2020-NKA-04 which has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the 2020-4.1.1-TKP2020 funding scheme. Péter Jeszenszky and Máté Szabó are supported by the EFOP-3.6.1-16-2016-00022 project. The project is co-financed by the European Union and the European Social Fund.

Keywords: Road network, traffic simulation, discrete time Markov chain, stationary distribution, OpenStreetMap

1. Introduction

Recently, the research and development of Smart City applications have become more important by providing services to inhabitants which can make everyday life easier [15]. These applications are based on emerging technologies such as big data analytics, cloud computing, and complex sensor systems (IoT) that can support their operation. By the year 2050, 70% of Earth's population is expected to live in cities [5] whose infrastructures will face new challenges, e.g., in the field of urban traffic. In the past few years, many developments have occurred in the automobile industry, e.g., autonomous (driverless) and pure electric cars are being introduced. Since more and more people live in urban areas, solutions for problems of dense traffic such as air pollution and congestion are highly demanding [20, 24, 30].

This research presented in this paper follows our development of a traffic simulation platform initiative called rObOCar World Championship (or OOCWC for short) [2, 3]. OOCWC is a multiagent-oriented environment for creating urban traffic simulations. The traffic simulations are performed by one of its components called Robocar City Emulator (RCE), which is an open source software released under the GNU GPL v3 and is available on GitHub.¹ RCE uses the OpenStreetMap (OSM) database and processes it with the Osmium Library. The traffic simulation model of RCE is based on the Nagel-Schreckenberg (NaSch) model [21]. The result of this processing is a routing map graph and a Boost Graph Library graph which can be visualized by various map viewers. For a detailed description of the operation of RCE, see [2]. There exist several traffic simulation platforms, e.g., Multi-Agent Transport Simulation [14], Simulation of Urban Mobility [18], Aimsun,² and PTV Vissim³. The main focus of their simulation algorithms is on microscopic traffic events, while our software system focuses only on the traffic flow on the road network of the whole city.

In [8] a mathematically rigorous stochastic model is proposed for investigating the traffic flow on a road network which is based on the synthesis of discrete time Markov chains and graph theory. In this model the transition probability matrix describes the dynamics of the traffic while its unique stationary distribution corresponds to the traffic equilibrium (or steady) state on the road network. In our previous paper [4], the concepts of Markov traffic and two-dimensional stationary distribution are introduced and a parameter estimation method is proposed by using the weighted least squares (WLS) approach. To investigate complex systems, the joint application of Markov chains and large graphs is well known, see [7, 10, 19].

Our contributions in this paper are as follows. Using the approach in [4], we

¹<https://github.com/nbatfai/robocar-emulator>

²<https://www.aimsun.com/>

³<http://vision-traffic.ptvgroup.com/en-us/products/ptv-vissim/>

present various Markov models for modeling traffic flow on different road graph models based on, e.g., open or closed and digraph or line digraph views. We prove the existence and uniqueness of a stationary distribution as a solution of the global balance equation, see Theorem 3.1. We define the configuration space of Markov traffic, describe the transition mechanism and prove the ergodicity of Markov traffic, see Theorem 4.1. Finally, we propose a simulation method for generating random trajectories for a Markov traffic whose two-dimensional distribution is closest to a prescribed mask matrix in the least squares sense, see Theorem 5.1. The results of this paper together with those obtained in [4], which contains some additional proofs, show that the Markovian approach still works when the scale of the road graph is significantly enlarged compared to such small one as ‘De Uithof’, which is a district in the city of Utrecht in Netherlands, see [11].

Several approaches exist for traffic flow simulation and prediction, some recent surveys are [22, 27, 31], but a few of them are based on Markov models, see [8, 23].

This paper is structured as follows. In section 2 we present various graph models of road networks. Section 3 is devoted to the probability distributions and Markov kernels on road networks. Section 4 introduces the notion of Markov traffic, describes its stationary distribution and proves its ergodicity. A simulation method is presented in section 5. In section 6 we discuss our findings, and in section 7 we conclude the paper. The Appendix provides a toy example and a proof.

2. Graph modeling of road networks

Recall that the ordered pair $G = (V, E)$ is a directed graph (digraph), where V is a finite set of vertices and E is a set of ordered pairs, called directed edges, of vertices. In the sequel, vertices (or nodes) are denoted by u, v, w , edges (or arcs or arrows) are denoted by e, f, g . For a directed edge $e = (v, w) \in E$ we also use the notation $v \rightarrow w$. We suppose that G is a simple digraph, i.e., it does not contain multiple arrows. For details, see the textbook [1].

A road network G is defined as a simple directed graph, $G = (V, E)$, where V is a set of nodes representing the terminal points of road segments, and E is a set of directed edges denoting road segments, see [25]. A road segment $e = (v, w) \in E$ is a directed edge in a road network graph, with two terminal points v and w . The vehicles move on this edge from v to w . The road network G represents the road system of a city.

Let S denote the diagonal set of V , i.e., $S := \{(v, v) | v \in V\}$. From a practical point of view, we suppose that $E \cap S = \emptyset$, i.e., there is no loop $v \rightarrow v$ in the road network in order to avoid that a vehicle is able to move in an infinite cycle. For $v \in V$, define $v^- := \{e \in E | \exists u \in V : e = (u, v)\}$ and $v^+ := \{e \in E | \exists w \in V : e = (v, w)\}$, i.e., v^- and v^+ are the sets of edges in and out the node v , respectively. Then, $\text{deg}^-(v) = |v^-|$ and $\text{deg}^+(v) = |v^+|$ are the indegree and outdegree of node v , respectively.

Let $L(G) = (V', E')$ be the line digraph (line road network, network line graph, see [9]) associated to G , see Section 4.5 in [1]. Here, $V' = E$ and the set E' consists

of the ordered pairs (e, f) where $e, f \in E$ such that there exist $u, v, w \in V$ that $e = (u, v)$ and $f = (v, w)$, i.e., $u \rightarrow v \rightarrow w$ is a path of length 2 (dipath) in G . The elements of E' can be described by triplets (u, v, w) , where $u, v, w \in V$, $(u, v), (v, w) \in E$, and for a directed edge in $L(G)$ we use the notation $(u, v) \rightarrow (v, w)$ too.

The digraph model of a road network assigns the vehicles moving in a city to the vertices (first-order or primal network). Contrarily, the line digraph model assigns the vehicles to the edges (second-order or dual network), see [26, 29]. When we are studying issues that are associated with the crossings (vertices) we will be concerned with the adjacency relationships of crossings, and so with the road network. On the other hand, when we are studying issues that associated with road segments we will be concerned with the adjacency relationships of road segments, and so our analyses will involve the line road network.

The digraphs G and $L(G)$ can be characterized by their degree distributions. The pairs (i, n_i^+) form the frequency histogram for the outdegree distribution of G where $n_i^+ := |\{v \in V \mid \deg^+(v) = i\}|$. The indegree frequency histogram can be defined similarly as (i, n_i^-) , where $n_i^- := |\{v \in V \mid \deg^-(v) = i\}|$. The pairs (i, m_i^+) form the frequency histogram for the outdegree distribution of $L(G)$ where $m_i^+ := \sum_{v \in G_i^+} \deg^-(v)$ and $G_i^+ := \{v \in V \mid \deg^+(v) = i\}$. (Note that $n_i^+ = |G_i^+|$.) Similarly, the pairs (i, m_i^-) form the frequency histogram for the indegree distribution of $L(G)$ where $m_i^- := \sum_{v \in G_i^-} \deg^+(v)$ and $G_i^- := \{v \in V \mid \deg^-(v) = i\}$. For the city of Debrecen (described later in this paper), the above mentioned degree distributions can be seen in Fig. 6. These histograms corroborate the fact that Debrecen's road network is a sparse graph since there is no node with higher in- and outdegree than 4.

Recall that a sequence $v_1, \dots, v_\ell \in V$, $\ell \in \mathbb{N}$, is called walk of length ℓ if $v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_\ell$. A walk is called path if its elements are different vertices. For a pair $u, v \in V$, $u \neq v$, it is said that v is reachable from u if there exists a walk v_1, v_2, \dots, v_ℓ such that $u = v_1$ and $v = v_\ell$. Clearly, if v is reachable from u , then there is a path from u to v . A digraph G is said to be strongly connected (diconnected) if every vertex is reachable from every other vertex. Clearly, the line digraph of a strongly connected digraph is also strongly connected. Namely, if $e = (u, v) \in V' (= E)$ and $f = (w, z) \in V'$ are arbitrary such that $e \neq f$, then, since G is strongly connected, there exists a walk (or a path) of length ℓ in G such that $v = v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_\ell = w$, where $v_1, \dots, v_\ell \in V$, and thus we have $e = (u, v) \rightarrow (v_1, v_2) \rightarrow \dots \rightarrow (v_{\ell-1}, v_\ell) \rightarrow (w, z) = f$, i.e., there exists a walk (or a path) of length ℓ in $L(G)$ between the vertices $e, f \in V'$. If $u \rightarrow v \rightarrow u$ for a pair $u, v \in V$ then we have $(u, v) \rightarrow (v, u) \rightarrow (u, v)$ in the line digraph, i.e., vehicles can turn back at vertex u into v . Sometimes the traffic regulations do not allow this kind of reversal, i.e., the edge set E' in $L(G)$ must not contain some triplet (u, v, u) , while some of these triplets are needed that $L(G)$ be strongly connected. By deleting all of the unnecessary triplets (u, v, u) , $u, v \in V$, such that the remaining line digraph be still strongly connected we get the minimal strongly connected line digraph of G . This line digraph is denoted by $ML(G)$. For example,

the vertices of $ML(G)$ for G in Fig. 1 are given in Table 1.

Recall that a cycle $C \subset V$ in digraph G is a path $v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_\ell \rightarrow v_1$. Here $\ell(C) = \ell$ is called the length of C . A digraph G is said to be aperiodic if the greatest common divisor of the lengths of its cycles is one. Formally, the period of G is defined as $per(G) := \gcd\{\ell > 0 : \exists C \subset V \text{ cycle such that } \ell(C) = \ell\}$. Then, G is called aperiodic if $per(G) = 1$. Clearly, if a digraph G is aperiodic then its line digraph $L(G)$ is also aperiodic. This statement follows from the following fact: if $v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_\ell \rightarrow v_1$ is a cycle then $(v_1, v_2) \rightarrow (v_2, v_3) \rightarrow \dots \rightarrow (v_\ell, v_1) \rightarrow (v_1, v_2)$ is a cycle in $L(G)$. Thus, if $\ell > 0$ and there exists a cycle $C \subset V$ such that $\ell(C) = \ell$ then there exists a cycle $C' \subset V'$ such that $\ell(C') = \ell$.

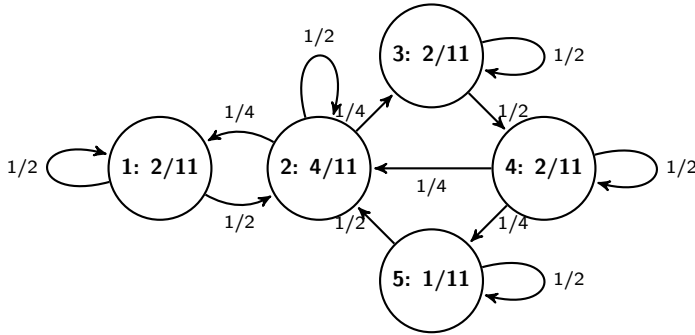


Figure 1. A Markov kernel (on edges) with its stationary distribution (on vertices with node's id) on a simple road network.

Table 1. An example for a Markov kernel on the minimal line digraph of the road network in Fig. 1.

	(1,2)	(2,3)	(3,4)	(4,2)	(2,1)	(4,5)	(5,2)
(1,2)	1/2	1/2	0	0	0	0	0
(2,3)	0	1/2	1/2	0	0	0	0
(3,4)	0	0	1/2	1/4	0	1/4	0
(4,2)	0	1/4	0	1/2	1/4	0	0
(2,1)	1/2	0	0	0	1/2	0	0
(4,5)	0	0	0	0	0	1/2	1/2
(5,2)	0	1/4	0	0	1/4	0	1/2

Let $A = (a_{uv})_{u,v \in V}$ denote the adjacency matrix of the digraph G , i.e., $a_{uv} = 1$ if and only if $(u, v) \in E$ and 0 otherwise. The number of directed walks from vertex u to vertex v of length k is the entry in the u -th row and the v -th column of the matrix A^k . For example, in Fig. 1, the number of directed walks of length 6 from vertex 2 to vertex 4 is 2, see Appendix 7. One can easily check that G is strongly connected if and only if there is a positive integer k such that the matrix $I + A + \dots + A^k$ is positive, i.e., all the entries of this matrix are positive. The

indegree and outdegree of a vertex v can be expressed by the adjacency matrix as $\text{deg}^-(v) = \sum_{u \in V} a_{uv}$ and $\text{deg}^+(v) = \sum_{u \in V} a_{vu}$. Let us introduce the vectors $\mathbf{d}^- := (\text{deg}^-(v))_{v \in V}$ and $\mathbf{d}^+ := (\text{deg}^+(v))_{v \in V}$. Then, we have $\mathbf{d}^- = A^T \mathbf{1}$ and $\mathbf{d}^+ = A \mathbf{1}$ where $\mathbf{1} := (1)_{v \in V}$ is the constant unit function. It is well known that the adjacency matrix A of an aperiodic, strongly connected graph G is primitive, i.e., irreducible and has only one eigenvalue of maximum modulus. Primitivity is equivalent to the following quasi-positivity: there exists $k \in \mathbb{N}$ such that the matrix $A^k > 0$, see Section 8.5 in [13].

In order to model the cases when vehicles leave or enter the city, we augment V by a new ideal vertex 0 and define $\bar{V} := V \cup \{0\}$, see [12]. Moreover, let \bar{E} denote the augmentation of E by directed edges $(0, v)$ and $(v, 0)$ for getting into and out of the city, respectively. Note that, for \bar{E} , it is not allowed to contain the loop $(0, 0)$. The augmentation $\bar{G} = (\bar{V}, \bar{E})$ of G is called the closure of the road network G . For $e = (v, w) \in \bar{E}$ we also use the notation $v \rightarrow w$. In what follows, we suppose that there exist $u, v \in V$ such that $u \rightarrow 0$ and $0 \rightarrow v$.

Each definition, including strong connectedness, periodicity, line digraph, given for G can be extended for \bar{G} in a natural way. Note that in the augmented line digraph $L(\bar{G}) = (\bar{V}', \bar{E}')$ the elements of the edge set \bar{E}' can be described by triplets (u, v, w) , where $u, v, w \in \bar{V}$ and if $v = 0$ then $u, w \neq 0$ and if u or w is 0 then $v \neq 0$ because triplets $(0, 0, v)$, $(v, 0, 0)$, and $(0, 0, 0)$ are excluded from \bar{E}' . One can easily see that if G is strongly connected then its closure \bar{G} is also strongly connected. Moreover, the strongly connected components of G , if there exist more than 1, can be connected through the ideal vertex 0 , resulting in a strongly connected \bar{G} . Thus, the augmented line digraph will also be strongly connected. Clearly, if G is aperiodic then \bar{G} is aperiodic too.

In the rest of this paper, it is assumed that the road network is closed by augmenting with the ideal vertex 0 .

3. Probability distributions and Markov kernels on road networks

On a road network, two kinds of probability distributions can be defined by considering the set V or E as the state space, respectively. However, the Markov kernels on the line road network must be defined with particular care.

A probability distribution (p.d.) on V is the vector $\boldsymbol{\pi} := (\pi_v)_{v \in V}$ where $\pi_v \geq 0$ for all $v \in V$ and $\sum_{v \in V} \pi_v = 1$. We may think of π_v as the proportion of all vehicles which drive through the crossing v with respect to all vehicles in the city. A Markov kernel or transition probability matrix on V is defined as a real kernel $P := (p_{uv})_{u, v \in V}$ such that $p_{uv} \geq 0$ for all $u, v \in V$ and $\sum_{v \in V} p_{uv} = 1$ for all $u \in V$. The quantity $p_{uv} \in [0, 1]$ is called the transition probability from vertex u to vertex v . In fact, P is a stochastic matrix on V and we assume that its support is the set

$E \cup S$. The sum condition for Markov kernel P can be rewritten as:

$$\sum_{w:v \rightarrow w} p_{vw} + p_{vv} = 1, \quad v \in V. \quad (3.1)$$

A p.d. π is a stationary distribution (s.d.) of the kernel P if $\sum_{u \in V} \pi_u p_{uv} = \pi_v$ for all $v \in V$. This so-called global balance equation can be expressed as:

$$\sum_{u:u \rightarrow v} \pi_u p_{uv} + \pi_v p_{vv} = \pi_v, \quad v \in V. \quad (3.2)$$

Fig. 1 presents a Markov kernel with its s.d. on a simple road network.

Since the vehicles are moving along the road segments of the road network G , it is natural to choose E to be the state space. In this case, to define probability distributions on the set of vertices again, we have to consider the line digraph $L(G)$ (or $ML(G)$). Formally, a probability distribution (p.d.) on $L(G)$ is the vector $\pi' := (\pi'_e)_{e \in E}$ where $\pi'_e \geq 0$ for all $e \in E$ and $\sum_{e \in E} \pi'_e = 1$. If we want to emphasize the vertices of the original road network G , instead of the edges, then the notation $\pi'_e = \pi'_{uv}$ is also used where $e = (u, v) \in E$. We may think of π'_e as the proportion of the vehicles at the road segment e with respect to all vehicles in the city. Note that G endowed with π' is a weighted digraph which is often called a network in itself as well.

A transition probability matrix (or Markov kernel) on E , i.e., on the line digraph $L(G)$, can be defined as a real kernel $P' := (p'_{ef})_{e, f \in E}$ such that $p'_{ef} \geq 0$ for all $e, f \in E$ and $\sum_{f \in E} p'_{ef} = 1$ for all $e \in E$. A p.d. π' on E is a s.d. of the kernel P' if $\sum_{e \in E} \pi'_e p'_{ef} = \pi'_f$ for all $f \in E$. Since G represents a road system we may suppose that if $e \neq f$ then $p'_{ef} > 0$ implies that $(e, f) \in E'$, i.e., there exist $u, v, w \in V$ such that $e = (u, v)$ and $f = (v, w)$, and hence, $u \rightarrow v \rightarrow w$ is a walk of length 2. In this case, we use the notation $p'_{ef} = p'_{uvw}$ as well. In fact, p'_{uvw} denotes the probability that a vehicle on the road segment (u, v) will go further to the road segment (v, w) in the next time point. Moreover, in the case of $e = f = (u, v)$, let $p'_{ee} = p'_{uv}$ be the probability that a vehicle remains on the same road segment in the next time point which can be non-zero as well. Thus, since P' is a Markov kernel, we have that, for all $u \rightarrow v$,

$$\sum_{w:v \rightarrow w} p'_{uvw} + p'_{uv} = 1 \quad (3.3)$$

and the global balance equation is given as:

$$\sum_{u:u \rightarrow v} \pi'_u p'_{uvw} + \pi'_v p'_{vv} = \pi'_v \quad (3.4)$$

for all $v \rightarrow w$.

An example for the Markov kernel P' on the minimal line digraph $ML(G)$ of the road network G in Fig. 1 is shown in Table 1. Fig. 2 shows the unique s.d. π' of the Markov kernel P' .

Probability distributions and Markov kernels on the closure \overline{G} of an open road network G can be defined similarly by considering the set \overline{V} or \overline{E} as the state space,

respectively. Note that π_0 denotes the proportion of the number of vehicles which drive in or out of the city's roads at a time point. Moreover, for any Markov kernel P on \bar{V} it is supposed that $p_{00} = 0$, i.e., the vehicles cannot move from 0 to 0, thus they either enter to the road network or leave the road network. Equations (3.1) and (3.2) remain true, too. Equation (3.1) can be rewritten as

$$\sum_{w \in V: v \rightarrow w} p_{vw} + p_{v0} + p_{vv} = 1, \quad v \in V,$$

$$\sum_{w \in V: 0 \rightarrow w} p_{0w} = 1.$$

The global balance equation (3.2) for the s.d. can be rewritten as

$$\sum_{u \in V: u \rightarrow v} \pi_u p_{uv} + \pi_0 p_{0v} + \pi_v p_{vv} = \pi_v, \quad v \in V, \quad 0 \rightarrow v,$$

$$\sum_{u \in V: u \rightarrow v} \pi_u p_{uv} + \pi_v p_{vv} = \pi_v, \quad v \in V, \quad 0 \not\rightarrow v,$$

$$\sum_{u \in V: u \rightarrow 0} \pi_u p_{u0} = \pi_0.$$

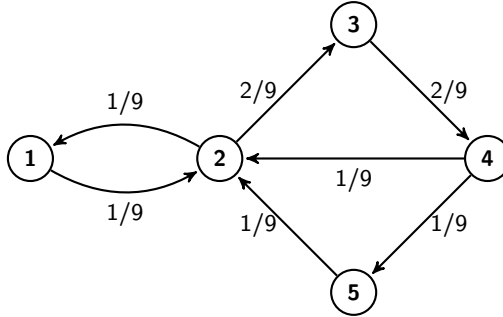


Figure 2. The stationary distribution of the Markov kernel in Table 1.

We can define Markov kernels on the line digraph $L(\bar{G})$ of the augmented road network \bar{G} , and thus on the augmented edge set \bar{E} similarly to the case of $L(G)$. Note that $(e, f) \in \bar{E}$ implies that $e = (u, v)$ and $f = (v, w)$ where $u, v, w \in \bar{V}$ excluding the triplets $(0, 0, v)$, $(v, 0, 0)$, and $(0, 0, 0)$. We shall also use the notation $p'_{uvw} = p'_{ef}$ if $e = (u, v)$ and $f = (v, w)$ and $p'_{uv} = p'_{ee}$ if $e = (u, v)$. However, three additional conditions should be added. The first one is that $p'_{u0u} = 0$ for all $u \in V$ such that $u \rightarrow 0 \rightarrow u$. This means that if a vehicle is on the edge $(u, 0)$, i.e., it leaves the city at vertex u then it cannot be on the edge $(0, u)$ at the next time point, i.e., it cannot enter at vertex u in the road network again, immediately. The second one is that $p'_{0v0} = 0$ for all $v \in V$ such that $0 \rightarrow v \rightarrow 0$, i.e., vehicles can

enter and leave the city at node v . This means that if a vehicle enters the city then it cannot leave the city at the next time point. Finally, the third one is that $p'_{u0} = p'_{0v} = 0$ for all $u, v \in V$ such that $u \rightarrow 0$ and $0 \rightarrow v$. That is a vehicle cannot remain on the road network at the edge $(u, 0)$ after two consecutive time points and if a vehicle enters into the road network at the edge $(0, v)$ (or at the vertex v) the first time then it does not remain on this edge after the next time point and it goes further immediately in the road network. Under these conditions, equations (3.3) and (3.4) remain true. Equation (3.3) can be rewritten as:

$$\begin{aligned} \sum_{w \in V: v \rightarrow w} p'_{uvw} + p'_{uv0} + p'_{uv} &= 1, \quad u, v \in V, u \rightarrow v, \\ \sum_{w \in V: v \rightarrow w} p'_{0vw} &= 1, \quad v \in V, 0 \rightarrow v, \\ \sum_{v \in V \setminus \{u\}: 0 \rightarrow v} p'_{u0v} &= 1, \quad u \in V, u \rightarrow 0. \end{aligned}$$

Equation (3.4) can be rewritten as:

$$\begin{aligned} \sum_{u \in V: u \rightarrow v} \pi'_{uv} p'_{uvw} + \pi'_{0v} p'_{0vw} + \pi'_{vw} p'_{vw} &= \pi'_{vw}, \quad v, w \in V, v \rightarrow w, \\ \sum_{u \in V: u \rightarrow v} \pi'_{uv} p'_{uv0} + \pi'_{v0} p'_{v0} &= \pi'_{v0}, \quad v \in V, v \rightarrow 0, \\ \sum_{u \in V \setminus \{w\}: u \rightarrow 0} \pi'_{uw} p'_{u0w} + \pi'_{0w} p'_{0w} &= \pi'_{0w}, \quad w \in V, 0 \rightarrow w. \end{aligned}$$

The s.d. in all cases, i.e., for Markov kernels on road networks, line road networks and their closures, can be derived by solving the above appropriate linear equations numerically. It turns out that there is a direct connection between the existence and uniqueness of s.d. of the Markov kernels P and P' and the strongly connected property of the physical road network G if the Markov and graph structures are compatible with each other.

The Markov kernel P on V is called G -compatible if, for any $u, v \in V$ such that $u \neq v$, $p_{uv} > 0$ if and only if $(u, v) \in E$. Similarly, the Markov kernel P' on E is called G -compatible if it is $L(G)$ -compatible Markov kernel on $L(G)$, i.e., for any $e, f \in E$ such that $e \neq f$, $p'_{ef} > 0$ if and only if $(e, f) \in E'$. This is equivalent to the statement that $p'_{uvw} > 0$, $u, v, w \in V$, if and only if $(u, v), (v, w) \in E$. Since $(e, f) \in E'$ if and only if there exist $u, v, w \in V$ such that $e = (u, v)$ and $f = (v, w)$ we can define the G -compatibility of a Markov kernel P' as, for any $e, f \in E$ such that $e \neq f$, $p'_{ef} > 0$ if and only if there exist $u, v, w \in V$ such that $e = (u, v)$ and $f = (v, w)$.

Clearly, if P is G -compatible then the strong connectivity of G implies that the Markov kernel (the transition matrix) P is irreducible. Thus, by Theorem 1 in [16], see also Theorem 3.1 and 3.3 in Chapter 3 of [6] the following theorem holds.

Theorem 3.1. *If a road network G is strongly connected then there is a unique stationary distribution π (π') to any G -compatible Markov kernel P (P'). Moreover, this distribution satisfies $\pi_v > 0$ for all $v \in V$ ($\pi'_{uv} > 0$ for all $(u, v) \in E$).*

The main consequence of this theorem is that, in case of any physical road network augmented by the ideal vertex 0, all of the Markov kernels defined on the road network that has positive transition probability on all roads have unique s.d.

4. Markov traffic on road networks

Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space. A sequence $\{X_t\}_{t \in \mathbb{Z}_+}$ of V -valued r.v.'s is a Markov chain on the state space V if the Markov property holds:

$$\mathbb{P}(X_t = v_t | X_{t-1} = v_{t-1}, \dots, X_0 = v_0) = \mathbb{P}(X_t = v_t | X_{t-1} = v_{t-1})$$

for all $t \in \mathbb{N}$, $v_0, \dots, v_t \in V$. If X, X' are V -valued r.v.'s then for the conditional distribution $P = (p_{vv'})_{v, v' \in V}$, $p_{vv'} := \mathbb{P}(X = v | X' = v')$, $v, v' \in V$, we shall also use the notation $X|X'$. Clearly, $X|X'$ is a Markov kernel on V . Similarly, a Markov chain $\{Y_t\}_{t \in \mathbb{Z}_+}$ of E -valued r.v.'s can also be defined through the Markov kernel $Y|Y'$ on the state space E .

In what follows, we suppose that the road network G is strongly connected and the Markov kernel P is G -compatible on V with unique s.d. π . The Markov chain $\{X_t\}_{t \in \mathbb{Z}_+}$ on V is called Markov random walk on the road network G with Markov kernel P if for its initial distribution $\pi_{X_0} = \pi$ and transition probabilities $X_t | X_{t-1} \sim P$ for all $t \in \mathbb{N}$. The set of k ($k \in \mathbb{N}$) mutually independent Markov random walks on G with Markov kernel P is called Markov traffic of size k and it is denoted by the quadruple (G, P, π, k) . Similarly, $\{Y_t\}_{t \in \mathbb{Z}_+}$ is a Markov random walk on the line road network if it is a Markov chain on the state space E such that $\pi'_{Y_0} = \pi'$ and $Y_t | Y_{t-1} \sim P'$ for all $t \in \mathbb{N}$.

A Markov random walk is the movement of a random vehicle which follows the stochastic rules defined by the Markov kernel. For a pair $u, v \in V$, the notation $u \Rightarrow v$ means that $(u, v) \in E \cup S$, i.e., either $u \rightarrow v$ or $u = v$. One can see that $X_t \Rightarrow X_{t+1} \Rightarrow \dots \Rightarrow X_{t+n}$ for all t and $n \in \mathbb{N}$. $\{X_t\}_{t \in \mathbb{Z}_+}$ is also called a first-order random walk on the road network where a vehicle moves from vertex u to vertex v with probability p_{uv} . On the other hand, $\{Y_t\}_{t \in \mathbb{Z}_+}$ may be referred as a second-order random walk where the vehicles move from edge to edge, i.e., we have to consider where the vehicle came from, the vertex visited before the current vertex. The second-order random walk has also been considered in graph analysis, see [29].

The state space of a first-order Markov traffic can be modeled by the function space \mathcal{F} where $\mathbf{f} \in \mathcal{F}$ is a non-negative integer valued function on V , i.e., $\mathbf{f} = (f_v)_{v \in V}$ such that $f_v \in \{0, 1, 2, \dots\}$ for all $v \in V$. The function \mathbf{f} is called a traffic configuration or a counting function and f_v measures the number of vehicles at vertex $v \in V$. Let $|\mathbf{f}|$ denote the size of the traffic configuration \mathbf{f} defined by $|\mathbf{f}| := \sum_{v \in V} f_v$. The size of a traffic configuration counts the number of vehicles on the road network at a time. Let \mathcal{F}_k ($k \in \mathbb{N}$) denote the subset of traffic

configurations of size k . A p.d. ϱ on \mathcal{F} is a function $\varrho : \mathcal{F} \rightarrow [0, 1]$ such that $\sum_{\mathbf{f}} \varrho(\mathbf{f}) = 1$. For a p.d. π on the road network G , let ϱ denote a multinomial distribution on \mathcal{F}_k with parameters k and π , see Chapter 35 in [17]. Thus, we have

$$\varrho(\mathbf{f}) := k! \prod_{v \in V} \frac{\pi_v^{f_v}}{f_v!} \quad (4.1)$$

for all $\mathbf{f} \in \mathcal{F}_k$. In fact, ϱ is the k -fold convolution of π . By formula (4.1) the probability of any complex event of the traffic can be computed.

A Markov kernel R on \mathcal{F}_k is a function $\mathcal{F}_k \times \mathcal{F}_k \rightarrow [0, 1]$ such that, for all $\mathbf{f} \in \mathcal{F}_k$, $\sum_{\mathbf{g} \in \mathcal{F}_k} R(\mathbf{f}, \mathbf{g}) = 1$. We demonstrate that every Markov kernel P induces a natural Markov kernel on \mathcal{F}_k . The matrix $K = (k_{uv})_{u,v \in V}$ is called transport matrix from traffic configuration \mathbf{f} to \mathbf{g} on the road network G if $K : V \times V \rightarrow \mathbb{N}_0$ such that $k_{uv} > 0$ implies $u \Rightarrow v$, $\sum_{v \in V} k_{uv} = f_u$ for all $u \in V$, and $\sum_{u \in V} k_{uv} = g_v$ for all $v \in V$. In fact, K has row and column marginals \mathbf{f} and \mathbf{g} , respectively, and, heuristically, K defines a way for transporting the vehicles from configuration \mathbf{f} into \mathbf{g} on the road network. An example for a transport matrix can be seen in Fig 3. For a pair $\mathbf{f}, \mathbf{g} \in \mathcal{F}_k$ let $\mathcal{M}(\mathbf{f}, \mathbf{g})$ denote the set of all transport matrices from \mathbf{f} to \mathbf{g} . Define the Markov kernel R on \mathcal{F}_k in the following way:

$$R(\mathbf{f}, \mathbf{g}) := \prod_{u \in V} f_u! \sum_{K \in \mathcal{M}(\mathbf{f}, \mathbf{g})} \prod_{u,v: u \Rightarrow v} \frac{p_{uv}^{k_{uv}}}{k_{uv}!} \quad (4.2)$$

where $\mathbf{f}, \mathbf{g} \in \mathcal{F}_k$. Then, R maps a p.d. ϱ into the p.d. $R\varrho$ on the state space \mathcal{F}_k in the following way:

$$(R\varrho)(\mathbf{g}) := \sum_{\mathbf{f} \in \mathcal{F}_k} \varrho(\mathbf{f}) R(\mathbf{f}, \mathbf{g}) \quad (4.3)$$

for all $\mathbf{g} \in \mathcal{F}_k$. To check that R is a Markov kernel indeed we note that, by the multinomial theorem,

$$\sum_{\mathbf{g} \in \mathcal{F}_k} R(\mathbf{f}, \mathbf{g}) = \prod_{u \in V} f_u! \sum_{\substack{\sum_{v \in V} k_{uv} = f_u \\ u,v: u \Rightarrow v}} \prod_{u,v: u \Rightarrow v} \frac{p_{uv}^{k_{uv}}}{k_{uv}!} = \prod_{u \in V} \left(\sum_{v \in V} p_{uv} \right)^{f_u} = 1. \quad (4.4)$$

Moreover, one can easily see similarly to (4.4), by the multinomial theorem, that if π is a s.d. of the Markov kernel P , then the p.d. ϱ defined by (4.1) is the s.d. of the induced Markov kernel R defined by (4.2). Namely, we have the global balance equation

$$\sum_{\mathbf{f} \in \mathcal{F}_k} \varrho(\mathbf{f}) R(\mathbf{f}, \mathbf{g}) = \varrho(\mathbf{g}) \quad (4.5)$$

for all $\mathbf{g} \in \mathcal{F}_k$. (For the proof see Appendix.)

Note that the concepts of traffic configuration and induced Markov kernel on them can be extended to the case of second-order Markov traffic by using the function space of non-negative integer valued functions on E as state space.

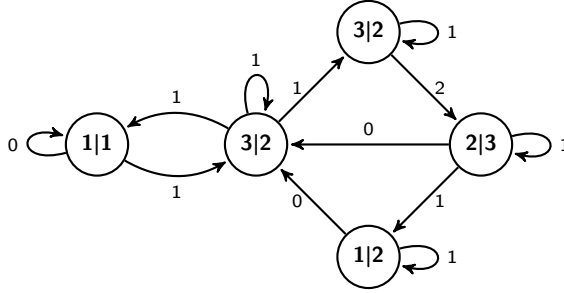


Figure 3. A transport matrix (on edges) on the road network in Fig. 1 from configuration $\mathbf{f} = (1, 3, 3, 2, 1)$ (left in vertices) to configuration $\mathbf{g} = (1, 2, 2, 3, 2)$ (right in vertices) with $k = 10$.

The applicability of the Markov traffic model is based on its ergodicity. Let ϱ_0 be an initial p.d. on \mathcal{F}_k and let us define the n th absolute p.d. ϱ_n on \mathcal{F}_k by the recursion $\varrho_n := R\varrho_{n-1}$, $n \in \mathbb{N}$, where R is a Markov kernel on \mathcal{F}_k induced by a G -compatible Markov kernel P on G , see formula (4.2). One can prove that the irreducibility and aperiodicity of P imply the same properties for R , respectively.

Our main result on ergodicity of Markov traffic, which follows from the ergodicity of irreducible aperiodic Markov chains, is the following theorem. Note that the n th power of R is defined recursively as $R^n \varrho := R(R^{n-1} \varrho)$, $n = 2, 3, \dots$, by formula (4.3).

Theorem 4.1. *Let G be a strongly connected and aperiodic road network and P be a G -compatible Markov kernel. Then, there is a unique stationary distribution ϱ to the Markov traffic described by the Markov kernel R on \mathcal{F}_k induced by P which has the form (4.1).*

Moreover, the Markov traffic is ergodic in the sense that we have

$$R^n(\mathbf{f}, \mathbf{g}) \rightarrow \varrho(\mathbf{g})$$

as $n \rightarrow \infty$ for all $\mathbf{f}, \mathbf{g} \in \mathcal{F}_k$ and, for all initial p.d. ϱ_0 on \mathcal{F}_k ,

$$\varrho_n(\mathbf{f}) \rightarrow \varrho(\mathbf{f})$$

as $n \rightarrow \infty$ for all $\mathbf{f} \in \mathcal{F}_k$.

By the ergodic theorem, Theorem 4.1 implies that the p.d. π on G can be unfolded by the limit of state space averages in time as

$$\frac{1}{k} \sum_{\mathbf{f} \in \mathcal{F}_k} f_v \varrho_n(\mathbf{f}) \rightarrow \pi_v$$

as $n \rightarrow \infty$ for all $v \in V$. This formula follows from the well-known fact that the expectation vector of a multivariate distribution with parameters k and π is equal

to $k\pi$, see formula (35.6) in [17]. Similar results hold for any G -compatible Markov kernel P' on $V' = E$.

These results guarantee that the unique s.d. of a G -compatible Markov kernel can be approximated and thus explored by long run behavior of absolute p.d.'s on the traffic configurations of the road network.

5. Simulation by two-dimensional stationary distribution

A Markov traffic can be reparametrized by using its two-dimensional stationary distribution. Let us define the two-dimensional distribution $Q = (q_{uv})$ on $V \times V$ as $q_{uv} := \pi_u p_{uv}$, $u, v \in V$. One can see that Q satisfies the following properties: (i) $q_{uv} \geq 0$ for all $u, v \in V$ and $q_{uv} = 0$ for all $u, v \in V$ such that $(u, v) \notin E \cup S$; (ii) $\sum_{u, v \in V} q_{uv} = 1$ (i.e., Q is a normalized matrix on V); and (iii) $\sum_{v \in V} q_{uv} = \sum_{v \in V} q_{vu}$ for all $u \in V$ (i.e., Q has equidistributed marginals). Q is called the two-dimensional stationary distribution (2D s.d.) of the Markov traffic. Clearly, if P is G -compatible, then Q is positive on E , i.e., $q_{uv} > 0$ for all $(u, v) \in E$.

Q can also be considered as a p.d. on the state space $E \cup S$, i.e., if we extend the set V' of vertices of $L(G)$ as $V' = E \cup S$, on the line road network. Thus, we can think of Q as the distribution of the vehicles on the edges of the road network, see formula (11) in [8]. The distribution Q , similarly to traffic trajectories, can also be visualized on the edges, see Fig. 8.

For a positive Q on E , let us define

$$\begin{aligned} \pi_u &:= \sum_{v \in V} q_{uv} = \sum_{v \in V} q_{vu}, \quad u \in V, \\ p_{uv} &:= \frac{q_{uv}}{\pi_u}, \quad u, v \in V. \end{aligned} \tag{5.1}$$

Note that $\pi_v > 0$ for all $v \in V$ by Theorem 3.1. Then, $P = (p_{uv})$ defines a G -compatible Markov kernel with s.d. π on G . Thus, a Markov traffic defined by the quadruple (G, P, π, k) can be introduced by an equivalent way through the triplet (G, Q, k) .

With the help of 2D s.d., we can assign a p.d. to any Markov traffic on the space of traffic configurations which are defined on the edges of the road network. Namely, let the traffic configuration $\mathbf{h} = (h_{uv})_{u \Rightarrow v}$ be a non-negative integer valued function on $E \cup S$. Here, h_{uv} denotes the number of vehicles on the edge (u, v) where $u, v \in V$ such that $u \Rightarrow v$. We define the two-dimensional distribution σ on the set of traffic configurations \mathbf{h} with size k ($k \in \mathbb{N}$), i.e., where $\sum_{u \Rightarrow v} h_{uv} = k$. Similarly to (4.1), the two-dimensional distribution σ induced by a p.d. π on G as its k -fold convolution has a multinomial distribution with parameter k and Q , i.e., for all \mathbf{h} , we have

$$\sigma(\mathbf{h}) := k! \prod_{u \Rightarrow v} \frac{q_{uv}^{h_{uv}}}{h_{uv}!}.$$

In fact, σ describes the 2D s.d. of a Markov traffic with size k . One can easily see that the concept of 2D s.d. can also be extended for the second-order Markov traffic.

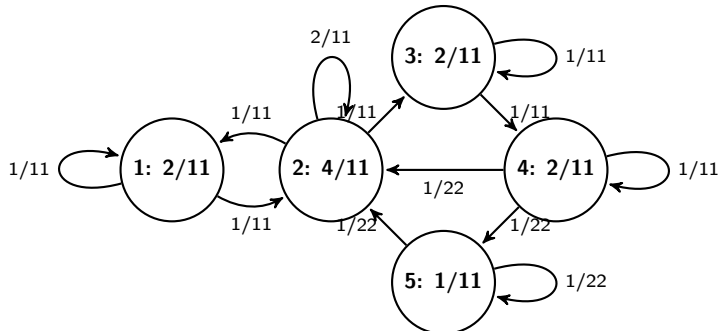


Figure 4. The two-dimensional stationary distribution (on edges) with its equidistributed marginals (on vertices) for the Markov kernel in Fig. 1. One can easily check that the sums of probabilities written on the edges in and out each vertex are equal, respectively.

The simulation algorithm presented in this paper is based on the 2D s.d. defined on the road graph. However, it is not an easy task to find a matrix Q which satisfies properties (i)-(iii) on a sparse graph. Hence, at first, we propose a method for finding such Q which is closest to a given mask matrix M on G in the least square sense. The role of the mask matrix is to specify the weight of edges by modeling the odds of consecutive occurrences of cars on the terminal points of edges in the road network. For example, these weights may stem from observed trajectories for the traffic in a time period.

Let us observe a random sample of trajectories $\{X^i\}$, $i = 1, \dots, k$, of size k defined by $X_1^i \Rightarrow X_2^i \Rightarrow \dots \Rightarrow X_{n_i}^i$, $i = 1, \dots, k$, where n_i denotes the length of the i th trajectory. The total sample size is given by $n := n_1 + \dots + n_k$. Define the total two-dimensional consecutive empirical frequencies as:

$$n_{uv} := \sum_{i=1}^k \sum_{j=1}^{n_i-1} I(X_j^i = u, X_{j+1}^i = v), \quad (5.2)$$

$u, v \in V$, where I denotes the indicator function. Plainly, n_{uv} is the number of consecutive pairs (u, v) ($u, v \in V$) in the trajectories. One can see that the support of the two-dimensional frequency matrix $N := (n_{uv})_{u,v \in V}$ is a subset of $E \cup S$. Clearly, $\mathbf{1}^\top N \mathbf{1} = n - k$, where $n - k$ is the corrected sample size. One can also see that the vectors $N^\top \mathbf{1} - N \mathbf{1}$ and $\mathbf{1}$ are orthogonal. In this case, the matrix N is a good candidate for the role of the mask matrix M .

We define the optimality criteria for determining Q by means of the least squares distance between matrices over G . Let $A = (a_{uv})_{u,v \in V}$ and $B = (b_{uv})_{u,v \in V}$ such that $a_{uv} = b_{uv} = 0$ for all $u, v \in V$ where $u \not\Rightarrow v$. The least square distance between

A and B is defined as

$$\|A - B\|_G^2 := \sum_{u,v:u \Rightarrow v} |a_{uv} - b_{uv}|^2.$$

In fact, $\|\cdot\|_G$ is the Frobenius norm of the matrices of dimension $|V| \times |V|$ which vanish on the entries outside of $E \cup S$.

To formulate our main result, we need some basic facts on the spectral theory of directed graphs, see [28] for details. The symmetric unnormalized graph Laplacian matrix L of a digraph G is defined as $L := D - A - A^\top$, where A denotes the adjacency matrix of G and $D := \text{diag}\{\mathbf{d}^+ + \mathbf{d}^-\}$.

Theorem 5.1. *Let M be a non-negative matrix on G . Then, there is a unique pair (Q, \varkappa) , where the matrix Q on G satisfies properties (i)-(iii) and $\varkappa \geq 0$, which minimizes the error function $\|\varkappa Q - M\|_G^2$. Moreover, the unique solution to this optimization problem is derived as*

$$\begin{aligned} \varkappa &:= \mathbf{1}^\top M \mathbf{1} + (\mathbf{d}^- - \mathbf{d}^+)^\top \boldsymbol{\lambda}, \\ Q &:= \varkappa^{-1} (M + (\mathbf{1} \boldsymbol{\lambda}^\top - \boldsymbol{\lambda} \mathbf{1}^\top) \circ A), \end{aligned}$$

where $\boldsymbol{\lambda} = (\lambda_v)_{v \in V}$ is called Lagrange vector and defined as a unique solution to the vector linear equation $L \boldsymbol{\lambda} = (M - M^\top) \mathbf{1}$ which satisfies the constraint $\mathbf{1}^\top \boldsymbol{\lambda} = 0$ (i.e., $\sum_{v \in V} \lambda_v = 0$), and \circ denotes the entrywise (Hadamard) product of matrices.

The proof of Theorem 5.1 is based on the Lagrange method, see Appendix in [4]. One can easily see that the error function at the optimum equals to the sum of squared differences (SSD) of the Lagrange vector defined by

$$\text{SSD} := \sum_{u \rightarrow v} (\lambda_u - \lambda_v)^2.$$

The fundamental statement of Theorem 5.1, as one of the main results of this paper, is that the optimal 2D s.d. Q is a low-dimensional perturbation of the mask matrix M . This perturbation term and the normalizing constant \varkappa depend on two components through a unique solution to a vector linear equation. The coefficient matrix of the linear equation is the Laplacian matrix L of the road graph which depends only on the graph structure of the road network and independent from the mask matrix. Thus, L can be computed and stored in advance for a given road network. Contrarily, the constant vector of the linear equation depends only on the marginals of the mask matrix, however, it does not depend on its entries and mainly on the road network itself.

After having defined or determined a 2D s.d. Q on a road network G , a simple simulation algorithm for generating random trajectories on G is the following. A trajectory t of length ℓ is a generalized path $v_0 \Rightarrow v_1 \Rightarrow \dots \Rightarrow v_{\ell-1}$, $v_i \in V$, $i = 0, 1, \dots, \ell - 1$, which is stored in an ordered list as $t = [v_0, v_1, \dots, v_{\ell-1}]$. Note that $v_i = v_{i+1}$ is also allowed for any index i , i.e., a vehicle may stay in place after a timestep. The temporary set of generated trajectories is stored in a dictionary D

which consists of key-value pairs (v, T_v) . Here, the key $v \in V$ identifies a node in the road graph, and the value $T_v = [t_0, t_1, \dots, t_{n_v-1}]$ is an ordered list of trajectories $t_j, j = 0, 1, \dots, n_v - 1$, of length n_v such that the last element of all trajectories in T_v is v , i.e., the trajectories end at the node v . In fact, T_v is a list of lists for each $v \in V$. Let T denote the final set of trajectories as the output of the algorithm. By generating random pairs (u, v) from Q successively, D is updated, and then T is derived in the following way. If T_u is not empty, then let the trajectory t be given by appending v to the first trajectory in T_u . Moreover, let us delete this trajectory from the list T_u . If the length of t is large enough, then let us add it to T , otherwise add it to the list T_v . If T_u was empty then append the list $[u, v]$ to T_v .

Algorithm 1: Trajectory simulation.

Input: Q : two-dimensional stationary distribution
 m : maximum trajectory length
 n : number of simulated consecutive pairs

Output: T : list of trajectories

```

/* initialization */
D = {}; /* temporary dictionary */
T = [];
/* iterating over simulated pairs */
for i = 1 to n do
    pick a random pair  $(u, v) \sim Q$ ;
    if  $D[u]$  is not empty then
         $t = D[u][0]$ ; /* temporary trajectory */
        append node  $v$  to  $t$ ;
        delete the first element of  $D[u]$ ;
    else
         $t = [u, v]$ ;
    end
    if  $length(t) = m$  then
        append  $t$  to  $T$ ;
    else
        if  $v \in D$  then
            append  $t$  to  $D[v]$ ;
        else
            append  $(v, t)$  to  $D$ ;
        end
    end
end
end
/* appending the trajectories in temporary dictionary to the output */
for v in D do
    append  $D[v]$  to  $T$ ;
end
end

```

Having finished the random generation of pairs, let us append the trajectories of whole D to the final set T . One can easily see that the longer trajectories are at the head of T . A pythonic pseudo-code of the above procedure is in Algorithm 1. After the simulation, the generated trajectories can be visualized by using a digital map system, e.g., Google Maps or OpenStreetMap. Finally, we note that, in a typical step of the algorithm, a trajectory moves from the first position of a trajectory list to the last position of an other one. This is a kind of mixing which helps to avoid the formation of very unbalanced trajectories.

6. Results

In our work, OpenStreetMap (OSM) was used which is a community project to build a free map of the world. OSM data is available under the Open Data Commons Open Database License (ODbL). The representation and storing of map data is based on only three modeling primitives: nodes, ways, and relations.⁴ A node represents a geographical entity with GPS coordinates. A way is an ordered list of at least two nodes. A relation is an ordered list of nodes, ways, and/or relations. Users can export map data at the OSM web site manually, selecting a rectangular region of the map. OSM uses OSM XML and PBF formats for exporting map data. Software libraries for parsing and working with OSM data are available for several programming languages.⁵

We started our processing by building a graph from the OSM map of Debrecen in the bounding box defined by the coordinates N47.4771, W21.5565, N47.571, W21.6918, see Fig. 5. Because we only need those nodes that can be reached by vehicles, we had to filter the OSM file and collect only specific types of way nodes. In the OSM file, a way is a sequence of OSM nodes, so naturally, the nodes of ways become nodes in the graph. For every node we store the node's OSM ID and its coordinates. We also insert an edge into the graph to connect each pair of nodes that follow each other in a way. We used the PyOsmium library for processing the OSM files and the NetworkX Python library for building the graph. The result of this processing is an aperiodic strongly connected road network of Debrecen augmented by the ideal vertex 0. The descriptive statistics of edges of the road graph are: Min=0.3395, Q1=10.7906, Med=24.7830, Mean=49.9052, Q3=67.6021, Max=1167.4902 (in meters). The degree distributions of this road network are visualized in Fig. 6.

To evaluate the performance of the proposed algorithm a simple simulation study was conducted at different sample sizes for the road network of Debrecen. In the simulations, we kept the length of trajectories low and the number of trajectories high compared to the size of the road network. By our experience, the real traffic trajectories possess these properties. All simulations were carried out in Python. The codes and datasets of our simulation are available upon request.

We have also implemented the model in the OOCWC system. Regarding RCE,

⁴<http://wiki.openstreetmap.org/wiki/Elements>

⁵<https://wiki.openstreetmap.org/wiki/Frameworks>

we have performed several modifications. First, we extended the operation of RCE to be able to handle kernel files for transition probability matrices and 2D stationary distributions, respectively. These kernel files can be loaded to the RCE software, so all nodes of the simulation graph will have the corresponding transition probability vector from the Markov kernel file. For this, we had to extend the shared memory segment of RCE.

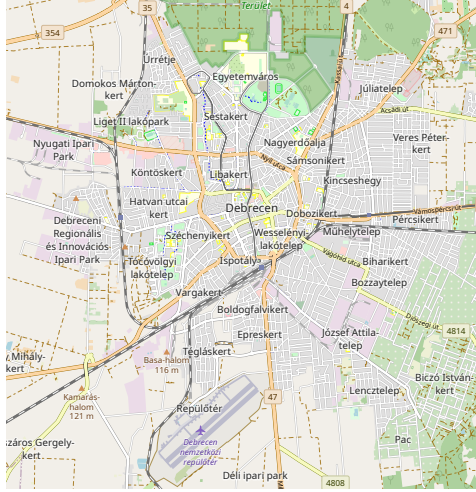


Figure 5. The map of the observed area. The graph created from the OSM data has 14,465 nodes, 29,770 edges, and covers a total of 799.4 km of road. The size of the area is about 106 km².

© OpenStreetMap contributors.

For a visual explanation of the transition probability vector, see Fig. 7. We are at the graph vertex (or intersection) of OSM node ID 26755459 (with GPS coordinates 47.5417164, 21.6097831). From this node, we can move towards nodes 1402222987, 1402222861, 1534652124, and 7834632455. The transition to each node has a certain probability, see Table 2.

We generated trajectories using Algorithm 1. For this, we created a Q for Debrecen, but since we have no real-world traffic data, we generated random values for the 2D stationary distribution. To compare our results, we generated trajectories using the same algorithm for Porto, Portugal. In case of Porto, we could calculate a Q that is approximated based on real-world data, namely, the Taxi Trajectory Prediction dataset, following the methods described in paper [4]. One can easily see on Fig. 8 that the trajectories generated based on a real Q have more realistic shapes (in case of Porto, see the left subfigure in Fig. 8), while the others are quite random (in case of Debrecen, see the right subfigure in Fig. 8b). An interesting question arises: can we tell if a Q reflects the real traffic system of a city? We assume that a Q can be validated with trajectories generated from it. If these trajectories reflect the real traffic in a certain level, we can accept Q . Elaborating

this validation technique is one of our future work.

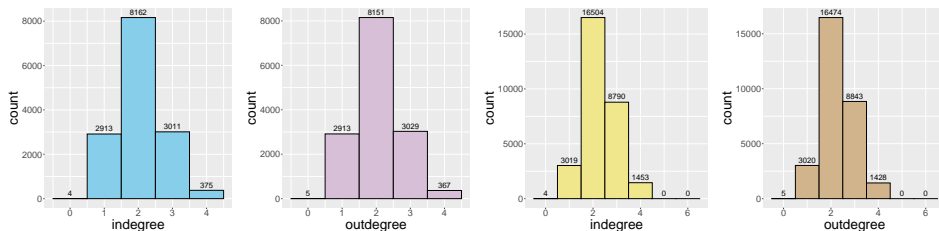


Figure 6. The degree distribution (first: in-vertices, second: out-vertices, third: in-edges, fourth: out-edges) histograms of the Debrecen map road graph.

Table 2. Transitions of intersection 26755459.

Neighbor node	Transition Probability
1402222987	0.24
1402222861	0.32
1534652124	0.26
7834632455	0.18
Sum	1

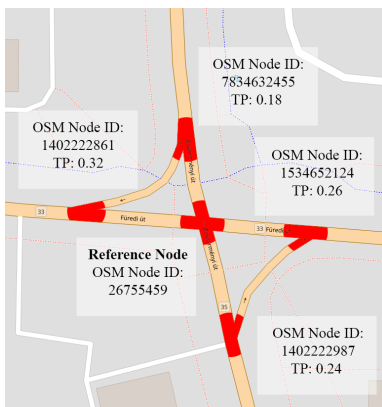


Figure 7. A visual explanation of transitions of intersection 26755459. TP means transition probability, nodes are highlighted with red. Base map and data from OpenStreetMap and OpenStreetMap Foundation. © OpenStreetMap contributors. Annotated by the authors.

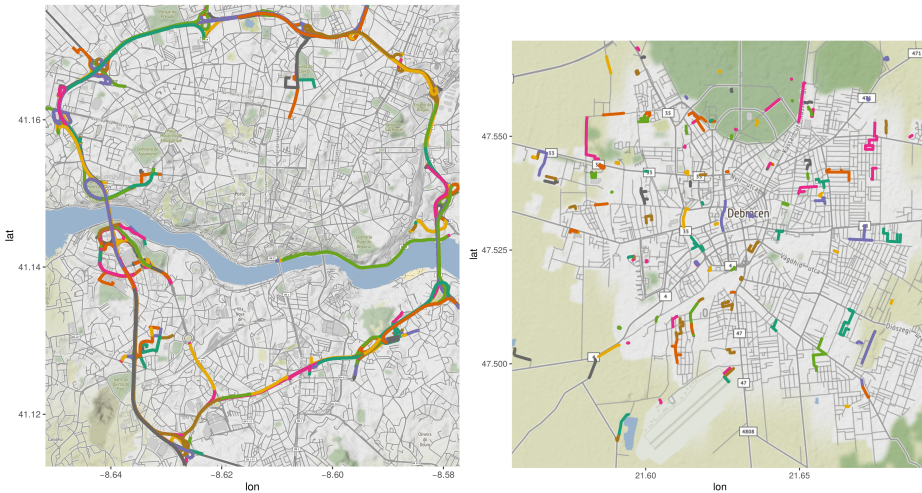


Figure 8. Generated trajectories in Porto (left: $n = 200,000,000$; $m = 75$) and Debrecen (right: $n = 50,000,000$; $m = 35$) simulated with Algorithm 1.

7. Conclusions

In this paper we have described various graph models for proper road networks and introduced the concept of Markov traffic. By tools of Markov chain theory, we have proven the existence and uniqueness of a stationary distribution for any Markov traffic on strongly connected and aperiodic road networks. We have also derived an explicit formula for the stationary distribution and the two-dimensional stationary distribution. Finally, we have proposed a simulation algorithm for generating random trajectories which follows the two-dimensional stationary distribution which being closest to a given mask matrix on the road network.

To test our theories, we have implemented the proposed model in our simulation program (RCE) using OpenStreetMap. The whole project (including RCE) is available for download.⁶

Future work will focus on the further improvements and the possible applications of our simulation algorithms, e.g., modelling the pollution or energy consumption in Smart Cities.

Acknowledgements. The authors would like to thank all actual and former members of the Smart City group of the University of Debrecen. We are especially grateful to all of the participants of the OOCWC competitions and the students of the BSc courses of “High Level Programming Languages” at the University of Debrecen.

⁶<https://github.com/rbesenczi/Crowd-sourced-Traffic-Simulator/blob/master/justine/install.txt>

References

- [1] J. BANG-JENSEN, G. Z. GUTIN: *Digraphs: Theory, Algorithms and Applications*, Berlin Heidelberg New York: Springer Science & Business Media, 2008, DOI: <https://doi.org/10.1007/978-1-84800-998-1>.
- [2] N. BÁTFAI, R. BESENCZI, A. MAMENYÁK, M. ISPÁNY: *OOVCW: The Robocar World Championship initiative*, in: *Telecommunications (ConTEL)*, IEEE 13th International Conference on, ed. by T. PLANCK, 2015, pp. 1–6, DOI: <https://doi.org/10.1109/ConTEL.2015.7231223>.
- [3] N. BÁTFAI, R. BESENCZI, A. MAMENYÁK, M. ISPÁNY: *Traffic simulation based on the Robocar World Championship initiative*, *Infocommunications Journal* 7.3 (2015), pp. 50–58.
- [4] R. BESENCZI, N. BÁTFAI, P. JESZENSZKY, R. MAJOR, F. MONORI, M. ISPÁNY: *Large-scale Simulation of Traffic Flow using Markov Model*, *PLoS ONE* 16.2 (2021), DOI: <https://doi.org/10.1371/journal.pone.0246062>.
- [5] P. BOCQUIER: *World Urbanization Prospects: An alternative to the UN model of projection compatible with the mobility transition theory*, *Demographic Research* 12 (2005), pp. 197–236, DOI: <https://dx.doi.org/10.4054/DemRes.2005.12.9>.
- [6] P. BRÉMAUD: *Markov chains. Gibbs fields, Monte Carlo simulation, and queues*, vol. 31, *Texts in Applied Mathematics*, New York: Springer-Verlag, 1999, DOI: <https://doi.org/10.1007/978-1-4757-3124-8>.
- [7] M. CAVERS, K. VASUDEVAN: *Spatio-temporal complex Markov Chain (SCMC) model using directed graphs: Earthquake sequencing*, *Pure and Applied Geophysics* 172.2 (2015), pp. 225–241, DOI: <https://doi.org/10.1007/s00024-014-0850-7>.
- [8] E. CRISOSTOMI, S. KIRKLAND, R. SHORTEN: *A Google-like model of road network dynamics and its application to regulation and control*, *International Journal of Control* 84.3 (2011), pp. 633–651, DOI: <https://doi.org/10.1080/00207179.2011.568005>.
- [9] M. CROVELLA, E. KOLACZYK: *Graph wavelets for spatial traffic analysis*, in: *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, vol. 3, 2003, pp. 1848–1857, DOI: <https://doi.org/10.1109/INFCOM.2003.1209207>.
- [10] C. DABROWSKI, F. HUNT: *Using Markov chain and graph theory concepts to analyze behavior in complex distributed systems*, tech. rep., U.S. National Institute of Standards and Technology, 2011.
- [11] M. FAIZRAHNEMOON: *Real-data modelling of transportation networks*, PhD thesis, Hamilton Institute, National University of Ireland Maynooth, 2016.
- [12] M. FAIZRAHNEMOON, A. SCHLOTE, E. CRISOSTOMI, R. SHORTEN: *A Google-like model for public transport*, in: *International Conference on Connected Vehicles and Expo (ICCVE)*, 2013, pp. 612–613, DOI: <https://doi.org/10.1109/ICCVE.2013.6799864>.
- [13] R. A. HORN, C. R. JOHNSON: *Matrix Analysis*, Cambridge: Cambridge University Press, 2012.
- [14] A. HORN, K. NAGEL, K. W. AXHAUSEN: *The Multi-Agent Transport Simulation MATSim*, Ubiquity Press London, 2016, DOI: <https://doi.org/10.5334/baw>.
- [15] E. ISMAGILOVA, L. HUGHES, Y. K. DWIVEDI, K. R. RAMAN: *Smart cities: Advances in research—An information systems perspective*, *International Journal of Information Management* 47 (2019), pp. 88–100, DOI: <https://doi.org/10.1016/j.ijinfomgt.2019.01.004>.

- [16] J. P. JARVIS, D. R. SHIER: *Graph-theoretic analysis of finite Markov chains*, in: Applied mathematical modeling: A multidisciplinary approach, ed. by D. R. SHIER, K. T. WALLENIUS, CRC Press, 1996, pp. 85–102.
- [17] N. L. JOHNSON, S. KOTZ, N. BALAKRISHNAN: *Discrete Multivariate Distributions*, New York: Wiley, 1997.
- [18] D. KRAJZEWICZ, J. ERDMANN, M. BEHRISCH, L. BIEKER: *Recent Development and Applications of SUMO - Simulation of Urban MObility*, International Journal On Advances in Systems and Measurements, International Journal On Advances in Systems and Measurements 5.3&4 (Dec. 2012), pp. 128–138, URL: <http://elib.dlr.de/80483/>.
- [19] A. LESNE: *Complex Networks: from Graph Theory to Biology*, Letters in Mathematical Physics 78 (2006), pp. 235–262, DOI: <https://doi.org/10.1007/s11005-006-0123-1>.
- [20] H. MENOVAR, I. GUVENC, K. AKKAYA, A. S. ULUAGAC, A. KADRI, A. TUNCER: *UAV-Enabled Intelligent Transportation Systems for the Smart City: Applications and Challenges*, IEEE Communications Magazine 55.3 (2017), pp. 22–28, DOI: <https://doi.org/10.1109/MCOM.2017.1600238CM>.
- [21] K. NAGEL, M. SCHRECKENBERG: *A cellular automaton model for freeway traffic*, Journal de Physique I France 2.12 (1992), pp. 2221–2229.
- [22] A. M. NAGY, V. SIMON: *Survey on traffic prediction in smart cities*, Pervasive and Mobile Computing 50 (2018), pp. 148–163, DOI: <https://doi.org/10.1016/j.pmcj.2018.07.004>.
- [23] E. NECULA: *Dynamic traffic flow prediction based on GPS data*, in: IEEE 26th International Conference on Tools with Artificial Intelligence, 2014, pp. 922–929, DOI: <https://doi.org/10.1109/ICTAI.2014.140>.
- [24] L. E. OLMOS, S. ÇOLAK, S. SHAFIEI, M. SABERI, M. C. GONZÁLEZ: *Macroscopic dynamics and the collapse of urban traffic*, Proceedings of the National Academy of Sciences 115.50 (2018), pp. 12654–12661, DOI: <https://doi.org/10.1073/pnas.1800474115>.
- [25] B. PAN, Y. ZHENG, D. WILKIE, C. SHAHABI: *Crowd sensing of traffic anomalies based on human mobility and social media*, in: Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2013, pp. 344–353, DOI: <https://doi.org/10.1145/2525314.2525343>.
- [26] S. PORTA, P. CRUCITTI, V. LATORA: *The Network Analysis of Urban Streets: A Dual Approach*, Physica A 369 (2006), pp. 853–866, DOI: <https://doi.org/10.1016/j.physa.2005.12.063>.
- [27] E. VLAHOGIANNI, M. KARLAFTIS, J. GOLIAS: *Short-term traffic forecasting: Where we are and where we are going*, Transportation Research Part C Emerging Technologies 43.1 (2014), DOI: <https://doi.org/10.1016/j.trc.2014.01.005>.
- [28] U. VON LUXBURG: *A tutorial on spectral clustering*, Statistics and Computing 17.4 (2007), pp. 395–416, DOI: <https://doi.org/10.1007/s11222-007-9033-z>.
- [29] Y. WU, X. ZHANG, Y. BIAN, Z. CAI, X. LIAN, X. LIAO, F. ZHAO: *Second-order random walk-based proximity measures in graph analysis: Formulations and algorithms*, The VLDB Journal 27.1 (2018), pp. 127–152, DOI: <https://doi.org/10.1007/s00778-017-0490-5>.
- [30] Y. XU, S. ÇOLAK, E. C. KARA, S. J. MOURA, M. C. GONZÁLEZ: *Planning for electric vehicle needs by coupling charging profiles with urban mobility*, Nature Energy 3.6 (2018), pp. 484–493, DOI: <https://doi.org/10.1038/s41560-018-0136-x>.

- [31] X. YIN, G. WU, J. WEI, Y. SHEN, H. QI, B. YIN: *A Comprehensive Survey on Traffic Prediction*, tech. rep., <https://arxiv.org/abs/2004.08555>, Arxiv, 2020.

Appendix

In order to demonstrate the results of this paper we present a simple toy example implemented in Python. Consider the road network $G = (V, E)$ on Fig. 1, where $V := \{1, 2, 3, 4, 5\}$ and $E := \{(1, 2), (2, 1), (2, 3), (3, 4), (4, 2), (4, 5), (5, 2)\}$. Then $|V| = 5$ and $|E| = 7$. The adjacency matrix A of G , where we denote the vertices as well, can be derived as:

$$A := \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 0 \\ 4 & 0 & 1 & 0 & 0 & 1 \\ 5 & 0 & 1 & 0 & 0 & 0 \end{array}.$$

Clearly, G is a strongly connected digraph. Since $1 \rightarrow 2 \rightarrow 1$ and $2 \rightarrow 3 \rightarrow 4 \rightarrow 2$ are cycles of length 2 and 3, respectively, we have $\text{per}(G) = 1$ and thus G is aperiodic. The first power k that $A^k > 0$ is $k = 6$ and

$$A^6 := \begin{bmatrix} 2 & 2 & 2 & 1 & 1 \\ 2 & 4 & 2 & 2 & 1 \\ 2 & 3 & 2 & 1 & 1 \\ 3 & 4 & 3 & 2 & 1 \\ 2 & 2 & 2 & 1 & 1 \end{bmatrix}.$$

The entries of this matrix are the number of directed walks of length 6 between the pairs of vertices. One can see that the in- and outdegree of vertices are given as $\mathbf{d}^- = (1, 3, 1, 1, 1)^\top$ and $\mathbf{d}^+ = (1, 2, 1, 2, 1)^\top$, respectively.

Define the Markov kernel P on the road network G as:

$$P := \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1/2 & 1/2 & 0 & 0 & 0 \\ 2 & 1/4 & 1/2 & 1/4 & 0 & 0 \\ 3 & 0 & 0 & 1/2 & 1/2 & 0 \\ 4 & 0 & 1/4 & 0 & 1/2 & 1/4 \\ 5 & 0 & 1/2 & 0 & 0 & 1/2 \end{array}.$$

Fig. 1 displays the Markov kernel P denoting the transition probabilities on the edges and its s.d. $\boldsymbol{\pi}$ denoting on the vertices. Note that $\boldsymbol{\pi} = 1/11(2, 4, 2, 2, 1)^\top$ and the 2D s.d. is given by:

$$Q = \frac{1}{22} \begin{bmatrix} 2 & 2 & 0 & 0 & 0 \\ 2 & 4 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 \\ 0 & 1 & 0 & 2 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

One can easily check that the marginals of Q coincide to $\boldsymbol{\pi}$.

We compute the 2D s.d. least square approximation of the adjacency matrix A by Theorem 5.1. The symmetric unnormalized graph Laplacian matrix L of the road network G is given as:

$$L = \begin{bmatrix} 2 & -2 & 0 & 0 & 0 \\ -2 & 5 & -1 & -1 & -1 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & -1 & -1 & 3 & -1 \\ 0 & -1 & 0 & -1 & 2 \end{bmatrix}.$$

The eigenvalues are $(0, 1.55, 2, 4, 6.45)$. The multiplicity of the smallest eigenvalue 0 is 1 which shows that the road network is strongly connected. The generalized (Moore-Penrose) inverse of L can be derived as

$$L^{-1} = \begin{bmatrix} 0.44 & 0.04 & -0.16 & -0.16 & -0.16 \\ 0.04 & 0.14 & -0.06 & -0.06 & -0.06 \\ -0.16 & -0.06 & 0.365 & -0.01 & -0.135 \\ -0.16 & -0.06 & -0.01 & 0.24 & -0.01 \\ -0.16 & -0.06 & -0.135 & -0.01 & 0.365 \end{bmatrix}.$$

Then, by solving the vector linear equation $L\boldsymbol{\lambda} = \mathbf{d}^+ - \mathbf{d}^-$, we have Lagrange multipliers $\boldsymbol{\lambda} = (-0.2, -0.2, 0.05, 0.3, 0.05)$. One can see that the sum of multipliers is 0. Thus, the 2D s.d. Q_A to the adjacency matrix A is

$$Q_A = \frac{1}{26} \begin{bmatrix} 0 & 4 & 0 & 0 & 0 \\ 4 & 0 & 5 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 \\ 0 & 2 & 0 & 0 & 3 \\ 0 & 3 & 0 & 0 & 0 \end{bmatrix}$$

with stationary marginals $\boldsymbol{\pi}_A = 1/26(4, 9, 5, 5, 3)^\top$. The error square of the approximation is $\text{SSD} = 0.5$.

Proof of formula (4.5). For all $\mathbf{g} \in \mathcal{F}_k$ we have by formulas (4.1) and (4.2) and the multinomial theorem that

$$\begin{aligned} \sum_{\mathbf{f} \in \mathcal{F}_k} \boldsymbol{\varrho}(\mathbf{f}) R(\mathbf{f}, \mathbf{g}) &= k! \sum_{\mathbf{f} \in \mathcal{F}_k} \prod_{u \in V} \pi_u^{f_u} \sum_{K \in \mathcal{M}(\mathbf{f}, \mathbf{g})} \prod_{u, v: u \Rightarrow v} \frac{p_{uv}^{k_{uv}}}{k_{uv}!} \\ &= k! \sum_{\mathbf{f} \in \mathcal{F}_N} \sum_{K \in \mathcal{M}(\mathbf{f}, \mathbf{g})} \prod_{u, v: u \Rightarrow v} \frac{(\pi_u p_{uv})^{k_{uv}}}{k_{uv}!} = k! \sum_{\sum_{u \in V} k_{uv} = g_v} \prod_{u, v: u \Rightarrow v} \frac{(\pi_u p_{uv})^{k_{uv}}}{k_{uv}!} \\ &= k! \prod_{v \in V} (g_v!)^{-1} \left(\sum_{u \in V} \pi_u p_{uv} \right)^{g_v} = k! \prod_{v \in V} \frac{\pi_v^{g_v}}{g_v!} = \boldsymbol{\varrho}(\mathbf{g}). \quad \square \end{aligned}$$

English language learning by visualizing the literary content of a knowledge base in the three-dimensional space*

István Károly Boda^a, Erzsébet Tóth^b

^aDebrecen Reformed Theological University,
Department of Mathematics and Informatics
boda.istvan@drhe.hu

^bUniversity of Debrecen, Faculty of Informatics
toth.erzsebet@inf.unideb.hu

Submitted: January 9, 2021

Accepted: April 7, 2021

Published online: May 18, 2021

Abstract

In our paper we would like to present and visualize the details of the data structure and organization of our three-dimensional virtual library model (3DVLM). The current implementation of the model is based on the features of the three-dimensional virtual space of the MaxWhere Seminar System [23] which provides carefully arranged smartboards in the 3D space for hypertext-based content.

Keywords: Three-dimensional virtual library model (3DVLM), MaxWhere Seminar System, Callimachus and the Library of Alexandria, text-based English language learning

AMS Subject Classification: 68U05, 68U35, 68T05, 91E10, 91E40

1. Introduction

The overall aim of the three-dimensional virtual library model (3DVLM) is to provide for the library users selected verbal and multimedia content in the virtual 3D

*This research was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was co-financed by the Hungarian Government and the European Social Fund.

space (and in parallel in the hypertextual 2D space). In general, the 3DVLM can be considered as a human-computer interaction-based application in the framework of CogInfoCom research [1, 2] focusing on the efficient organization and visualization of the presented content in order to help preserve and transmit classical cultural heritage and to support autonomous and text-based English language learning [15].

The 3DVLM is intended to achieve two basic goals:

- first, giving a general introduction and overview of the classical heritage which the European culture is based on. In this respect, the users can have access to selected texts about Callimachus who was one of the most productive and influential Hellenistic scholar-poets of his age. In addition, the library contains selected literary texts by Callimachus and other prominent authors (e.g. epigrams, lyric poems, anecdotes etc.). Because an average user of the library may have difficulties in understanding the provided literary texts, the library provides the readers with supporting materials (e.g. vocabulary items, concordances and quotes, selected parts of relevant Wikipedia entries, commentaries etc.) which help them to understand and interpret (and in turn enjoy) the preprocessed primary texts. Note that the co-reference and/or intertextual relationships are represented, in the first place, by hypertext links between the primary texts and the supporting materials;
- second, the virtual library is intended to support language learning by carefully preparing and commenting the provided texts in order that the users, and especially the young members of the generation CE with supposed intermediate or advanced English language skills can easily acquire the accumulated knowledge and preserved values that the ancient authors created centuries ago. Although the content and structure of the virtual library focus, in the first place, on the improvement of reading as one of the four basic language skills, the first goal of the 3DVLM outlined above naturally involves the development of the learners' cultural awareness.

The learning philosophy of the model is to help the readers understand and interpret the *primary texts* of the virtual library 'at once', supplying them with the necessary knowledge represented by *secondary materials* covering the relevant linguistic or dictionary, generic, encyclopedic and background knowledge. Because the efficient (i.e. easy, simple, self-evident, user-friendly etc.) access to the primary texts and the associated secondary materials is essential in the learning process, we tried to structure, arrange and visualize the compiled material by using different and varied colors and typography (e.g. font and paragraph styles, images, graphics, icons, lists, tables, graphs, spatial maps, etc.). Note that exploiting the excellent and spectacular features of the MaxWhere 3D environment might possibly motivate the members of the generation CE in itself, and therefore encouraging and urging them to enhance their knowledge (being either linguistic or cultural). In addition, the importance of English language skills is undoubtedly crucial in the Internet era for everyone who wants to access and benefit from the global information sources available. Therefore we firmly hope that our potential users will, either immediately

or eventually, make up their mind, and gradually improve their English competence in the course of reading, understanding and memorizing the preprocessed material provided by our virtual library.

We would like to provide a brief overview of similar research directions, there are several other CogInfoCom researches related to educational opportunities in virtual reality, and especially in the MaxWhere Seminar System.

- VR, in general, provides efficient virtual workspace for education even in industrial or company-based environment (e.g. dual education, VR simulations for special training etc. [3, 22]). As “a desktop virtual environment for education, learning and working” [5], the MaxWhere Seminar System seems to be especially suitable to develop educational content in the virtual 3D space [18], and to provide personalized, customizable learning environment and paths [19].
- The MaxWhere system opens up a wide range of excellent and spectacular opportunities when we want to choose the most appropriate 3D space for a special application. There are several existing 3D spaces available, and new ones are being published from time to time. Due to its inherent flexibility, the system enables the development of new 3D environments where various objects, places, buildings, rooms etc. can be constructed or reconstructed, along with the plausible and striking arrangement of different smartboards [17, 24].
- According to in-depth studies, there are more benefits of the MaxWhere Seminar System. It is a substantial benefit of the 3D virtual workspace that it can display some extra information permanently for its users in designated smartboards [4]. In addition to the immersive experience that a well-designed VR environment can offer, the different navigation methods of the MaxWhere Seminar System provide a heightened sense of spatial presence for its users [5, 6].
- The elaborated 3D spaces of the MaxWhere system can be metaphorically considered as a “memory palace” [16] where the conceptual mapping of the various concepts of the presented content to specific parts or components of the virtual 3D environment, representing a unique spatial arrangement and/or temporal ordering of 3D objects [16], can greatly help the learners memorize the concepts, their insights, and in turn the whole material to be learned (by memorizing individual properties of the 3D objects with or without following a pre-determined scenario). Moreover, it was demonstrated by published experiments that when using MaxWhere 3D spaces for educational purposes its users have remarkably more effective memory recall, better comprehension and faster activity than using 2D web-based content [21].

All those benefits can be extensively exploited when we would like to design and create an interactive learning material in the 3D space. Therefore the emerging

VR applications for educational purposes would have true potential, and they are supposed to achieve fast progress in the near future. If this is so, we can, by all means, consider “virtually augmented, interactive education” [2] in the 3D space as one of the major research areas of CogInfoCom.

In our paper we would like to focus on the data structure and organization of the current implementation of the 3DVLM. For those, who are interested in the three-dimensional virtual library model, we recommend our previous publications [7–15] where further details can be found. Note that the 2D version of the library is available in the internet [20].

2. Structure patterns of the model

The content, organization and presentation of the virtual library model are equally important for supporting efficient English language learning. In our paper *we would like to focus on the data structure and organization of the virtual library model* selecting, introducing and illustrating typical *structure patterns* by visualizing them. Each pattern follows the binary relation form $\rho(x,y)$ where the domain (X) and codomain (Y) of the relation depend on the type of the pattern introduced. For example, in the simplest pattern both X and Y are sets containing primary texts (about Callimachus etc.) represented by slides and denoted by S01, S02, etc. That is,

$$X \equiv \{S01, S02, \dots\} \quad \text{and} \quad Y \equiv \{S01, S02, \dots\},$$

where the element ‘x’ is associated to the element ‘y’ if and only if there is a hypertext link from the slide ‘x’ to the slide ‘y’. As we shall see later, the actual representation of the link can differ according to the context (e.g. a link can be represented by a keyword or “key phrase” within the slide, a link symbol, a recommended link at the bottom of the slide etc.).

In the next subsections, we would like to introduce the most important structure patterns of the 3DVLM. We developed a kind of “navigation map” for the main items of the virtual library which can serve as a practical tool for visualizing the main patterns.

The selected items of the virtual library and the most typical relationships between them are illustrated on the interactive *navigation page* of the virtual library [20]. Here each item is identified by a simple code (e.g. the first slide about Callimachus is identified by S01, the second slide by S02 etc.) which are associated by a hypertext link to the exact location (e.g. an URL#anchor) where the referred item can be found. Therefore the page actually represents a kind of “navigation map” for the main items of the virtual library (see Fig. 1).

Because each structure pattern corresponds to a given type of relationship between certain items of the library, the navigation page is a convenient way of presenting a quick overview of the main patterns.

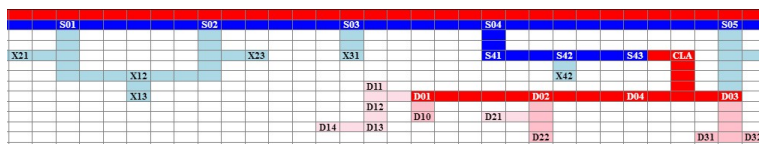


Figure 2. A selected part of the navigation map including primary texts.

2.2. Links between primary and/or secondary texts [ELINK]

This structure pattern represents links between primary texts and/or secondary texts (both presented in separate web pages or slides). The main function of the secondary texts is to explain certain concepts which might be unknown or unclear to an average user (that is, those texts represent encyclopedic knowledge). In the current implementation of the 3DVLM, there are slides about Cyrene ([X12](#)), Alexandria ([X23](#)), the Ptolemaic Dynasty ([X21](#)) etc. The hypertext link is represented by a special symbol (showing two parallel arrows pointing to different directions, \rightleftarrows) placed immediately after the keyword or key phrase which is associated to the linked text. In the slide about Callimachus ([S01](#)) the link symbol can be seen after *Cyrene* (pointing to the corresponding slide [X12](#)), the *Library of Alexandria* (pointing to the corresponding slide [X23](#)) etc. (see Fig. 3).

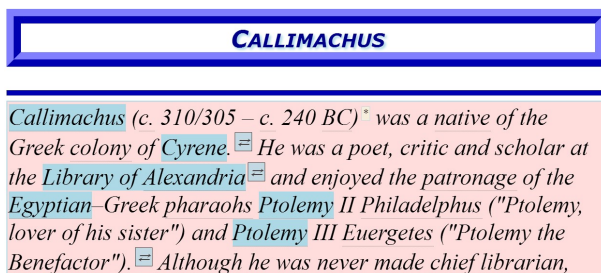


Figure 3. A selected part of the slide about Callimachus ([S01](#)) illustrating VLINK (...), XLINK (*) and ELINK (\rightleftarrows).

2.3. Links between slides and vocabulary items [VLINK]

This structure pattern represents links between different kind of texts (e.g. primary texts, secondary texts, thesaurus pages etc. presented in separate slides) and vocabulary items. The main function of vocabulary items is to explain certain words (rare words, proper nouns, abbreviations etc.) which might be unknown or unclear to an average user (that is, those items represent dictionary knowledge in the usual form of dictionary entries including pronunciation, different forms, explanation etc.). The hypertext link to a vocabulary item is represented by a dotted line

(...) under the keyword or key phrase which is associated to the linked vocabulary entry. In the slide about Callimachus in Fig. 3, the dotted link symbol can be seen under *Callimachus, c., BC, native, colony, Cyrene, Alexandria, patronage* etc. The vocabulary item which the uncountable noun “patronage” is associated to can be seen in Fig. 4. Note that the links to the vocabulary items, because of their quantity, are not presented in the navigation page.

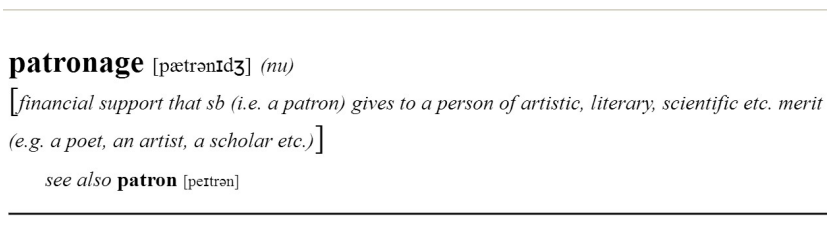


Figure 4. The vocabulary item about “patronage”.

2.4. Links between keywords (key phrases) and various notes [XLINK and CLINK]

This structure pattern (being either XLINK or CLINK) represents links between keywords (or key phrases) and notes which might reveal some additional information (e.g. definitions, explanations, commentaries, context, pronunciation, meaning, synonyms and antonyms, sentence samples, selected concordances etc.) about the corresponding keyword or key phrase. The main difference between XLINK and CLINK is, on the one hand, the place of the note within the page where the keyword or key phrase can be found. On the other hand, in XLINK the notes have mainly explanatory function, whereas in CLINK the notes are usually background materials (revealing possible contexts etc.).

In both XLINK and CLINK, the hypertext link to the note is represented by an asterisk (*) which is placed immediately after the keyword or key phrase which is associated to the corresponding note.

In the first structure pattern type (XLINK), notes are presented in the form of numbered endnotes, and placed usually in the same slide where the associated keywords can be found (although links to the notes from other slides are quite possible). In the slide about Callimachus in Fig. 3, there is an asterisk after the key phrase (c. 310/305 – c. 240 BC) which is explained as “Callimachus was born around 305 BC and died around 240 BC” in the endnote denoted by ^[1] (see Fig. 5).

Note that although both the explanatory notes which XLINKS point to (e.g. ^[1] *Callimachus was born...*) and RLINKS which point to specific reference items (e.g. ... *Hellenistic age*. ^[1] and ... *the Muses*. ^[2]) are numbered in superscript position which might lead to some confusion. To avoid this, *both the way we mark the numbers and their position are completely different*: in case of explanatory

notes, we use square brackets before and after the numbers, and the numbers are placed at the end of page; in case of RLINKS, however, we use border-boxes (i.e. four border edges) around the numbers and they are placed right after the content which represents a specific reference (to a given bibliographic item or source in another page).

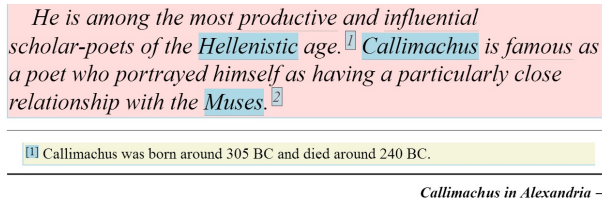


Figure 5. The note no. [1] for (c. 310/305 – c. 240 BC) illustrating XLINK the source of which can be seen in Fig. 3, represented by an asterisk (*). There is a link at the bottom of the slide to another slide (entitled “Callimachus in Alexandria”) illustrating DLINK.

In the second structure pattern type (CLINK), the additional information which a selected keyword or key phrase is associated to is placed immediately under the sentence or paragraph containing the keyword or key phrase. For example, in the thesaurus page (T02) the key phrase “ESL class” (emphasized by colored borders and having an asterisk as a link symbol) is associated to a short text (extracted from a web page) providing a broader context of the sentence which contains the key phrase (“In an ESL class, you would be assisting the main **TEACHER** who plans the lessons and is primarily **responsible for** the class.”, see Fig. 6).

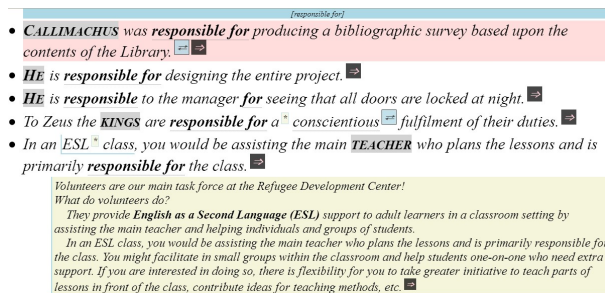


Figure 6. A selected part of the thesaurus page (T02) containing concordances about “responsible for”. The figure illustrates VLINK (...), CLINK (\rightleftharpoons and *), and RLINK (\Rightarrow).

The structure and function of thesaurus pages will be explained later in subsection 2.5.

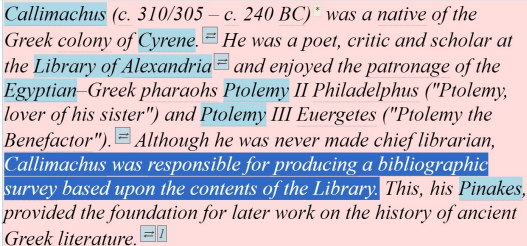
Where a concordance is selected from one of the primary or secondary texts it is emphasized by colored background. In such cases, the source text itself provides

the broader context of the concordance, so it can serve as a kind of “note” (in the sense we used before) for the concordance. Although this case can also be considered as a CLINK structure pattern type, the hypertext link to the source text is represented by the same symbol that we are using in the ELINK pattern type (showing two parallel arrows pointing to different directions). For example, the first concordance in Fig. 6 (“*CALLIMACHUS was responsible for producing a bibliographic survey based upon the contents of the Library.*”) can be found in the slide about Callimachus (S01) and therefore the link, presented by the sign ⇔ placed immediately after the concordance, is pointing to this slide.

2.5. Links between thesaurus pages [TLINK]

This structure pattern represents links between vocabulary items and thesaurus pages. While vocabulary entries represent dictionary knowledge, thesaurus pages represent chiefly generic knowledge (which, in our case, expressed by semantic relationships between certain words or phrases). Thesaurus pages contain a selected bunch of concordances and quotations, all of which fit a given microcontext, that is, one or two collocation patterns built of semantically related words or phrases upon a specific subject or meaning. For example, the thesaurus page (T02) is organized around words or phrases concerning “people with respect to responsibility, motivation and commitment”, and the concordances or quotations in the page fit either the collocation pattern [*adj+noun*] or [*noun+BE+adj/pp*] (e.g. enthusiastic person; determined woman; resolute leader; etc; He is eager, willing, and compliant to ...; Callimachus was determined not to ...; the students are ... highly motivated; Callimachus was responsible for ...; etc.). Note that we can have direct access to all thesaurus pages through the navigation map (see Fig. 1) where the corresponding codes of the thesaurus pages (T01, T02, ..., T05, T06, ... etc.) can be found in the middle of the page around the code CPT (which is the abbreviation of “Collocation PaTterns”, and identifies a separate web page containing all collocation patterns used in the virtual library).

As an example, let us select the first slide (S01) about Callimachus, and seek for the key sentence “*CALLIMACHUS was responsible for producing a bibliographic survey based upon the contents of the Library.*” (W02) (see Fig. 7).



Callimachus (c. 310/305 – c. 240 BC) was a native of the Greek colony of Cyrene. He was a poet, critic and scholar at the Library of Alexandria and enjoyed the patronage of the Egyptian-Greek pharaohs Ptolemy II Philadelphus ("Ptolemy, lover of his sister") and Ptolemy III Euergetes ("Ptolemy the Benefactor"). Although he was never made chief librarian, Callimachus was responsible for producing a bibliographic survey based upon the contents of the Library. This, his Pinakes, provided the foundation for later work on the history of ancient Greek literature.

Figure 7. Another part of the slide about Callimachus (S01) showing the selected key sentence (W02).

Clicking the dotted phrase “responsible for” in the page, we can go to the corresponding vocabulary page (see Fig. 8) where we can find two links to the relevant microcontexts in the corresponding thesaurus page (T02) (see Fig. 6). As we can see, the hypertext link to a thesaurus page is represented by an expression containing the key phrase “*responsible for*” (marked by colored background, and introduced by the symbol \Rightarrow) which is associated to the linked thesaurus page.

Note that the navigation page (see Fig. 1) shows explicitly the code of the key sentence (W02), and in turn illustrates the indirect relationship (that is, the above-mentioned two-step relationship through the corresponding vocabulary page) between the first slide (S01) and the thesaurus page (T02).

responsible [rɪˈspɒnsəbl] (*for sb/sth*)

[*having control and authority over sth/sb and the duty of taking care of it/them (Cambridge 1995)*]

see also **responsibility** [rɪˌspɒnsəˈbɪləti]

microcontexts:

\Rightarrow *responsible for producing sth*

\Rightarrow *responsible for a conscientious fulfilment of sth*

Figure 8. The vocabulary item about “responsible for” illustrating TLINK.

2.6. Links between various content units and reference items [RLINK]

This structure pattern represents links between various content units from the library (that is, sentences, concordances, quotations, paragraphs, texts etc. from primary texts, secondary texts, thesaurus pages and notes) and reference items which describe the original source of the content units (printed and electronic books, journals, newspapers, Wikipedia and dictionary entries, various web pages and electronic materials available in the internet etc.). The hypertext link to a reference item is represented by either a reference number in superscript position (e.g.¹) in slides, or a double arrow in superscript position (\Rightarrow) in thesaurus pages. The link symbols are placed immediately after the corresponding content unit. For example, in the slide about Callimachus (S01) in Fig. 5 there are two references: the first is a Wikipedia entry (57th item), and the second is a chapter from a book (219th item) (see Fig. 9).

Another example is the concordance “*In an ESL class, you would be assisting the main **TEACHER** who plans the lessons and is primarily **responsible for** the class.*” in the thesaurus page T02 in Fig. 6. Here the referred item is a web page (363rd item) (see Fig. 10).

The reference items, that is, the bibliographic descriptions of all sources having been referred to from the content of the virtual library, are available in a separate

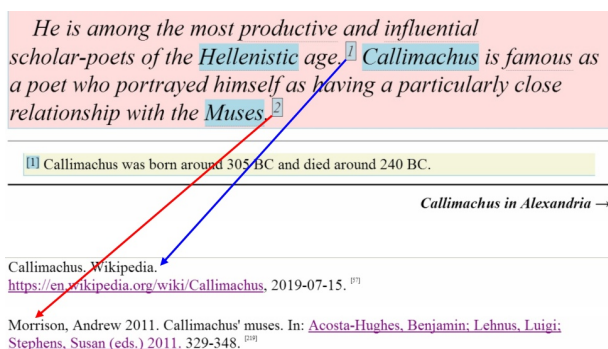


Figure 9. References from the slide about Callimachus (S01) to the 57th and 219th reference items illustrating RLINK.

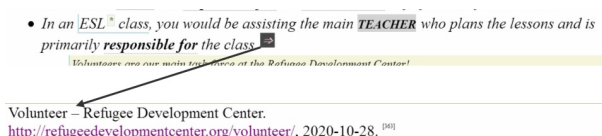


Figure 10. Reference from the thesaurus page (T02) to the 363rd reference item illustrating RLINK.

web page REF. In the current implementation of the 3DVLM there are more than 370 reference items. Although the code of the reference page does appear in the navigation map (in the top right corner), the individual links to the reference items are not presented there.

TIME	EVENT
c. 1260/1180 BC	the Trojan War ^{Wiki}
c. 1100 BC	beginning of the Greek Dark Ages ^{Wiki} (Homeric Age or Geometric Period)
c. 1102/850 BC	birth of Homer ^{(supposed) Wiki} (cf. c. 750/700 BC)
c. 850/750 BC	composition of 'The Iliad' and 'The Odyssey' ^{(supposed) Wiki}
	end of the Greek Dark Ages ^{Wiki}

Trojan War. Wikipedia.
https://en.wikipedia.org/wiki/Trojan_War, 2019-07-15. [97]

Figure 11. Reference from the top of the timeline page (TIM) to the 357th reference item illustrating RLINK.

Although we have had a comprehensive overview of the most important structure patterns of the virtual library model, there are some additional features. For example there is a timeline page (TIM) of the most important events (“histori-

cal milestones”), an index page of selected keywords (**KWI**), and a category page which describes the main categories of the hierarchical classification scheme (**CLA**) of the ancient Library of Alexandria. Note that this famous scheme, referred to as “Pinakes”, was invented by Callimachus in the 3rd century BC. In the current implementation of the 3DVLM each content unit of those pages has a hypertext link to a selected item in the reference page (e.g. to a relevant Wikipedia entry) where additional information can be found about the content (see Fig. 11).

3. Final remarks

Although the current implementation of the model in the MaxWhere Seminar System uses the 3D Castle space, the model, because of its flexible organization, is fully compatible with other MaxWhere 3D spaces. In the 3D Castle space the content of the virtual library is presented in carefully arranged smartboards (see Fig. 12 and Fig. 13). With a special care we have developed the navigation map which presents the various kinds of relationships between the content units of the virtual library and thus making the retrieval of these units more effective for the users.

In our paper we provided a brief overview of how the content of the virtual library is organized. We emphasized and visualized those features of the model which might be useful for the possible users of the virtual library, focusing especially on the needs of English language learners who, we firmly hope, are going to have strong interest in and motivation about the ancient culture and heritage.

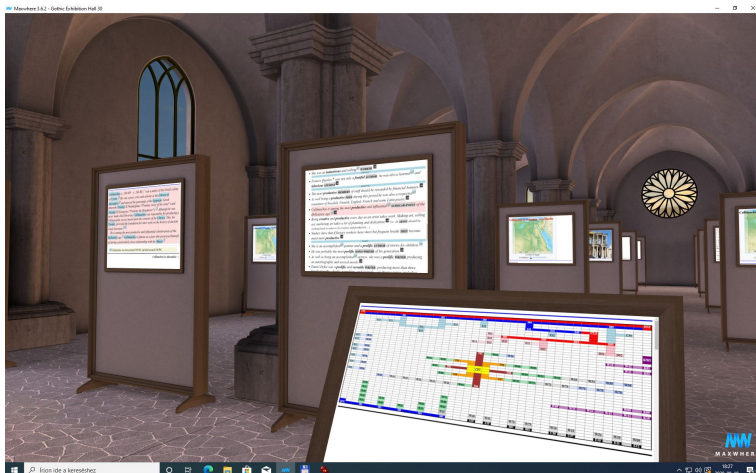


Figure 12. The content of the virtual library in the 3D Castle space of the MaxWhere Seminar System (1).

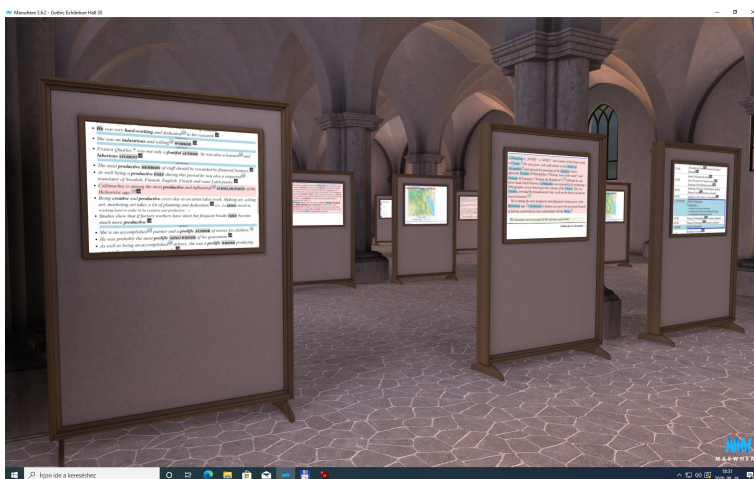


Figure 13. The content of the virtual library in the 3D Castle space of the MaxWhere Seminar System (2).

Acknowledgements. The results presented in the paper have partially been achieved in the Virtual Reality Laboratory of the Faculty of Informatics of the University of Debrecen, Hungary.

References

- [1] P. BARANYI, Á. CSAPÓ: *Definition and synergies of Cognitive Infocommunications*, Acta Polytechnica Hungarica 9.1 (2012), pp. 67–83.
- [2] P. BARANYI, Á. CSAPÓ, G. SALLAI: *Cognitive Infocommunications (CogInfoCom)*, Berlin, Heidelberg: Springer, 2015, DOI: <https://doi.org/10.1007/978-3-319-19608-4>.
- [3] F. BELLALOUNA: *Virtual-Reality-based Approach for Cognitive Design-Review and FMEA in the Industrial and Manufacturing Engineering*, in: CogInfoCom 2019. Proceedings of the 10th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2019, pp. 41–46, DOI: <https://doi.org/10.1109/CogInfoCom47531.2019.9090002>.
- [4] B. BERKI: *Desktop VR as a Virtual Workspace: a Cognitive Aspect*, Acta Polytechnica Hungarica 16.2 (2019), pp. 219–231.
- [5] B. BERKI: *Navigation Power of MaxWhere: a Unique Solution*, in: CogInfoCom 2020. Proceedings of the 11th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2020, pp. 511–515, DOI: <https://doi.org/10.1109/CogInfoCom50765.2020.9237904>.
- [6] B. BERKI: *Sense of Presence in MaxWhere Virtual Reality*, in: CogInfoCom 2019. Proceedings of the 10th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2019, pp. 91–94, DOI: <https://doi.org/10.1109/CogInfoCom47531.2019.9089976>.

- [7] I. BODA, M. BÉNYEI, E. TÓTH: *New dimensions of an ancient Library: the Library of Alexandria*, in: CogInfoCom2013. Proceedings of the 4th IEEE International Conference on Cognitive Infocommunications, New York, NY, USA: IEEE, 2013, pp. 537–542, DOI: <https://doi.org/10.1109/CogInfoCom.2013.6719306>.
- [8] I. BODA, E. TÓTH: *From Callimachus to the Wikipedia: an ancient method for the representation of knowledge in the WWW era*, in: CogInfoCom2018. Proceedings of the 9th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2018, pp. 205–210, DOI: <https://doi.org/10.1109/CogInfoCom.2018.8639895>.
- [9] I. BODA, E. TÓTH, M. BÉNYEI, I. CSONT: *A three-dimensional virtual library model of the ancient Library of Alexandria*, in: ICAI 2014. Proceedings of the 9th International Conference on Applied Informatics, Eger, Hungary: Eszterházy Károly Teacher Training College, 2014, vol. 1, 103–111, DOI: <https://doi.org/10.14794/ICAI.9.2014.1.103>.
- [10] I. BODA, E. TÓTH, I. CSONT, L. T. NAGY: *Developing a knowledge base of ancient literary texts in virtual space*, in: CogInfoCom2016. Proceedings of the 7th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2016, pp. 263–270, DOI: <https://doi.org/10.1109/CogInfoCom.2016.7804559>.
- [11] I. BODA, E. TÓTH, I. CSONT, L. T. NAGY: *The use of mythological content in virtual learning environment*, in: CogInfoCom2017. Proceedings of the 8th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2017, pp. 307–314, DOI: <https://doi.org/10.1109/CogInfoCom.2017.8268262>.
- [12] I. BODA, E. TÓTH, I. CSONT, L. T. NAGY: *Toward a knowledge base of literary content focusing on the ancient Library of Alexandria in the three dimensional space*, in: CogInfoCom2015. Proceedings of the 6th IEEE International Conference on Cognitive Infocommunications, New York, NY, USA: IEEE, 2015, pp. 251–258, DOI: <https://doi.org/10.1109/CogInfoCom.2015.7390600>.
- [13] I. BODA, E. TÓTH, F. Z. ISZÁLY: *Text-based approach to second language learning in the virtual space focusing on Callimachus' life and works*, in: CogInfoCom 2019. Proceedings of the 10th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2019, pp. 439–444, DOI: <https://doi.org/10.1109/CogInfoCom47531.2019.9089933>.
- [14] I. K. BODA, E. TÓTH: *Classical Heritage and Text-Based Second Language Learning in Three-Dimensional Virtual Library Environment*, in: ICAI 2020. Proceedings of the 11th International Conference on Applied Informatics, Aachen, Germany: CEUR-WS, Vol. 2650., 2020, pp. 46–56.
- [15] I. K. BODA, E. TÓTH: *English language learning in virtual 3D space by visualizing the library content of ancient texts*, in: CogInfoCom2020. Proceedings of the 11th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2020, pp. 305–311, DOI: <https://doi.org/10.1109/CogInfoCom50765.2020.9237887>.
- [16] Á. B. CSAPÓ, I. HORVÁTH, P. GALAMBOS, P. BARANYI: *VR as a Medium of Communication: from Memory Palaces to Comprehensive Memory Management*, in: CogInfoCom 2018. Proceedings of the 9th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2018, pp. 389–394, DOI: <https://doi.org/10.1109/CogInfoCom.2018.8639896>.
- [17] A. GILÁNYI, A. RÁCZ, A. M. BÓLYA, J. DÉCSEI, K. CHMIELEWSKA: *A Presentation Room in the Virtual Building of the First National Theater of Hungary*, in: CogInfoCom 2020. Proceedings of the 11th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2020, pp. 519–523, DOI: <https://doi.org/10.1109/CogInfoCom50765.2020.9237828>.

- [18] I. HORVÁTH: *How to Develop Excellent Educational Content for 3D VR*, in: CogInfoCom 2019. Proceedings of the 10th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2019, pp. 483–489, DOI: <https://doi.org/10.1109/CogInfoCom47531.2019.9089916>.
- [19] I. HORVÁTH: *Personalized Learning Opportunity in 3D VR*, in: CogInfoCom 2020. Proceedings of the 11th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2020, pp. 425–439, DOI: <https://doi.org/10.1109/CogInfoCom50765.2020.9237895>.
- [20] *Interactive map of the three dimensional virtual library (3DVLM)*, URL: <https://bodaistvan.hu/callimachus/map.html> (visited on 01/05/2021).
- [21] D. KISS, P. BARANYI: *3D Webspace VS 2D Website*, in: CogInfoCom 2020. Proceedings of the 11th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2020, pp. 517–518, DOI: <https://doi.org/10.1109/CogInfoCom50765.2020.9237898>.
- [22] Z. KVASZNICZA, J. KOVÁCS, G. MAZA, B. PÉLI: *VR-based Dual Education at E.ON. The Win-win-win Situation for Companies, Graduates and Universities*, in: CogInfoCom 2019. Proceedings of the 10th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2019, pp. 479–482, DOI: <https://doi.org/10.1109/CogInfoCom47531.2019.9089961>.
- [23] *MaxWhere VR Even more*, URL: <http://www.maxwhere.com/> (visited on 01/05/2021).
- [24] A. RÁCZ, A. GILÁNYI, A. M. BÓLYA, J. DÉCSEI, K. CHMIELEWSKA: *On a Model of the First National Theater of Hungary in MaxWhere*, in: CogInfoCom 2020. Proceedings of the 11th IEEE International Conference on Cognitive Infocommunications, Piscataway, NJ, USA: IEEE, 2020, pp. 575–576, DOI: <https://doi.org/10.1109/CogInfoCom50765.2020.9237848>.

A barycentric coordinates-based visualization framework for movement of microscopic organisms*

Andrea Bodonyi^{ab}, Győző Kurucz^c, Gábor Holló^c,
Roland Kunkli^b

^aUniversity of Debrecen, Doctoral School of Informatics

^bUniversity of Debrecen, Faculty of Informatics
{bodonyi.andrea,kunkli.roland}@inf.unideb.hu

^cUniversity of Debrecen, Faculty of Humanities
kurucz.gyozo@arts.unideb.hu
hollogabor@gmail.com

Submitted: December 22, 2020

Accepted: April 19, 2021

Published online: May 18, 2021

Abstract

As many research projects face the need to manage a large amount of generated research-specific data, usually with a specific structure, the demand for visualization systems is common. Likewise, the emerging data volume could turn substantially transparent cast in a visual appearance. Also, the non-trivial character of the dataset made the construction of a custom visualization necessary. Taking the possessed requirements into account, we designed a tool for processing the simulation data and handling the issue resulted from the indirectness with a previously analyzed barycentric conversion method. The system also visualizes the microscopic organism's behavior and supports a straightforward data analysis through several built-in tools.

Keywords: Visualization framework, animation, movement, microscopic organism, barycentric coordinates

AMS Subject Classification: 68U05

*This work was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund.

1. Introduction

Dealing with a massive volume of research-specific data is a common problem in the case of a broad spectrum of research projects. It may be about searching for anomalies, patterns, or even visual forms of data; it usually leads to a visualization system's demand. Inspecting the microscopic world is a perfect example as the users need to extract new knowledge regarding the ecosystem's nature [4].

As our research aimed to explore various aspects of the microscopic organisms' behavior in a well-defined environment [2, 3], we encountered the problem mentioned above. The research-specific data volume could, with high probability, provide for us some higher-level properties presented visually, being difficult to observe otherwise. Thanks to the non-trivial characteristics of our output datasets, no out-of-the-box solution could be put into practice, which led us to design a visualization system satisfying the condition set.

This work's main purpose is to demonstrate our visualization framework's creation process and operation to visualize microscopic organisms' movement and behavior. The framework first needs to deal with the input dataset's indirectness and obtain the environmental elements' per-frame locations. We realized this by building on a method using barycentric coordinates to convert from local to global coordinates.

The object locating problem derives from the fact that the simulation dataset includes a frame sequence describing an organism's step-by-step movement indirectly, by the environment's behavior. Each frame consists of the environmental change from the previous step described from the organism's perspective. Considering this information, we had the demand for locating every element in the global world.

First, we would like to present the starting problem in Section 2 by describing the environmental elements and the data structure. Next, we give a brief overview of the method applied to the organism localization in its world in Section 3. We also demonstrate the implementation details and its features here. We continue with the system's evaluation throughout several test cases and present their accuracy profile in Section 4 and close our paper with a conclusion in Section 5.

2. Problem definition

2.1. Environment description

The project aims to visualize a schematic, microscopic, free-moving organism that locomotes in a 3D watery environment. In this size range and environment, viscous forces dominate over inertial forces; thus, locomotion occurs in the realm of the so-called low Reynolds numbers (e.g., [6]), where locomotion is essentially a sort of "creeping" through the water. The environment also contains stimuli relevant for the animal modeled by point-like light sources representing food.

The scene to be visualized involves several elements. In the center of the picture,

there is a moving organism having in our case a microscopic size. The organism is built up of its main body and several sensors with fixed fields of view. Different simulations may vary in the number, the field of view, and the positioning of the sensors on the organism-body.

The organism senses one particular food if it falls into the field of view of at least one sensor and its intensity is high enough to be relevant. The animal moves around the space, possibly perceiving some of the foods, even consuming them. The organism “eats” a piece of food if its locomotion trajectory intersects the given food point, which is much smaller than the animal. Thus, to eat food, the animal has to have a precisely oriented locomotion.

Figure 1 presents the main elements in the organism simulation. The main object located in origin represents the organism, which has a γ angle of view. Points F_i ($i = 1, 2, \dots, 5$) mark the food positions in the environment.

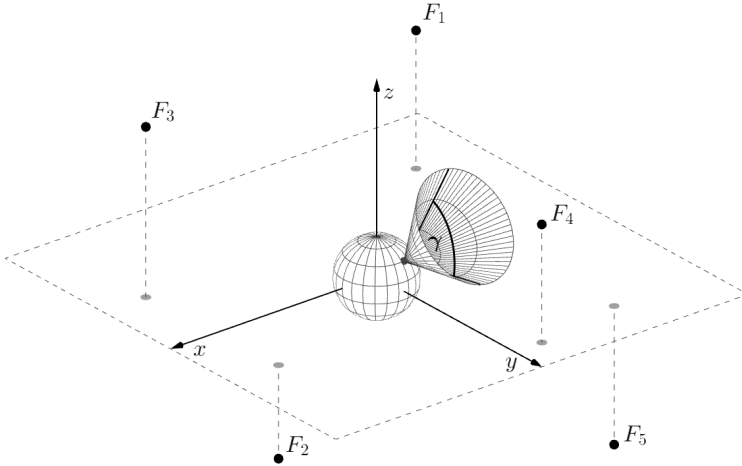


Figure 1. Schematic overview of the main elements involved in the simulation of the organism.

The foods’ fixed lifetimes are supposed to imitate food consumption by other competitors in the space leading to the fact that some foods can expire. A food may disappear after the consumption by either our organism or one of the competitors (the food’s lifetime has expired).

2.2. Data structure

The simulation system’s only purpose is to simulate the organism’s behavior in its environment and its attitude to the foods. As it is only concerned with the simulation, the system does not provide any visual feedback. In each of the cases, the system’s output is a single dataset describing the step-by-step movement stored in a JSON format.

As Figure 2 shows, the data file contains the organism's structural description. It has a fixed size, several sensors, and an angle of view. The description also includes the spherical coordinates for the sensors, which indicate their position on the organism-body in its local coordinate system. In this example, the organism has four sensors distributed evenly with an 85-degree angle of view.

```

"structure" : {
  "size" : 0.05,
  "sensors" : [
    { "theta" : 0.523599, "phi" : 0.0 },
    { "theta" : 0.523599, "phi" : 1.5708 },
    { "theta" : 0.523599, "phi" : 3.14159 },
    { "theta" : 0.523599, "phi" : 4.71239 }
  ],
  "angleofview" : 85
}

```

Figure 2. Example output of the simulation framework, describing the structure of the organism.

Step-by-step movement data follow the structural description of the organism. The frame sequence consisting of the steps defines the organism's movement, each frame having the same structure. As Figure 3 presents, a frame object contains several attributes, such as the frame number, rotation vector, translation vector, and the food collection. A food object includes its coordinates relative to the organism, an intensity value, and a reset index, which indicates the potential food repositioning.

```

"frames" : [
  { "frame_num" : 0,
    "rv" : [0, 0, 0],
    "tv" : [0, 0, 0],
    "foods" : [
      { "x" : -1.08496, "y" : 0.838925,
        "z" : 1.13508, "i" : 0.587424,
        "reset" : 1,
      }, ...
    ]
  }, ...
]

```

Figure 3. Example simulation output, describing the per-frame state of the simulated environment.

3. Visualization framework

3.1. Method for determining animal location

As the data sequence lacked the current organism positions, we needed to extract these location vectors from the indirect description of the movement relative to the organism. To this end, we implemented a previously presented mathematical method [1] for the conversion from local to the global coordinate system.

As presented in this prior work, solving the problem using the matrix-based approach leads to a higher overall floating-point error propagation, which is a direct result of relying on earlier frames. The strength of the barycentric method proposed in the previous work lies in the capability of avoiding such precision issues and in the ability to locate the object in the global system entirely without any information from earlier frames.

To apply the barycentric method to the problem at hand, we used the food positions (defined in the organism's local coordinate system) as the required reference positions. Such an approach was made possible by our prior knowledge about the simulated environment; in every input frame we have five food points, which satisfies the constraints for obtaining the barycentric coordinates of a three-dimensional point.

With the thus-constructed reference frame available, the proposed method consists of two main steps. The purpose of the first step is to determine the barycentric coordinates of the central object in its local coordinate system. To this end, we relied on the fact that the input positions were relative to the organism; the direct consequence of this property is that the main object's location is always $(0, 0, 0)$. Building on [5, p. 46] (with several simplifications) as a starting point, and using the mentioned information, we could determine the barycentric coordinates for use in the next step.

The second step of the barycentric method consists of applying the previously determined barycentric coordinates, which is the moving organism's actual global position in the current step. This step requires a static basis in the global coordinate system. Assuming that the organism starts its path in the global origin, we own the information that the origins of the global and the organism's local coordinate systems are the same in the first frame, which also means that the positions in the whole scene are in the global coordinate system in the starting moment. With this knowledge in our hands, we can form the required static basis by taking the first frame's food positions. Thus, we can obtain the moving organism's global position by converting the obtained barycentric coordinates back to the global system concerning the formed basis.

3.2. Food regeneration problem

One issue with our previously described usage of the barycentric approach is that the reference basis's food positions are not always stationary. As we described earlier, the reference foods can arbitrarily disappear and reappear in other places,

making it necessary to update the static basis after every changing food locations.

Every food has a randomly generated, fixed lifetime in our simulation, representing its other competitors' consumption. As a result, a food can disappear either because the moving organism consumed it or because its lifetime expired, meaning that another animal consumed it.

There is always a replacement for the missing food with a new one at a new location. The newly appeared food point's location is also in the organism's local coordinate system, like every other one. When this situation occurs, we have to form a new reference basis with potentially new objects. To facilitate this, we can only rely on food positions for which we can obtain the global coordinates, but since we might not necessarily have required information about their world space positions (since they might have just appeared), we have to obtain their world space positions the same way as we calculate the position of the moving object. As it can be seen in Figure 4, the object locating problem always starts with an examination for food regeneration. The conversion is possible since every frame is guaranteed to contain more than four surrounding points, so if one disappears, we use the remaining ones as the reference points for the barycentric coordinates.

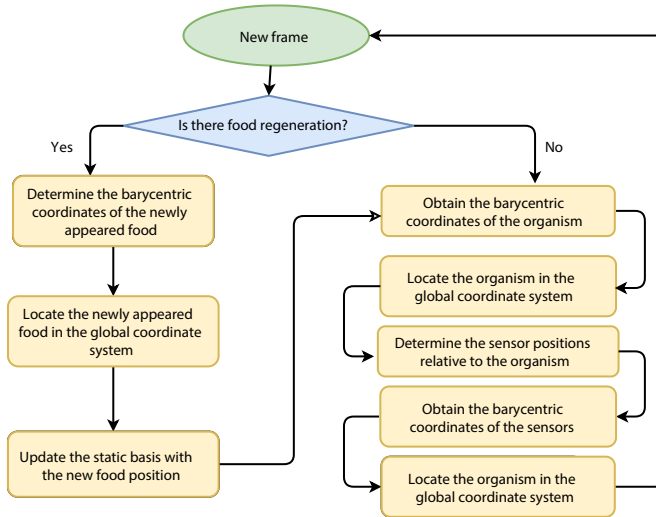


Figure 4. Building blocks of the localization method extended by the regeneration problem.

In the first step, we calculate the barycentric coordinates for the new reference objects based on the remaining four points of the frame, then use these to compute new global coordinates by weighting the world space positions of said remaining points the same way we did for the main object, as described in Section 3.1. In the last step, the newly calculated positions have to be stored as a new reference basis to use them later for the barycentric calculations.

3.3. Implementation

Considering all the requirements described in Section 2 and properties of the barycentric method, we constructed a tool to process the datasets resulting from the organism simulation, to determine the necessary global positions and visualize the results in a three-dimensional scene. Figure 4 presents the entire process for locating the object (including the ability to handle regenerated food points).

Throughout the framework’s implementation, we considered it essential to provide a system capable of delivering an interactive visualization environment, which serves as an efficient means of exploring the input data with user-driven mechanisms.

We implemented the visualization system using OpenGL in C++. Figure 5 shows an example output of the framework with the main elements. As shown in this example, the simulation objects are represented by spheres in our framework, mimicking the model described in Section 2.1. The object colors serve as a way for the efficient and unequivocal identification of the particular food objects and the sensors located on the organism-body.

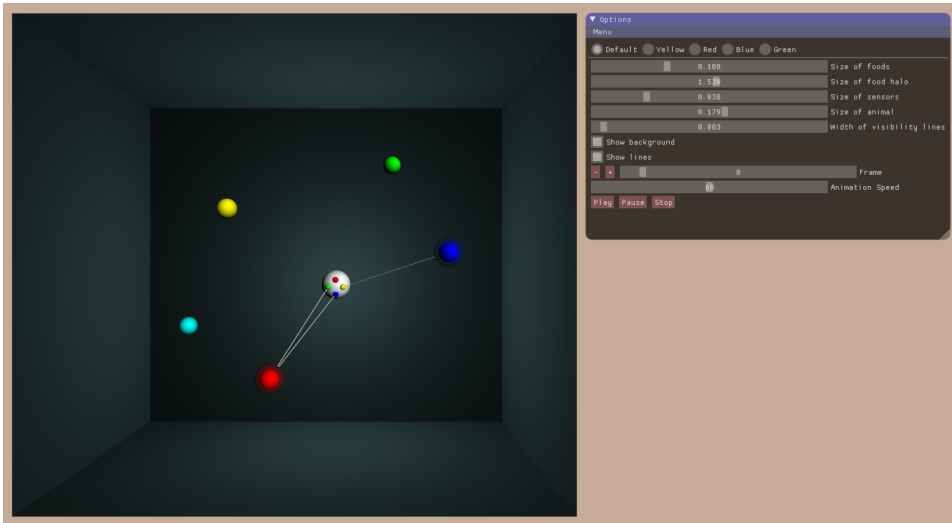


Figure 5. Example output of the visualization framework. In this scenario, the central, white sphere representing the moving organism perceives the red food with two sensors and the blue food with one sensor. The translucent haloes around the sensed foods and the links to the appropriate sensor visualize the perceptions.

3.4. Features

While designing the framework, we implemented several functionalities that provide the necessary tools for the users for an easy and transparent data analysis,

which offers an accessible way to answer their questions. First, we made the navigation in the frame sequence to any desired moment, both in the forward and backward direction. Similarly, there is a possibility of freezing the simulation at any time to move around the scene so that the user can examine the organism's momentary behavior from multiple viewpoints. Furthermore, the camera can be repositioned arbitrarily into any existing sensor, narrowing down the viewpoint only to the region that the selected sensor perceives. The user can also manage specific visualization settings through the graphical user interface, such as object sizes, line widths, and object visibilities, which eases their ability to focus on any simulation aspect.

From the perspective of visual analytics, the marking of the food visibilities was essential. As already mentioned, the organism sensed one particular food if it falls into its field of view and has an intensity high enough to be relevant. First, foods perceived by at least one sensor are marked by a translucent halo with an adjustable size. Furthermore, we link the foods sensed with a line to the sensor(s) perceiving them, so it is easy to analyze which food the organism chooses for consumption and which direction the organism selects for movement. A bounding box surrounds the environment representing the maximum possible extent of the foods and animal positions from the loaded dataset.

Figure 6 shows two different moments from the same dataset visualization with a zoomed-in viewpoint. The halos and the links can easily determine the food visibilities. On the left side, three different sensors sense all of the three perceptible foods. By comparison, the organism perceives only two of the foods on the right side, but it senses the red one with all of the four sensors.

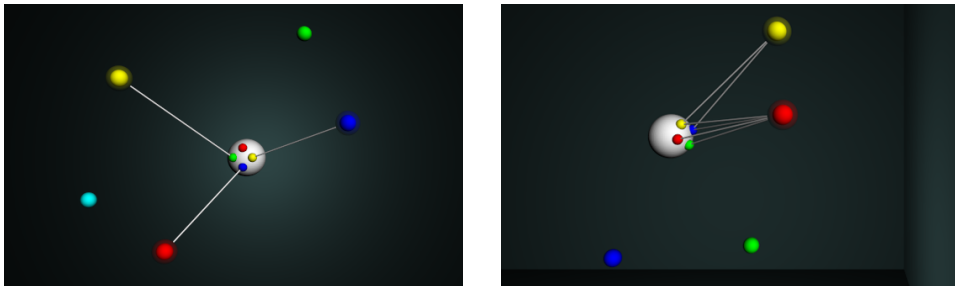


Figure 6. Two scenarios from the visualization framework with a zoomed in point of view with different food perceptions.

4. Results

To obtain our visualization system's accuracy profile, we analyzed its precision characteristics through several different test scenarios. Since the datasets resulting from the simulation do not provide the true location information necessary for such

a comparison, we first had to produce a suitable basis for the precision measurements. To this end, we generated multiple different artificial animal movements, which established a baseline for testing the accuracy of the proposed barycentric approach.

Figure 7 shows the movement paths in three of the aforementioned, custom generated test scenarios. Thanks to the absence of the organism’s physiological properties and the surrounding environment’s essential attributes, these object movements tend to be simpler and rule-following than the real-life organism behavior. Black lines visualize the generated movement paths, and the orange points mark the reference food positions.

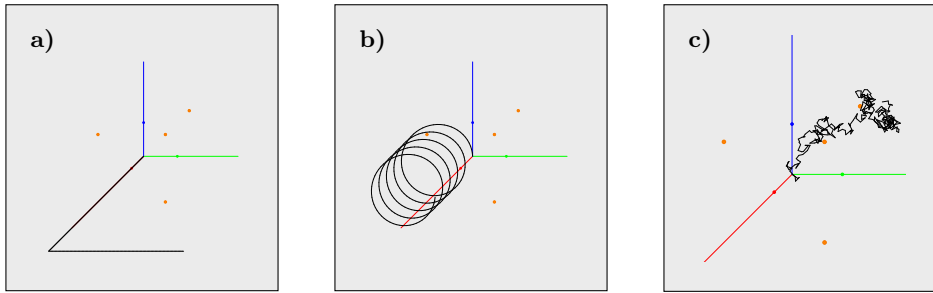


Figure 7. Artificially generated movement paths. **a)** Periodic movement with a few direction changes **b)** Spiral movement **c)** Fully random movement with random rotations.

As the previous test cases provided us a ground-truth data for the resulting organism locations to be compared to, we concluded the precision measurements of the system in Figure 8 and Figure 9. The red line marks the location difference for the spiral movement, the green for the periodic movement with a few direction changes, while the blue for the random movement. Figure 8 presents these cases without any food replacement, while Figure 9 presents the precision in the same cases, but with a food regeneration in every 200 frames. The magnitude of the object positions extends to approximately 14–15 units. In Figure 8 and Figure 9 can be seen that the precision difference is about three to five magnitudes lower in each of the cases.

The analysis of the resulting error metrics showed that the reconstruction error and the average distance of the reference points from the animal correlate in each frame. This is best illustrated by the oscillation of the error in the case of the spiral movement with regenerating foods, where the reference points tend to get clustered after a number of food regenerations, which leads to a constantly alternating increase and decrease of the average animal-food distances as the animal traverses the spiral path.

Finally, we evaluated our visualization system on real-life datasets, which described a specific organism’s movement in their well-defined environment. We show a few of such datasets in Figure 10. Specific cases differed in the attributes of the

organism and the movement lengths. The tested datasets' organisms contained varying numbers of sensors in a potentially different alignment in each input scenario. The dataset owners validated the output of the system in these cases.

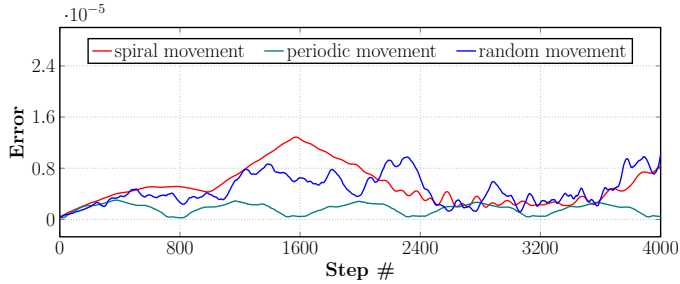


Figure 8. Precision measurements for the artificially generated movement paths shown in Figure 7 without food regeneration.

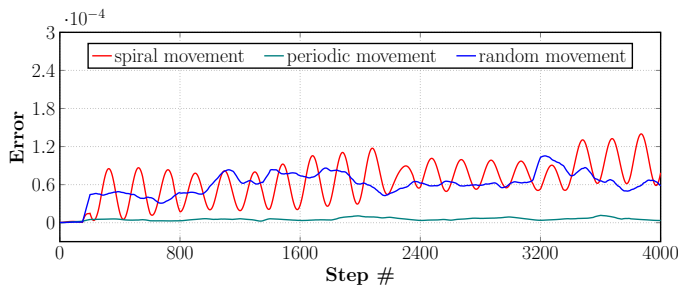


Figure 9. Precision measurements for the artificially generated movement paths shown in Figure 7 with a food regeneration in every 200 frames.

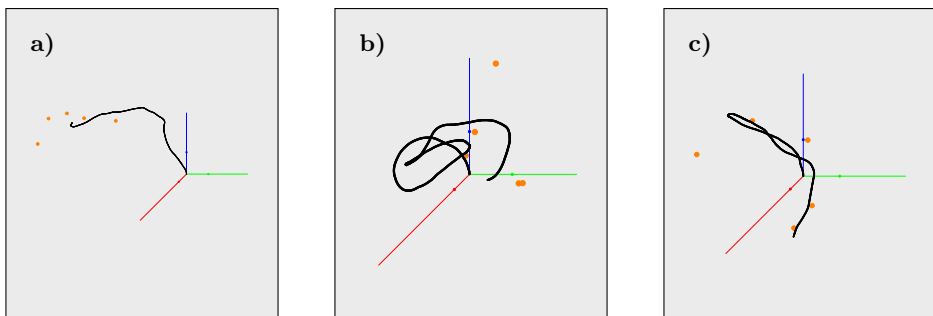


Figure 10. Movement paths generated using real-life datasets. The three paths are traversed by organisms with different sensor positions.

5. Conclusion

In this paper we presented our framework's creation process and operation, aiming to visualize microscopic organisms' movement and behavior in a well-defined environment. Considering all the requirements demonstrated in this work and putting the proposed barycentric method in [1] into practice, we created a research-specific data processing tool, dealing with the speciality arising from the type of data and delivering the results' visualization in a three-dimensional scene. It facilitates efficient analysis of the input data through real-time, user-driven mechanisms such as simulation freezing, scene inspection from multiple viewpoints, using analytical tools, and managing visualization-specific settings.

We first demonstrated the problem leading to our system's demand by a world description and presenting the special data structure. Then we gave a brief overview of the method building on barycentric coordinates for the conversion process that originates from the need to go from local to global coordinates. Building on the barycentric method, we could solve the problem given by the research-specific datasets' non-trivial characteristics. Next, we presented the implementation details and our framework's operation throughout the description of unique features making a transparent and straightforward data analysis possible.

As soon as the system's design process was finished, we installed it to the main research project regarding the properties of microscopic organisms' behavior. During its operation, our tool provided for us a wide range of higher-level knowledge through transforming our simulation data into a visual form, which would otherwise has been left unnoticed by the tool. We could exploit a great benefit in the moments when a portion of food was just being consumed in front of our organism. The recognition of which direction the organism chose to replan its way for the nutrition was a cardinal question, and along with several issues, they would not have been observable without any visual representation.

References

- [1] A. BODONYI, R. KUNKLI: *Efficient object location determination and error analysis based on barycentric coordinates*, Visual Computing for Industry, Biomedicine, and Art 3 (2020), 18:1–7, DOI: <https://doi.org/10.1186/s42492-020-00052-y>.
- [2] G. HOLLÓ: *A new paradigm for animal symmetry*, Interface Focus 5.6 (2015), 20150032:1–10, DOI: <https://doi.org/10.1098/rsfs.2015.0032>.
- [3] G. HOLLÓ, M. NOVÁK: *The manoeuvrability hypothesis to explain the maintenance of bilateral symmetry in animal evolution*, Biology Direct 7.1 (2012), 22:1–7, DOI: <https://doi.org/10.1186/1745-6150-7-22>.
- [4] T. ISHIKAWA: *Suspension biomechanics of swimming microbes*, Journal of The Royal Society Interface 6.39 (2009), pp. 815–834, DOI: <https://doi.org/10.1098/rsif.2009.0223>.
- [5] S. MARSCHNER, P. SHIRLEY: *Fundamentals of Computer Graphics*, 4th ed., A K Peters/CRC Press, 2015, ISBN: 9781482229394.

- [6] S. VOGEL: *Comparative Biomechanics: Life's Physical World*, 2nd ed., Princeton University Press, 2013, ISBN: 9780691155661.

Replacing the SIR epidemic model with a neural network and training it further to increase prediction accuracy^{*†}

Gergő Bogacsovics, András Hajdu, Róbert Lakatos,
Marcell Beregi-Kovács, Attila Tiba, Henrietta Tomán

University of Debrecen
bogacsovics.gergo@inf.unideb.hu
hajdu.andras@inf.unideb.hu
robert.lakatos@it.unideb.hu
beregikovacs.marcell@science.unideb.hu
tiba.attila@inf.unideb.hu
toman.henrietta@inf.unideb.hu

Submitted: December 21, 2020

Accepted: February 17, 2021

Published online: May 18, 2021

Abstract

Researchers often use theoretical models which provide a relatively simple, yet concise and effective way of modelling various phenomena. However, it is a well-known fact that the more complex the model, the more complex the mathematical description is. For this reason, theoretical models generally avoid large complexity and aim for the simplest possible definition, which although makes models mathematically more manageable, in practice it also often leads to sub-optimal performance. Furthermore, the data collected during the observations usually contain confounding factors, for which a simple theoretical model can not be prepared. Overall, mathematical models are usually too rigid and sophisticated, and therefore cannot really deal with

*The research was partly supported by the *1st Cloud Funding for Research Open Call of Open Clouds for Research Environments (OCRE)*, by the *ÚNKP-20-4-I New National Excellence Program of the Ministry for Innovation and Technology from the Source of the National Research, Development and Innovation Fund* and by the project *EFOP-3.6.2-16-2017-00015* supported by the European Union, co-financed by the European Social Fund.

†Equal contribution.

sudden changes in the environment. The application of artificial intelligence, however, provides a good opportunity to develop complex models that can combine the basic capabilities of the theoretical models with the ability to learn more complex relationships. It has been shown [16] that with neural networks, we can build such models that can approximate mathematical functions. Trained artificial neural networks are thus able to behave like theoretical models developed for different fields, while still retaining their overall flexibility, which guarantees an overall better performance in a complex real-world environment.

The aim of our study is to show our notion that we can create an architecture using neural networks, which is able to approximate a given theoretical model, and then further improve it with the help of real data to suit the real world and its various aspects better. In order to validate the functionality of the architecture developed by us, we have selected a simple theoretical model, namely the Kermack-McKendrick one [4] as the base of our research. This is an SIR [2] model, which is a relatively simple compartmental epidemic model, based on differential equations that can be used well for infections that spread very similarly to influenza or COVID. However, on one hand, the SIR model relies too heavily on its parameters, with slight changes in them leading to drastic overall changes of the S, I and R curves, and on the other hand, the simplicity of the SIR model distorts its accuracy in many cases. In our paper, by using the SIR model, we will show that the architecture described above can be a valid approach to modeling the spread of a given disease (such as influenza or COVID-19). To this end, we detail the accuracy of our models with different settings and configurations and show that it performs better than both a simple mathematical model and a plain neural network with randomly initialized weights.

Keywords: Deep learning, neural networks, mathematical models, approximation, parameter optimization, SIR model

1. Introduction

The appearance of COVID-19 made 2020 a very memorable year. It affected nearly all of the countries in the world, leading to quarantines and regulations that the people of the 21st century have never experienced before, while also delivering a heavy blow to the global economy, thus resulting in people losing their jobs. Not only that, the number of people dying due to this disease has also reached heights that was previously unimaginable. Therefore, it goes without saying that examining and researching this disease is of utmost importance to better understand its mechanics and driving forces.

Pure mathematical models have been very frequently used for modeling the spread of diseases like influenza [9, 12], measles [17] and so on. This is mainly due to their simplicity and mathematical foundations, which guarantees a compact, yet concise, effective and simple-to-explain model. However, the accuracy of pure mathematical models, such as SIR [4], SIS [1], SEIR [7], SEIRS [5], is often sub-optimal, which is precisely due to this simplicity. The prediction curve of an SIR

model for example is a simple bell-shaped curve, which means that a single SIR model cannot perform well when examining diseases like COVID-19, that have multiple waves. One possible solution to that problem is to fit multiple SIR models to the data. However, that is not optimal either, because we need to manually group the data into segments first, and then fit the SIR models to these groups. Selecting these groups manually however is not a trivial task, as the grouping affects the fitting phase, meaning that a bad grouping results in worse fitting and therefore yields worse results. This can easily be solved by using neural networks, since they can generalize well and achieve state-of-the-art accuracy in just about every field of science. One problem with using pure neural network models is the lack of quality data. For example, even though almost a year has passed since the initial outbreak of COVID-19, there are only around 200-300 data points per country to learn from, since data was usually gathered on a daily basis. Another problem is that COVID data often have huge amounts of noise due to how the recording of the daily cases took place and other factors. These can however heavily affect both the training phase (since the network will be much likely have a higher bias) and the performance of the network (since it will not be able to generalize well).

The goal of our research is to show that mathematical models, which are used frequently in investigating disease spread, epidemiology and even COVID-19 [15], can be applied to the training of neural network models as a pretraining step, resulting in a more accurate model. In our paper we show that if we train a neural network first on an SIR model, then train it further on real data, this method outperforms not only the original mathematical model, but also a neural network trained only on real data. This is because this way the neural network not only has access to more data (SIR data and real data), but the initial training phase (approximating the SIR model) is mathematically well-defined and the data points are not noisy, unlike the real data. This way, the neural network will have a solid set of weights before being trained on real data, that are roughly equivalent to a mathematical SIR model, making the second part of the training much smoother and easier. We also show that a neural network trained in this way can easily overcome the biggest obstacle of SIR models described above, which is not being able to forecast multiple waves by making predictions regarding the second wave of COVID-19 by training the models only on the first wave and evaluated on the second wave.

In this paper we outline a simple architectural solution regarding the model by using a simple neural network as the base. The data that the models operate on contain the daily number of newly infected patients and all the models were trained on data constructed from the original data by using a sliding window approach. This means that the models receive the number of infected people in the last t days, and predict that for the next T days. In other words, inputs were of length t , and the outputs were of length T , where each element was the number of newly infected people on a given day. We have used different values for t and T , respectively to achieve even better performances. Since due to the nature of the model, there will be multiple predictions for a given day, we also outline an aggregated solution

that combines the outputs of a single model for a given day by taking the mean of the predicted values. It is important to note that while in this paper we focus on deterministic models, due to its simple nature our proposed architecture can be used for stochastic models as well without any further restrictions.

The rest of the paper is organized as follows. In section 2 we present the SIR model and talk about its parameters and their effects on the performance of the model. In section 3 we briefly describe how the data used in the research was gathered. In section 4 we present our proposed two-step architecture and how it can be applied in practice. In section 5 we show how this architecture can be used on a simple and easier Influenza dataset, then in section 6 we show how these models achieve better results than normal neural networks whose weights have been initialized randomly for COVID-19 data. Finally, in the last section we summarize the results of our research.

2. SIR model

SIR is a general virus spread model that can be interpreted easily. The model can be described with an ordinary differential equation and has only a few parameters. Furthermore, in terms of behaviour it is a non-linear system. It is primarily recommended to be used for viruses where infected individuals cannot develop long-lasting immunity after recovery. This theoretical approach with regard to the spread of viruses was first described by William Ogilvy Kermack and Anderson Gray McKendrick and became generally known as the Kermack – McKendrick theory. We used this model in our research because currently, for both influenza and COVID-19, the scientific consensus is that, based on the behavioral characteristics of the virus, individuals cannot get sustained immunity after recovering.

The classic SIR model can be considered as a system of the following differential equations:

$$\begin{aligned}\frac{dS}{dt} &= \frac{\beta IS}{N} \\ \frac{dI}{dt} &= \frac{\beta IS}{N} - \gamma I \\ \frac{dR}{dt} &= \gamma I\end{aligned}$$

with the following notation of the parameters:

- S : Number of susceptible individuals.
- I : Number of infectious individuals.
- R : Number of recovered individuals.
- N : Total population.
- t : A given moment in time.

- β : Potential exposure rate per capita. Namely, it describes how many additional individuals a particular infected can infect at a given point in time.
- γ : Rate of recovery, which is practically the recovery/death rate and $1/\gamma$ is the infectious period.

We only need to study the equations for two of the three variables S , I and R , because

$$\frac{dS}{dt} + \frac{dI}{dt} + \frac{dR}{dt} = 0,$$

which follows from the fact that

$$S(t) + I(t) + R(t) = N.$$

For the basic reproduction number, we have

$$R(0) = \frac{\beta}{\gamma}.$$

3. Countries and data

For our research, we used infection data regarding influenza and COVID-19 viruses. For influenza-related studies, we examined the data published by the World Health Organization (WHO) between 2018 and 2019. WHO makes the data FluNet¹ [14] available by year and every record in the dataset consists of weekly values.

For COVID-19, we were working on data for new cases published by the Johns Hopkins Coronavirus Research Center [3]. Johns Hopkins University actualizes the data every day and makes it available on the open humanitarian data sharing platform (HDX). On the provided HDX platform, the start date of public dataset² [11] is 22 January, 2020.

Furthermore, for both COVID-19 and influenza we tested our proposed architecture on data for several different countries, like Austria, China, Croatia, Germany, Hungary, Japan, Romania. The full list of countries considered can be seen in the corresponding sections of the paper (Section 5 and 6). Our experimental results first focus on Hungary by comparing the Hungarian data with the ones of its neighbors. Then we extend our study to developed, leading countries providing presumably the most accurate data.

In case of the SIR model, one of the most important parameters is the population of the studied area. The knowledge of the population is essential for the SIR model because one of the starting conditions for calculating the theoretical model is the number of infectious individuals. Population data for the countries were collected from a public (total population) database³ [13] which is currently maintained by the World Bank. In the analysis, we used the most recent and available population data for 2019 for both viruses.

¹https://www.who.int/influenza/gisrs_laboratory/flunet/en/

²<https://data.humdata.org/dataset/novel-coronavirus-2019-ncov-cases>

³https://data.worldbank.org/indicator/SP.POP.TOTL?name_desc=false

4. Two-step architecture

In this section, we describe the mechanism of our proposed two-step architecture. First, we detail how we used the SIR model and how we extracted the optimal γ and β parameters for the model. Then, we present how we applied this fitted SIR model for training a more robust neural network model.

4.1. Determining the optimal parameters

A properly parameterized SIR model already has an acceptable prediction capability by itself. So when we approximate a theoretical model with a neural network, primarily this capability of the theoretical model is needed to be learnt. Namely, we would like to keep all information in the machine-learned model that can be extracted from the theoretical model. Furthermore, with the neural network, we aim to further improve such parameterized SIR models that best describe the real data. Accordingly, the first step in creating our own models was to find the γ and β parameters that generate such theoretical models which fit the real data the best.

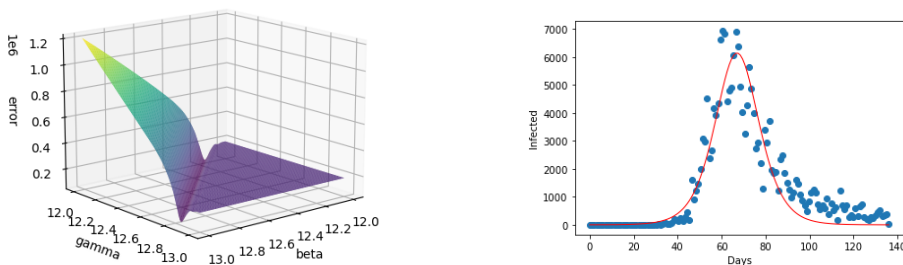


Figure 1. A slice from the big solution space (left) and a SIR curve fitted to the German data (right).

In the case of SIR, we have a non-linear least-squares minimization problem with a large solution space due to the peculiarities of the SIR model, as it can be seen on Figure 1. There are several methods for solving such problems, like gradient descent, Gauss-Newton algorithm, Levenberg-Marquardt algorithm [6, 8], or differential evolution [10]. In the experimental phase, we compared several solution methods (BFGS, Newton, brute force, etc.) out of which the Levenberg-Marquardt algorithm (LMA) and differential evolution algorithms (DEA) proved to be the most efficient ones. LMA is a fast and effective algorithm finding an ideal solution even if started from initial values relatively far from the optimum. Because of this property, it is applicable to determine proper starting parameters and their associated parameter range. DEA is proved to be slower than LMA, but starting with the same initial parameters, it generally finds a better solution than LMA. Because of its construction, DEA works more efficiently in the case of large solution spaces, like those occurring in optimization of SIR models.

Overall, our experience shows that both methods are well applicable to the curve fitting problem and can be used to find the optimal SIR model fitting best to real data (see Figure 1). Levenberg-Marquardt algorithm is especially advantageous because of its speed and the differential evolution algorithm is effective in further refining the existing near-optimal parameters.

4.2. Approximation

For approximating the spread of infectious disease, we apply a two-step architecture. First, we fit a SIR model to the given data (e.g. by searching for the optimal β and γ values). Then, we fit a neural network to the (infected) curve I of this SIR model, following the mechanism (concentrating only on curve I) of most of the methods applied in this field. This approach, of course, led to shorter training times and easier convergence. After the neural network obtained a set of weights that was roughly equivalent to the given SIR model (the predictions were close to the curve I of the SIR model), we decreased the learning rate and trained the neural network on real data. This step is applied to make sure that the weights of the neural network do not change substantially, which could have resulted in a model that no longer resembles the original SIR model. Another advantage of the decreased learning rate is that the impact of the noise present in real data can be reduced in this way.

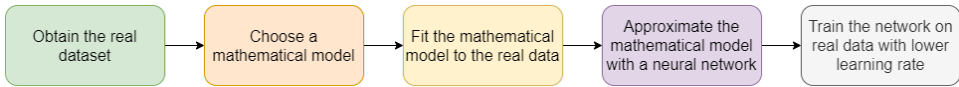


Figure 2. The basic workflow of the two-step architecture.

This approach (see Figure 2) has quite a few benefits compared to training a neural network directly on real data. First of all, the amount of noise generated throughout the training phase is considerably reduced, thanks to the model being taught on a much smoother and mathematically well-defined function, which is the output of the SIR model. The shape of the infected curve (I) for a given SIR model is bell-shaped, with no irregularities and noise, hence it is easier for neural networks to be trained on. Moreover, since the network has a solid set of weights after the first phase is finished and the learning rate is smaller during the second phase, the irregularities present in real data do not affect the training as much as they would normally. This fact, along with the ability of an SIR model to approximate the original data fairly well, leads to a more controlled training, during which the network can first extract meaningful information about the nature of the disease, and then it has access to a more irregular dataset for further training. Another huge advantage of this architecture is the increased amount of data available during the learning phase. It is beneficial when the available dataset contains only a handful of records (as in case of the Influenza dataset in our study). However, by pre-training on a mathematical model that behaves roughly the same, we can generate the necessary data points to obtain such a starting set of weights that only needs

to be further refined on real data, hence leading to a smoother and more controlled training process.

During the experiments, we used a simple dense network with three hidden layers of 20, 40, 20 neurons and ReLU activation function in each layer. We focused on this simpler architecture instead of using more sophisticated ones like RNNs, LSTMs or GRUs to show that the proposed architecture can be used for a variety of problems. This time, we made the model function similar to a simple RNN by feeding it data containing several timesteps as input but this is not required; the architecture itself can be used for non-time step series data as well. Additionally, for handling time series data, we have considered using a neural network that does not only receive the data for the previous day and predicts the next day, but receives a sequence of t days as input, and makes predictions regarding a sequence of T days. We hand-picked the potential values for t and T , respectively, according to recent public forecasts that focus on the recent past and near future and kept T smaller than t , since predicting more days than what the network has information on would be impractical.

5. Influenza

To demonstrate the basic idea behind the architecture, we decided to start by showing how it can be used for known diseases, like Influenza. For these diseases, there are already some mathematically defined models, which are used heavily in practice due to their simplicity and good overall performances. Therefore, we will show how one such model, the SIR model performs on data for a few selected countries and how we can further improve the performance by using our proposed two-step architecture.

To measure the efficiency of these models, we used the available data of Germany, Hungary and Romania for the influenza season 2019 (starting from the winter of 2018 and ending in the spring of 2019). We chose these countries specifically from a bigger pool of countries by selecting those that had a data roughly resembling an SIR curve (see Figure 3). Our aim is to show that we can improve the overall performance of the original mathematical model even in cases where they perform relatively well.

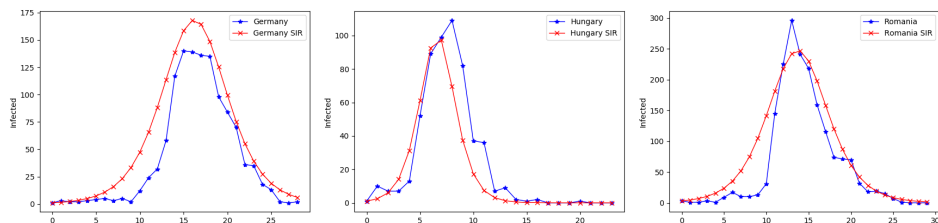


Figure 3. A comparison between real influenza data and the fitted SIR curves for Germany, Hungary and Romania.

The dataset we used contained the weekly number of newly infected people for any given time step. Since the data was aggregated at a weekly level, we used the configuration $t \in \{2, 4\}, T = 1$ for the neural network model. This way, it could process some relatively recent information (the last 2 or 4 weeks) without relying too much on older information (where $t > 4$), while keeping the model relatively simple ($T = 1$) and suitable for showcasing the potential of the architecture.

The predictions were evaluated by calculating the mean square error (MSE) and root mean square error (RMSE) metrics. Furthermore, for every country and configuration, we trained five different neural networks to calculate the spread of the errors. Table 1 shows the summarized results of both the SIR and the two-step architecture considering a confidence level of 95% ($p = 0.05, n = 5$, using t -statistics).

Table 1. The results of the SIR and the two-step architecture on the influenza dataset.

Country	Model	t	T	MSE	RMSE
Germany	SIR	-	-	603.88	24.57
Germany	Two-step	2	1	176.66 ± 49.96	13.23 ± 1.79
Germany	Two-step	4	1	170.67 ± 29.50	13.04 ± 1.14
Hungary	SIR	-	-	276.92	16.64
Hungary	Two-step	2	1	184.58 ± 21.57	13.57 ± 0.79
Hungary	Two-step	4	1	127.41 ± 15.36	11.28 ± 0.67
Romania	SIR	-	-	1452.93	38.12
Romania	Two-step	2	1	877.12 ± 124.16	29.58 ± 2.05
Romania	Two-step	4	1	1178.39 ± 175.33	34.28 ± 2.54

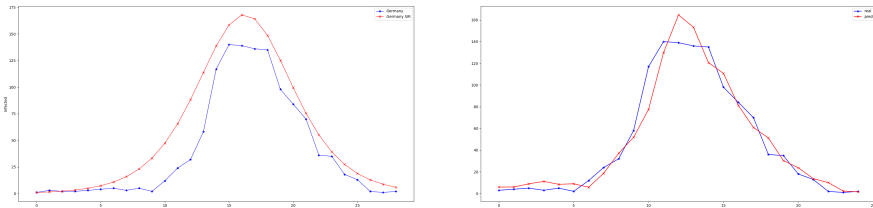


Figure 4. A comparison between the original SIR model (left) and one of the $t = 4, T = 1$ models (right) for Germany.

It can be seen that by using our proposed two-step neural network architecture, we were able to drastically decrease the overall error in our predictions. The improvements were the most drastic for the German and Romanian data, since while the SIR model fit relatively well to the real data, there were still a number of data points that were far away from the curves I of the models (see Figure 4). This shows that using our proposed architecture can further increase the overall performance even in cases where the original mathematical model performs well and thus can be a plausible solution for tackling the spread of some diseases to achieve

state-of-the-art results. In the next section, we will show how this approach can be used for predicting the spread of COVID-19.

6. Covid-19

To fully demonstrate the capabilities of the proposed architecture, we ran some experiments on COVID-19 data. We think that choosing this disease can better showcase the performance and reliability of our architecture, since at the time of writing this paper there are no mathematical models that can perform really well on COVID-19 data. As outlined previously, there are several factors, such as the numerous waves, noise in the data, regulations that make predicting COVID-19 hard or impossible for a single simple mathematical model and our architecture aims to overcome these hardships. Moreover, these factors make training a neural network harder, too, since the noise present in the dataset may mislead the networks during the training phase.

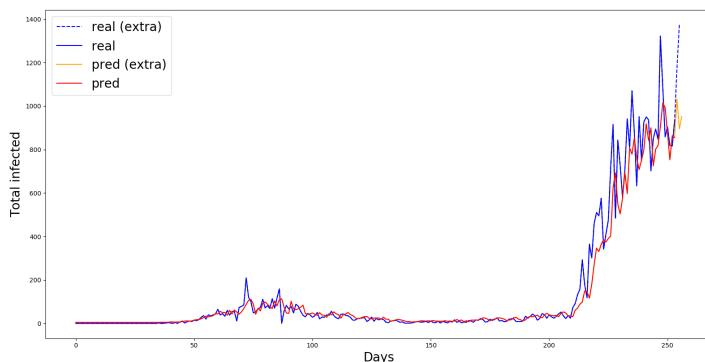


Figure 5. A model with the configuration $t = 7, T = 3$ and its predictions (red & orange) for Hungary.

During the experiments, we used different values for T and t , respectively to find out how many days' worth of data can better describe the disease as well as to improve the overall performance. Namely, we used the configurations $\{(t, T) \mid t \in \{14, 7, 3\} \text{ and } T \in \{7, 3, 1\} \text{ and } T < t\}$, since the available data was aggregated at a daily level. This way, we experimented with how many days the model should take into account when making predictions and find out whether increasing the size of the input resulted in any substantial performance gains. We also tried changing the output size to experiment with whether doing so could make the model more reliable by having it constantly focus on a series of next days. The dataset that we used contained the number of newly infected people for any given time step.

We found that using $t > 14$ made the training of the model much harder and resulted in models that performed worse due to making the model more complex and it focuses too much on days too far away and therefore does not contribute to

the current number of infected people. For a similar reason, we observed that when T was closer to t , the overall performance plummeted, since the model simply did not have enough information to make accurate long-term predictions. Moreover, we found that when $T > 3$, the quality of the predictions regarding the future started to deteriorate, suggesting that predictions with a larger output window size for the COVID-19 dataset are not yet feasible. Instead, we suggest that it is better to use smaller T values instead of trying to make long-term predictions (see Figure 5).

We trained all the models on the first wave of COVID-19 for any given country. This means that we first fit a SIR model to the data of the first wave, then approximated the curve I with a neural network, then trained it further on the real data of the first wave. Then, we tested the models on the second wave of COVID-19 for the given country. To test the overall reliability and performance of our proposed architecture, we compared its results with a plain neural network that was not initialized with weights similar to an SIR model but simply trained on the available COVID-19 data. These plain neural networks were also trained in the exact same way: first fit to the first wave of the real data and then tested on the second wave. We repeated each experiment for a given (t, T) pair a total of n times to measure how the results fluctuated. During our research, we found the number 10 to be the best for this purpose: the samples gathered proved to be representative enough to reliably calculate the overall error while the time required to train the models was still manageable. Tables 2, 3, 4 and 5 show the summarized results of both the two-step architecture (denoted as “Two-step”) and the plain neural network models (denoted as “Plain”), calculated with a confidence level of 95% ($p = 0.05, n = 10$, using t-statistics) for Hungary and Germany. We deliberately put more focus on these two countries since we wanted to examine how the spread of the disease can be modelled for Hungary and for a more developed country, like Germany. The results regarding the other countries can be found in Appendix, calculated with a confidence level of 95% but with a lower sample size of 3 ($p = 0.05, n = 3$, using t-statistics).

It can easily be seen that the results of the proposed architecture are generally way better than that of a simple, randomly initialized neural network. It shows that having the model learn a less complex and mathematically better defined function that roughly resembles the target data may be a beneficial pre-training step and could yield potentially better results when trained further on real data compared to models that are trained only on the latter. Another important note is that the target mathematical function does not need to match precisely with the real data, as it was the case for our research regarding COVID-19: the only important part is that it should contain some key information (in this case the bell-shape curve hinting that the number of diseases should keep increasing until a certain point and then start decreasing from then on) that can provide a strong foundation for the network to build upon in the second phase of the training. This approach also makes it harder for the model to focus on dispensable features due to the first step containing the differential equations in its loss function. Another interesting point is that training the network on a SIR model for the first wave is proved to

Table 2. Hungary - first wave errors.

t-T	Model	First wave errors			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	86.84 ± 20.39	-	9.21 ± 1.05	-
	Plain	271.51 ± 52.64	-	16.35 ± 1.54	-
7-1	Two-step	107.55 ± 19.33	-	10.29 ± 0.99	-
	Plain	186.41 ± 31.68	-	13.57 ± 1.16	-
3-1	Two-step	98.51 ± 11.96	-	9.89 ± 0.59	-
	Plain	141.14 ± 16.04	-	11.85 ± 0.68	-
14-7	Two-step	99.04 ± 31.11	78.53 ± 25.20	9.75 ± 1.49	8.66 ± 1.40
	Plain	301.53 ± 70.00	210.27 ± 57.76	17.19 ± 1.87	14.30 ± 1.79
14-3	Two-step	84.98 ± 12.25	66.66 ± 15.06	9.18 ± 0.63	8.07 ± 0.92
	Plain	290.72 ± 40.98	272.25 ± 104.48	16.98 ± 1.18	15.93 ± 3.24
7-3	Two-step	92.60 ± 20.33	81.15 ± 24.65	9.53 ± 1.01	8.82 ± 1.39
	Plain	214.27 ± 70.07	168.69 ± 73.63	14.30 ± 2.34	12.40 ± 2.92

Table 3. Hungary - second wave errors.

t-T	Model	Second wave errors			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	17249.43 ± 5150.43	-	129.28 ± 17.47	-
	Plain	31268.19 ± 7438.15	-	174.67 ± 20.78	-
7-1	Two-step	17881.75 ± 2883.75	-	132.85 ± 11.48	-
	Plain	25712.03 ± 5261.55	-	159.01 ± 15.62	-
3-1	Two-step	12987.38 ± 448.37	-	113.93 ± 1.97	-
	Plain	27127.49 ± 3578.38	-	164.07 ± 10.91	-
14-7	Two-step	12825.61 ± 1246.62	30125.69 ± 9607.12	113.02 ± 5.48	169.84 ± 26.97
	Plain	37160.11 ± 27293.57	51981.83 ± 5450.44	180.53 ± 50.97	227.41 ± 12.30
14-3	Two-step	14273.22 ± 2687.75	36985.18 ± 18002.58	118.57 ± 11.01	182.28 ± 46.23
	Plain	54495.87 ± 43075.81	42304.32 ± 11569.83	210.89 ± 75.48	202.34 ± 27.83
7-3	Two-step	13117.43 ± 2038.86	25895.53 ± 9037.72	113.91 ± 8.97	156.67 ± 27.69
	Plain	76711.68 ± 51465.54	53237.65 ± 23834.57	247.41 ± 93.88	221.32 ± 49.19

be really beneficial for predicting even the second wave, not only surpassing the original mathematical model but also proving that the network can learn important features present in the mathematical model which it can use to recognize similar patterns in future data and remarkably surpass the performance of plain neural networks.

Overall, this two-step approach made the training of the model easier and more manageable, since it is always easier to fine-tune a network to fit to a mathematically well-defined function. This also reduced the amount of noise the networks faced during training thanks to first being trained on a mathematical model and then switching to the real data with a smaller learning rate and an already robust set of weights instead of random ones. Moreover, we did not need any pre-configured network even though we basically pre-train the model, since the

Table 4. Germany - first wave errors.

t-T	Model	First wave errors (Germany)			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	153052.79 ± 58837.38	-	379.54 ± 71.56	-
	Plain	207112.48 ± 60990.25	-	446.17 ± 67.64	-
7-1	Two-step	129618.92 ± 35071.52	-	354.20 ± 48.64	-
	Plain	204516.15 ± 43125.16	-	447.09 ± 47.91	-
3-1	Two-step	112975.97 ± 13019.34	-	335.10 ± 19.68	-
	Plain	156238.49 ± 44879.63	-	386.93 ± 56.88	-
14-7	Two-step	111459.81 ± 39019.06	73770.07 ± 24833.80	324.50 ± 59.19	265.40 ± 43.53
	Plain	151080.24 ± 46846.55	218403.27 ± 112174.08	378.65 ± 66.20	441.36 ± 115.85
14-3	Two-step	113579.63 ± 29820.44	105877.50 ± 41614.79	332.29 ± 42.41	315.94 ± 58.69
	Plain	250274.30 ± 97146.08	243237.79 ± 132332.29	485.23 ± 91.82	460.28 ± 133.57
7-3	Two-step	116270.58 ± 26964.29	138751.71 ± 56922.80	336.79 ± 40.20	358.14 ± 77.22
	Plain	165839.62 ± 64844.88	184524.62 ± 85200.30	393.65 ± 78.66	412.91 ± 89.32

Table 5. Germany - second wave errors.

t-T	Model	Second wave errors (Germany)			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	386211.34 ± 55594.07	-	618.70 ± 44.12	-
	Plain	350296.40 ± 59203.48	-	588.17 ± 49.78	-
7-1	Two-step	373297.62 ± 40519.73	-	609.47 ± 32.37	-
	Plain	378415.93 ± 36600.51	-	613.73 ± 29.51	-
3-1	Two-step	396793.00 ± 23700.56	-	629.43 ± 18.60	-
	Plain	538979.60 ± 393800.78	-	684.51 ± 83.92	-
14-7	Two-step	244444.79 ± 55772.79	268252.21 ± 90832.01	489.20 ± 53.98	508.17 ± 75.45
	Plain	595380.90 ± 453947.98	268208.75 ± 66890.82	680.48 ± 274.30	510.72 ± 64.73
14-3	Two-step	284438.69 ± 34150.12	365277.10 ± 121480.28	531.59 ± 32.42	592.98 ± 88.09
	Plain	336347.85 ± 60559.36	404303.46 ± 128238.32	575.86 ± 51.85	624.77 ± 89.13
7-3	Two-step	286050.18 ± 27531.60	352900.11 ± 38205.91	533.76 ± 25.53	592.47 ± 32.73
	Plain	593034.40 ± 519253.12	326211.38 ± 56222.30	677.37 ± 276.24	567.29 ± 49.99

mathematical model can be relatively easily defined. This in turn provided a fast and cheap, yet effective way of using transfer learning.

7. Conclusion

In this paper we outlined a new two-step approach for training more accurate and reliable neural networks. The key idea we used was to let the model first train on a simplified version of the real data, which was a mathematical model adjusted to a part of the real data (i.e. the first wave in the case of COVID-19) and was defined by differential equations. This way, the models could first grasp the most important aspects of the data (i.e. spread, nature, bell-like shape etc.) without being affected by the outliers and noise being present in the real data, and then learn further on real data once their set of weights was already solid enough.

First, we showed how this approach performs on one simpler problem, which was predicting the weekly number of influenza patients and how it delivered better results than mathematical models that are currently used for this problem. Then we went one step further and showed how this approach fares with much noisier COVID-19 data, which currently no mathematical models can predict reliably. We summarized the results of the architecture for a number of countries and various configurations by changing the input and output size of the model and showed how it performs better than simple neural networks that are only trained on real data.

We also showed how this approach can combine the benefits of the two main pillars which were the mathematical models and the neural networks. For this, we showed how one model trained using this architecture can not only surpass plain neural networks that are initialized randomly but how the features regarding the spread of the disease extracted from the first wave can help with making predictions for the second wave, surpassing the original mathematical model, too, which could only predict a single wave.

Lastly, we presented how this method could be used as an easier type of transfer learning, where we do not need to download large pretrained neural networks, but can instead choose a mathematical model that roughly resembles the real data and have the neural network learn on that function. It is important to note that there are no restrictions regarding this mathematical model as it can be any kind of model as long as its outputs can be compared to a neural network's outputs. Therefore this architecture can be used in a number of disciplines and can be applied to solving a variety of problems. We also showed how this mathematical model does not need to perfectly fit the real data and how it is enough if the key features of the real data (spread, nature, etc.) are encoded in the mathematical model. This can save a lot of time during the training process while also giving the model a better mathematical foundation.

References

- [1] F. BRAUER, P. D. DRIESSCHE, J. WU: *Lecture Notes in Mathematical Epidemiology*, Berlin, Germany: Springer, 2008, DOI: <https://doi.org/10.1007/978-3-540-78911-6>.
- [2] T. HARKO, F. S. LOBO, M. MAK: *Exact analytical solutions of the Susceptible-Infected-Recovered (SIR) epidemic model and of the SIR model with equal death and birth rates*, Applied Mathematics and Computation 236 (2014), pp. 184–194, DOI: <https://doi.org/10.1016/j.amc.2014.03.030>.
- [3] *Johns Hopkins Coronavirus Resource Center*, URL: <https://coronavirus.jhu.edu/>.
- [4] W. O. KERMACK, A. G. MCKENDRICK: *A contribution to the mathematical theory of epidemics*, Proceedings of the Royal Society of London. Series A, Containing papers of a mathematical and physical character 115.772 (1927), pp. 700–721, DOI: <https://doi.org/10.1098/rspa.1927.0118>.
- [5] M. LANGLAIS, C. SUPPO: *A remark on a generic seirs model and application to cat retroviruses and fox rabies*, Mathematical and Computer Modelling 31.4-5 (2000), pp. 117–124, DOI: [https://doi.org/10.1016/S0895-7177\(00\)00029-7](https://doi.org/10.1016/S0895-7177(00)00029-7).

- [6] K. LEVENBERG: *A method for the solution of certain non-linear problems in least squares*, Quarterly of Applied Mathematics 2.2 (1944), pp. 164–168, DOI: <https://doi.org/10.1090/qam/10666>.
- [7] M. Y. LI, J. R. GRAEF, L. WANG, J. KARSAI: *Global dynamics of a SEIR model with varying total population size*, Mathematical Biosciences 160.2 (1999), pp. 191–213, DOI: [https://doi.org/10.1016/S0025-5564\(99\)00030-9](https://doi.org/10.1016/S0025-5564(99)00030-9).
- [8] D. W. MARQUARDT: *An algorithm for least-squares estimation of nonlinear parameters*, Journal of the Society for Industrial and Applied Mathematics 11.2 (1963), pp. 431–441, DOI: <https://doi.org/10.1137/0111030>.
- [9] D. OSTHUS, K. S. HICKMANN, P. C. CARAGEA, D. HIGDON, S. Y. DEL VALLE: *Forecasting seasonal influenza with a state-space SIR model*, The Annals of Applied Statistics 11.1 (2017), p. 202, DOI: <https://doi.org/10.1214/16-AOAS1000>.
- [10] R. STORN: *On the usage of differential evolution for function optimization*, in: Proceedings of North American Fuzzy Information Processing, IEEE, 1996, pp. 519–523.
- [11] *The Humanitarian Data Exchange Novel Coronavirus (COVID-19) Cases Data*, URL: <https://data.humdata.org/dataset/novel-coronavirus-2019-ncov-cases>.
- [12] S. TOWERS, K. V. GEISSE, Y. ZHENG, Z. FENG: *Antiviral treatment for pandemic influenza: Assessing potential repercussions using a seasonally forced SIR model*, Journal of Theoretical Biology 289 (2011), pp. 259–268, DOI: <https://doi.org/10.1016/j.jtbi.2011.08.011>.
- [13] *World bank total population data*, URL: https://data.worldbank.org/indicator/SP.POP.TOTL?name_desc=false.
- [14] *World Health Organization, FluNet*, URL: https://www.who.int/influenza/gisrs_laboratory/flunet/en/.
- [15] R. S. YADAV: *Mathematical Modeling and Simulation of SIR Model for COVID- 2019 Epidemic Outbreak: A Case Study of India*, medRxiv (2020), DOI: <https://doi.org/10.1101/2020.05.15.20103077>.
- [16] Z. ZAINUDDIN, O. PAULINE: *Function approximation using artificial neural networks*, WSEAS Transactions on Mathematics 7.6 (2008), pp. 333–338.
- [17] L. ZHOU, Y. WANG, Y. XIAO, M. Y. LI: *Global dynamics of a discrete age-structured SIR epidemic model with applications to measles vaccination strategies*, Mathematical Biosciences 308 (2019), pp. 27–37, DOI: <https://doi.org/10.1016/j.mbs.2018.12.003>.

Appendix

This chapter contains the results for the COVID-19 data for all countries except for Germany and Hungary, which were both shown previously. For these tables, we used $n = 3$, meaning that we ran the experiments a total of 3 times, unlike for Germany and Hungary, where n was 10. This is because while we mainly focused on those countries, we still experimented with others. The summarized results were calculated with a confidence level of 95% ($p = 0.05, n = 3$, using t-statistics).

		First wave errors (Austria)			
t-T	Model	Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	1515.32 ± 1319.10	-	38.52 ± 17.00	-
	Plain	12725.54 ± 11505.19	-	111.64 ± 49.19	-
7-1	Two-step	2295.17 ± 1322.18	-	47.68 ± 14.21	-
	Plain	8658.42 ± 11595.28	-	90.89 ± 60.53	-
3-1	Two-step	1598.95 ± 760.34	-	39.87 ± 9.28	-
	Plain	2745.88 ± 2389.63	-	51.90 ± 21.95	-
14-7	Two-step	2047.91 ± 577.55	2660.89 ± 1619.78	45.20 ± 6.50	51.30 ± 16.40
	Plain	15398.78 ± 33313.03	13124.99 ± 23690.55	114.48 ± 145.72	106.34 ± 129.65
14-3	Two-step	2321.18 ± 2669.21	1034.83 ± 1040.80	47.34 ± 27.18	31.66 ± 17.30
	Plain	11627.03 ± 8161.38	12458.17 ± 8198.15	107.07 ± 38.76	110.97 ± 36.35
7-3	Two-step	1320.84 ± 1098.38	2026.87 ± 1669.98	36.03 ± 14.56	44.64 ± 17.87
	Plain	3612.96 ± 1323.07	3141.33 ± 8950.15	59.99 ± 11.23	49.56 ± 79.61
		Second wave errors (Austria)			
t-T	Model	Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	18278.89 ± 20982.18	-	132.27 ± 85.15	-
	Plain	24284.74 ± 13589.91	-	155.20 ± 42.67	-
7-1	Two-step	12566.70 ± 5288.75	-	111.82 ± 23.95	-
	Plain	19281.21 ± 14584.62	-	137.85 ± 50.74	-
3-1	Two-step	8440.83 ± 1662.04	-	91.82 ± 9.00	-
	Plain	9971.46 ± 1728.25	-	99.82 ± 8.61	-
14-7	Two-step	19226.29 ± 13849.41	36166.18 ± 9576.15	137.75 ± 48.30	190.00 ± 24.82
	Plain	54091.63 ± 102159.87	44140.50 ± 19431.46	221.47 ± 216.03	209.57 ± 45.36
14-3	Two-step	18191.20 ± 16938.29	49582.47 ± 41318.60	133.37 ± 61.15	220.71 ± 89.60
	Plain	22559.15 ± 7545.53	27781.21 ± 15107.42	149.98 ± 24.76	165.96 ± 47.05
7-3	Two-step	13647.95 ± 5141.28	16497.89 ± 530.98	116.61 ± 21.58	128.44 ± 2.06
	Plain	50650.52 ± 161764.05	40733.20 ± 78861.44	194.40 ± 345.02	192.02 ± 86.11
		First wave errors (Croatia)			
t-T	Model	Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	220.89 ± 106.47	-	14.81 ± 3.67	-
	Plain	393.09 ± 242.98	-	19.71 ± 6.34	-
7-1	Two-step	224.54 ± 49.53	-	14.97 ± 1.68	-
	Plain	473.41 ± 435.75	-	21.51 ± 10.02	-
3-1	Two-step	190.21 ± 30.08	-	13.79 ± 1.09	-
	Plain	342.16 ± 210.90	-	18.41 ± 5.54	-
14-7	Two-step	196.23 ± 64.71	176.65 ± 6.04	13.99 ± 2.27	13.29 ± 0.23
	Plain	544.45 ± 796.18	486.36 ± 571.65	22.45 ± 19.40	21.67 ± 12.41
14-3	Two-step	170.69 ± 75.34	159.45 ± 59.65	13.03 ± 2.82	12.60 ± 2.40
	Plain	686.58 ± 475.52	392.85 ± 341.94	26.01 ± 9.55	19.61 ± 8.69
7-3	Two-step	197.37 ± 120.24	186.21 ± 54.89	13.98 ± 4.21	13.63 ± 2.00
	Plain	654.83 ± 521.17	288.30 ± 173.74	25.34 ± 10.80	16.89 ± 5.31
		Second wave errors (Croatia)			
t-T	Model	Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	4974.64 ± 5811.28	-	69.13 ± 42.51	-
	Plain	4165.83 ± 300.80	-	64.54 ± 2.32	-
7-1	Two-step	4749.04 ± 2851.06	-	68.60 ± 20.10	-
	Plain	3323.75 ± 1121.54	-	57.56 ± 9.88	-
3-1	Two-step	3577.49 ± 421.19	-	59.80 ± 3.52	-
	Plain	3999.04 ± 1063.80	-	63.18 ± 8.43	-
14-7	Two-step	4858.05 ± 1388.48	7928.49 ± 2046.42	69.62 ± 9.83	88.96 ± 11.53
	Plain	9508.23 ± 19872.40	4747.52 ± 3963.52	90.82 ± 107.99	68.17 ± 30.58
14-3	Two-step	4142.48 ± 2595.50	5816.76 ± 3388.36	64.04 ± 19.68	75.91 ± 22.41
	Plain	15893.52 ± 25960.88	6649.36 ± 5851.83	119.34 ± 123.66	80.64 ± 36.75
7-3	Two-step	3483.77 ± 985.94	5306.68 ± 2668.54	58.96 ± 8.39	72.61 ± 17.87
	Plain	15097.58 ± 26632.08	8454.94 ± 6991.72	114.68 ± 134.20	91.13 ± 37.35
		First wave errors (Japan)			
t-T	Model	Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	31173.83 ± 12835.80	-	176.14 ± 37.05	-
	Plain	26445.08 ± 7112.04	-	162.46 ± 22.21	-
7-1	Two-step	2172.84 ± 2015.93	-	45.98 ± 23.26	-
	Plain	3966.92 ± 2946.28	-	62.54 ± 22.68	-
3-1	Two-step	1163.46 ± 903.46	-	33.81 ± 13.69	-
	Plain	1789.36 ± 940.17	-	42.13 ± 11.50	-

14-7	Two-step	1135.15 ± 984.16	1174.94 ± 414.79	33.35 ± 14.56	34.22 ± 6.09
	Plain	3861.56 ± 9161.62	5227.93 ± 6190.41	57.64 ± 70.66	70.98 ± 41.86
14-3	Two-step	1845.98 ± 2939.82	1679.00 ± 2505.62	41.63 ± 32.39	39.72 ± 30.66
	Plain	5590.88 ± 6236.25	4202.21 ± 10617.89	73.60 ± 40.12	59.36 ± 79.26
7-3	Two-step	2214.84 ± 659.38	3048.57 ± 680.96	47.05 ± 3.85	55.18 ± 6.13
	Plain	2938.19 ± 4027.50	3394.17 ± 3026.15	52.38 ± 42.42	57.68 ± 25.01
Second wave errors (Japan)					
t-T	Model	Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	31173.83 ± 12835.80	-	176.14 ± 37.06	-
	Plain	26445.08 ± 7112.04	-	162.45 ± 22.21	-
7-1	Two-step	36345.59 ± 14877.43	-	190.23 ± 38.48	-
	Plain	27978.35 ± 11084.69	-	166.89 ± 34.01	-
3-1	Two-step	30483.74 ± 3487.63	-	174.57 ± 9.94	-
	Plain	28974.52 ± 5629.58	-	170.13 ± 16.69	-
14-7	Two-step	34384.30 ± 11577.81	97623.25 ± 257667.12	185.15 ± 30.74	284.64 ± 392.01
	Plain	174277.57 ± 577842.89	104793.24 ± 185061.89	351.93 ± 683.19	311.15 ± 271.78
14-3	Two-step	43934.07 ± 16622.80	149252.23 ± 46094.94	209.19 ± 40.18	385.83 ± 59.77
	Plain	46122.41 ± 13903.01	59816.83 ± 37494.31	214.51 ± 31.86	243.26 ± 76.98
7-3	Two-step	37621.15 ± 9175.78	44685.11 ± 5812.97	193.81 ± 23.56	211.34 ± 13.72
	Plain	170816.99 ± 583602.03	41983.28 ± 30683.75	346.46 ± 685.63	203.51 ± 72.44

t-T	Model	First wave errors (Romania)			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	2003.19 ± 1241.37	-	44.50 ± 14.49	-
	Plain	2854.63 ± 2958.38	-	52.67 ± 27.29	-
7-1	Two-step	1892.33 ± 1360.64	-	43.19 ± 15.78	-
	Plain	3562.15 ± 1736.53	-	59.50 ± 14.41	-
3-1	Two-step	2334.57 ± 750.72	-	48.25 ± 7.65	-
	Plain	2831.42 ± 1387.99	-	53.05 ± 12.74	-
14-7	Two-step	2094.13 ± 1347.42	4506.94 ± 4762.57	45.49 ± 5.27	65.93 ± 38.55
	Plain	2777.60 ± 2285.47	2595.81 ± 1950.88	52.25 ± 20.99	50.53 ± 19.92
14-3	Two-step	2120.78 ± 741.48	9232.91 ± 14948.22	45.97 ± 8.23	91.09 ± 93.08
	Plain	5255.90 ± 2531.97	4891.36 ± 3755.55	72.28 ± 17.06	69.37 ± 27.12
7-3	Two-step	1876.50 ± 1312.50	2116.70 ± 1138.65	43.03 ± 15.08	45.81 ± 12.81
	Plain	4089.11 ± 1796.34	3370.65 ± 2518.38	63.78 ± 13.96	57.58 ± 22.50
Second wave errors (Romania)					
t-T	Model	Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	71877.76 ± 45011.52	-	266.62 ± 85.63	-
	Plain	98788.67 ± 96281.68	-	310.52 ± 147.90	-
7-1	Two-step	85973.31 ± 45135.96	-	292.12 ± 76.88	-
	Plain	110134.85 ± 55830.77	-	330.67 ± 85.71	-
3-1	Two-step	76012.83 ± 22299.81	-	275.38 ± 40.37	-
	Plain	111523.80 ± 45812.08	-	333.22 ± 67.23	-
14-7	Two-step	47787.30 ± 31708.20	235030.18 ± 303607.15	217.30 ± 72.52	470.78 ± 352.09
	Plain	57212.18 ± 8566.19	190334.16 ± 274741.94	239.12 ± 18.06	425.03 ± 299.34
14-3	Two-step	70061.01 ± 21365.08	437848.72 ± 772867.92	264.37 ± 39.91	617.43 ± 724.00
	Plain	118355.19 ± 92843.39	128156.15 ± 64752.80	341.25 ± 132.73	356.81 ± 88.27
7-3	Two-step	92617.64 ± 59595.46	162631.00 ± 270817.29	302.72 ± 95.06	389.54 ± 317.54
	Plain	109021.98 ± 48827.75	175688.98 ± 294938.94	329.31 ± 73.05	404.48 ± 334.46
First wave errors (Slovakia)					
t-T	Model	Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	364.20 ± 363.90	-	18.79 ± 10.15	-
	Plain	850.22 ± 276.73	-	29.12 ± 4.70	-
7-1	Two-step	254.09 ± 105.92	-	15.90 ± 3.28	-
	Plain	417.64 ± 232.20	-	20.35 ± 5.57	-
3-1	Two-step	245.41 ± 42.92	-	15.66 ± 1.37	-
	Plain	272.86 ± 75.48	-	16.50 ± 2.32	-
14-7	Two-step	364.77 ± 512.76	419.50 ± 208.44	18.62 ± 12.91	20.42 ± 5.01
	Plain	910.09 ± 693.09	753.42 ± 723.37	29.90 ± 12.14	27.05 ± 14.20
14-3	Two-step	356.35 ± 381.45	371.54 ± 568.87	18.57 ± 10.29	18.41 ± 17.34
	Plain	918.37 ± 333.55	1428.92 ± 243.88	30.25 ± 5.46	37.79 ± 3.26
7-3	Two-step	358.69 ± 102.05	332.77 ± 262.37	18.92 ± 2.74	18.08 ± 7.43
	Plain	391.13 ± 252.35	322.46 ± 302.11	19.67 ± 6.26	17.71 ± 9.06
Second wave errors (Slovakia)					
t-T	Model	Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	14912.62 ± 7672.64	-	121.66 ± 32.10	-
	Plain	16968.34 ± 14892.37	-	128.98 ± 55.54	-
7-1	Two-step	17242.32 ± 6597.18	-	131.05 ± 25.08	-
	Plain	19465.78 ± 10695.60	-	138.97 ± 37.73	-
3-1	Two-step	16635.98 ± 1603.47	-	128.96 ± 6.19	-
	Plain	26667.26 ± 8118.43	-	163.09 ± 25.29	-
14-7	Two-step	4511.25 ± 1665.30	9543.07 ± 6423.34	67.04 ± 12.68	97.08 ± 33.07
	Plain	6347.09 ± 3009.71	12430.07 ± 5152.36	79.43 ± 18.69	111.23 ± 23.33
14-3	Two-step	12988.05 ± 3375.90	17975.61 ± 13027.75	113.86 ± 14.64	133.18 ± 47.01
	Plain	16988.85 ± 6788.30	17755.88 ± 6476.31	130.07 ± 25.53	133.02 ± 23.89
7-3	Two-step	13405.69 ± 950.71	19663.17 ± 12020.02	115.78 ± 4.09	139.55 ± 41.78
	Plain	30500.83 ± 60495.42	25222.86 ± 15568.28	166.17 ± 163.52	157.98 ± 49.47

t-T	Model	First wave errors (Slovenia)			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	77.79 +- 26.09	-	8.81 +- 1.46	-
	Plain	194.75 +- 189.55	-	13.75 +- 7.30	-
7-1	Two-step	94.58 +- 35.24	-	9.71 +- 1.83	-
	Plain	163.92 +- 141.55	-	12.66 +- 5.73	-
3-1	Two-step	69.73 +- 6.29	-	8.35 +- 0.38	-
	Plain	120.07 +- 36.98	-	10.94 +- 1.68	-
14-7	Two-step	83.68 +- 40.57	84.27 +- 36.23	9.12 +- 2.28	9.16 +- 1.93
	Plain	169.05 +- 139.79	124.69 +- 139.62	12.88 +- 5.40	10.99 +- 5.97
14-3	Two-step	66.58 +- 37.72	65.08 +- 44.48	8.13 +- 2.29	8.02 +- 2.76
	Plain	246.59 +- 106.49	148.34 +- 131.93	15.66 +- 3.48	12.03 +- 5.72
7-3	Two-step	85.38 +- 40.53	93.02 +- 31.76	9.21 +- 2.22	9.63 +- 1.68
	Plain	115.81 +- 50.61	132.63 +- 70.59	10.73 +- 2.41	11.47 +- 3.12
t-T	Model	Second wave errors (Slovenia)			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	1022.18 +- 930.44	-	31.60 +- 14.72	-
	Plain	1247.82 +- 457.40	-	35.26 +- 6.55	-
7-1	Two-step	1266.76 +- 701.12	-	35.44 +- 10.01	-
	Plain	1349.53 +- 894.06	-	36.53 +- 11.92	-
3-1	Two-step	1106.60 +- 194.35	-	33.25 +- 2.91	-
	Plain	1506.04 +- 601.12	-	38.73 +- 7.66	-
14-7	Two-step	543.05 +- 220.61	1123.13 +- 974.21	23.25 +- 4.76	33.18 +- 14.26
	Plain	526.60 +- 136.71	1116.75 +- 832.46	22.93 +- 2.94	33.16 +- 12.51
14-3	Two-step	785.70 +- 31.57	1328.43 +- 1375.75	28.03 +- 0.56	35.94 +- 18.37
	Plain	1202.36 +- 524.14	2294.93 +- 3308.89	34.58 +- 7.76	46.53 +- 34.71
7-3	Two-step	1164.56 +- 828.97	1303.17 +- 444.59	33.88 +- 12.38	36.04 +- 6.26
	Plain	1544.92 +- 935.20	1535.80 +- 171.83	39.10 +- 12.14	39.18 +- 2.20

t-T	Model	First wave errors (Switzerland)			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	1650.36 ± 1237.49	-	40.28 ± 16.00	-
	Plain	26767.85 ± 31289.39	-	159.86 ± 24.61	-
7-1	Two-step	1751.15 ± 1829.87	-	41.11 ± 23.86	-
	Plain	3134.05 ± 3333.79	-	55.16 ± 29.07	-
3-1	Two-step	1434.07 ± 1915.86	-	36.71 ± 28.35	-
	Plain	1829.81 ± 184.46	-	42.77 ± 2.17	-
14-7	Two-step	10433.31 ± 28717.23	19846.34 ± 30024.83	90.86 ± 141.96	135.19 ± 120.53
	Plain	38316.98 ± 37304.36	81278.59 ± 58801.01	192.92 ± 100.77	283.09 ± 102.70
14-3	Two-step	4861.04 ± 3697.90	8565.31 ± 6611.52	69.15 ± 27.21	91.75 ± 36.96
	Plain	15490.40 ± 34452.23	23939.43 ± 68419.65	109.28 ± 181.19	136.68 ± 220.61
7-3	Two-step	2204.43 ± 1265.82	1479.59 ± 1492.09	46.73 ± 13.81	37.84 ± 21.07
	Plain	2454.06 ± 4000.73	6121.91 ± 12599.90	47.31 ± 44.66	69.21 ± 111.02
t-T	Model	Second wave errors (Switzerland)			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	114978.94 ± 33095.94	-	338.71 ± 48.59	-
	Plain	127891.97 ± 13207.35	-	357.57 ± 18.58	-
7-1	Two-step	128959.18 ± 59863.34	-	358.11 ± 81.48	-
	Plain	118014.09 ± 4629.21	-	343.52 ± 6.74	-
3-1	Two-step	143138.45 ± 17755.58	-	378.26 ± 23.46	-
	Plain	128062.63 ± 15330.01	-	357.79 ± 21.30	-
14-7	Two-step	33776.03 ± 18393.62	35900.58 ± 19553.14	183.07 ± 49.24	188.68 ± 52.65
	Plain	46881.59 ± 19283.71	70863.13 ± 43875.14	216.03 ± 44.51	264.83 ± 82.05
14-3	Two-step	43197.12 ± 23294.05	45179.99 ± 10374.36	207.01 ± 56.27	212.41 ± 24.12
	Plain	65516.57 ± 73767.49	56636.31 ± 30539.63	251.92 ± 137.84	236.98 ± 66.57
7-3	Two-step	47442.24 ± 12857.64	45676.70 ± 9910.13	217.59 ± 29.78	213.58 ± 23.50
	Plain	58206.81 ± 89425.72	47090.02 ± 39493.20	234.25 ± 175.72	215.08 ± 87.70
t-T	Model	First wave errors (USA)			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	1.21e+07 ± 1.55e+07	-	3403.95 ± 2147.00	-
	Plain	9.91e+06 ± 2.53e+06	-	3086.60 ± 1874.68	-
7-1	Two-step	2.07e+07 ± 8.19e+06	-	4543.58 ± 881.82	-
	Plain	2.02e+07 ± 8.00e+06	-	4482.86 ± 872.58	-
3-1	Two-step	1.20e+07 ± 1.28e+06	-	3458.82 ± 186.73	-
	Plain	1.61e+07 ± 6.62e+06	-	4004.74 ± 808.62	-
14-7	Two-step	1.53e+07 ± 1.02e+07	5.83e+07 ± 1.11e+08	3885.53 ± 1263.98	7287.40 ± 6927.27
	Plain	9.50e+06 ± 4.62e+06	1.44e+07 ± 1.91e+07	3072.02 ± 773.72	3717.61 ± 2396.30
14-3	Two-step	7.50e+06 ± 6.73e+06	8.11e+07 ± 3.09e+08	2705.74 ± 1281.45	7015.28 ± 17171.39
	Plain	1.42e+07 ± 9.26e+06	1.11e+08 ± 2.21e+08	3749.50 ± 1252.22	10010.71 ± 9951.60
7-3	Two-step	2.37e+07 ± 8.17e+06	3.71e+07 ± 5.44e+07	4863.90 ± 824.99	5926.72 ± 4242.28
	Plain	4.04e+08 ± 8.25e+08	1.23e+08 ± 2.91e+08	17765.36 ± 28567.51	10076.78 ± 13949.80
t-T	Model	Second wave errors (USA)			
		Test set MSE		Test set RMSE	
		Next Day	Aggregated	Next Day	Aggregated
14-1	Two-step	4.63e+07 ± 6.02e+07	-	6653.56 ± 4331.68	-
	Plain	5.43e+07 ± 6.71e+07	-	7199.30 ± 4825.38	-
7-1	Two-step	5.29e+07 ± 2.24e+07	-	7258.06 ± 1566.00	-
	Plain	4.67e+07 ± 7.05e+06	-	6835.16 ± 511.52	-

3-1	Two-step	3.50e+07 ± 2.77e+06	-	5918.34 ± 235.20	-
	Plain	4.39e+07 ± 3.25e+07	-	6578.76 ± 2375.17	-
14-7	Two-step	5.69e+07 ± 4.11e+07	2.79e+08 ± 4.44e+08	7494.99 ± 2638.09	16168.56 ± 12595.59
	Plain	7.91e+07 ± 9.11e+07	1.20e+08 ± 8.87e+07	8745.85 ± 4903.62	10872.10 ± 4007.51
14-3	Two-step	3.40e+07 ± 7.37e+06	3.23e+08 ± 1.16e+09	5831.17 ± 637.36	14644.18 ± 31712.39
	Plain	5.43e+07 ± 2.65e+07	5.03e+08 ± 7.40e+08	7340.15 ± 1856.98	21833.43 ± 15675.72
7-3	Two-step	6.91e+07 ± 6.30e+07	1.59e+08 ± 2.90e+08	8228.62 ± 3641.60	12046.02 ± 11162.70
	Plain	1.21e+09 ± 2.51e+09	3.78e+08 ± 1.01e+09	30522.01 ± 51160.81	17268.64 ± 27200.69

Post-processing methods for calibrating the wind speed forecasts in central regions of Chile*

Mailiu Díaz^{ab}, Orietta Nicolis^{bc}, Julio César Marín^{ac},
Sándor Baran^d

^aDepartamento de Meteorología, Universidad de Valparaíso, Chile
mailiudp@gmail.com

^bFacultad de Ingeniería, Universidad Andres Bello, Chile
orietta.nicolis@unab.cl

^cCentro Interdisciplinario de Estudios Atmosféricos y Astro-Estadística,
Universidad de Valparaíso, Chile
julio.marin@meteo.uv.cl

^dFaculty of Informatics, University of Debrecen, Hungary
baran.sandor@inf.unideb.hu

Submitted: December 22, 2020

Accepted: March 17, 2021

Published online: May 18, 2021

Abstract

In this paper we propose some parametric and non-parametric post-processing methods for calibrating wind speed forecasts of nine Weather Research and Forecasting (WRF) models for locations around the cities of Valparaíso and Santiago de Chile (Chile). The WRF outputs are generated with different planetary boundary layers and land-surface model parametrizations and they are calibrated using observations from 37 monitoring stations. Statistical calibration is performed with the help of ensemble model output statistics and quantile regression forest (QRF) methods both with regional and semi-local approaches. The best performance is obtained by the QRF using a semi-local approach and considering some specific weather variables from WRF simulations.

*This research was partially supported by the Interdisciplinary Center of Atmospheric and Astro-Statistical Studies, University of Valparaíso, Chile.

Keywords: Ensemble model output statistics, ensemble calibration, quantile regression forest, statistical post-processing, wind speed

1. Introduction

Numerical weather prediction (NWP) models have been used for many years in research and operational weather forecasting due to their advantage to simulate the state of the atmosphere in any region of the globe at high spatial and temporal resolutions. The Weather Research and Forecasting (WRF) model [32, 37] is one of the most widely used systems, which has received strong support by the atmospheric science community over the years. However, despite its continuous improvement and successful use, the model still presents biases in the prediction of near-surface variables; specifically, in the prediction of wind speed over complex terrain [23, 33, 35], which may be related to the smoothed topography used in the model and the misrepresentation of small-scale atmospheric processes [19, 23]. Those limitations negatively influence obtaining accurate predictions of surface wind speed and direction, which are used in a large number of applications in Chile, such as wind energy [24, 28], air-quality [6, 33, 36], and precipitation over the Andes cordillera [9].

In the last 15 years several statistical post-processing models have been developed to obtain sharp and calibrated forecasts, e.g. the non-homogeneous regression or ensemble model output statistics [EMOS; 13], which method provides full predictive distribution of the future weather quantity using a single parametric distribution with parameters connected to the ensemble members.

Recently, some studies have been focused on the use of machine learning techniques for statistical post-processing. [39] introduced a new post-processing method based on quantile regression forests (QRF), which is a generalization of random forests and allows the estimation of conditional quantiles from the cumulative distribution function (CDF) in an efficient and simple way. This approach has the important advantage of allowing the inclusion of other features as predictors in the post-processing model. One can also mention [34], where QRF is applied to improve 2m temperature forecasts and a flexible alternative using a neural network is also proposed.

In [7], we evaluated two parametric models for calibrating surface temperature forecasts from nine WRF simulations at 19 meteorological stations in Santiago city. Now, the main aim of this study is to compare the forecast skill of some parametric and non-parametric post-processing methods to calibrate the wind speed using both regional and semi-local approaches at 37 monitoring stations around Valparaíso and Santiago city. Furthermore, the importance of each simulation and that of other weather variables from WRF included as predictors in QRF are examined on the basis of the continuous ranked probability score (CRPS).

The paper is organized as follows. Section 2 provides a description of data from meteorological stations, WRF configurations and variables included in the study, and a preliminary statistical analysis considering the error forecast and verification

rank. Methods for modeling wind speed and some verification tools are described in Section 3. The results of statistical post-processing are given in Section 4 with a comparison of the various approaches and with specification of the importance of ensemble members and other included variables. Finally, Section 5 presents the major results and some possible future extensions.

2. Description of the data and preliminary analysis

Nine WRF simulations were generated for the period between 1 June 2017 and 30 January 2018 to predict wind speed at 37 meteorological stations, around Valparaíso and Santiago cities, using the same configurations of [7]. The WRF simulations and the data from monitoring stations are briefly described below. A preliminary analysis of the forecasts is also provided.

2.1. WRF configurations and data description

The Advanced Research WRF core (ARW-WRF) [37] Version 3.7.1 was employed to generate a nine-member forecast ensemble using three nested domains at 18 km, 6 km and 2 km horizontal resolutions (see Figure 1a) and 44 vertical levels with variable resolution between 60 and 200 m from 1 June 2017 to 30 January 2018 at 3 hour time steps from 00 UTC to 21 UTC. The detailed description of nine ensemble members is presented in [7].

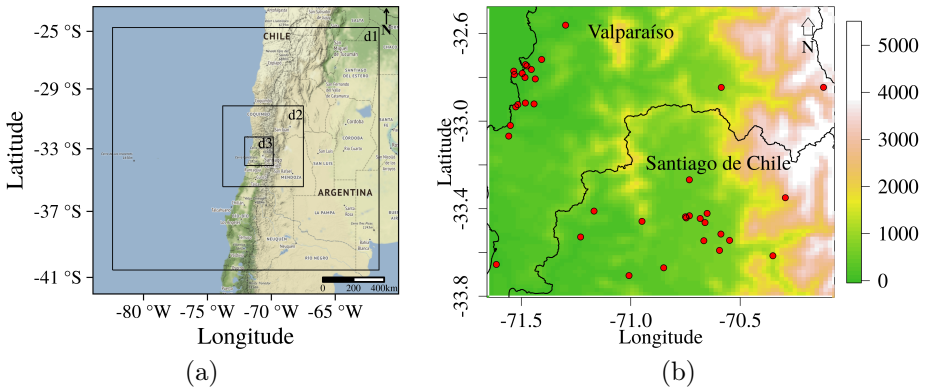


Figure 1. (a) Representation of the domains 1, 2 and 3 used in the WRF model at 18 km, 6 km, and 2 km horizontal resolutions respectively and (b) Altitude map and the location of monitoring stations represented with red points.

The ensemble members differ both in the applied planetary boundary layer (PBL) and land-surface model (LSM) parametrizations. The Land-surface processes are represented by the 5-layer [8], Noah [5] and Pleim-Xiu [31] schemes; we use the Mellor-Yamada-Janjic (MYJ) [18], Yonsei University (YSU) [16] and

Mellor-Yamada Nakanishi and Ninno 2.5 level (MYNN) [29] schemes to represent the PBL and surface layer processes.

The rest of the parametrizations are kept the same in all simulations: Kain-Fritsch (Kain-F) [20] cumulus parametrization, the Rapid Radiative Transfer Model (RRTMG) [17] to represent the long wave and short wave radiative processes, and the WRF single-moment 3-class (WSM3) [15] scheme to represent microphysics processes.

The initial and boundary conditions were provided by the Final Operational Global Analysis (FNL) at 0.25×0.25 degrees horizontal resolution every 6 hours to obtain the variables described in Table 1 from the highest resolution domain (d3).

Table 1. The variables included in the study from the highest resolution domain (d3) of WRF simulations.

Nomenclature	Description	Unit
XLONG	Longitude	degree-east
XLAT	Latitude	degree-north
U10	Zonal (East-West) wind component at 10 m	m s^{-1}
V10	Meridional (North-South) wind component at 10 m	m s^{-1}
T2	Temperature at 2 m	K
PSFC	Surface pressure	Pa
Q2	Specific humidity at 2 m	kg kg^{-1}
VAR	Orographic variance	
LU	Land use category	
HGT	Terrain height	m

Wind speed and relative humidity forecasts are obtained from the predictions of variables described in Table 1: 10 m wind speed equals $WS = \sqrt{U10^2 + V10^2}$ and it is expressed in m s^{-1} , whereas 2 m relative humidity is obtained using an approximation of equations presented by [3], namely

$$RH = Q2 / ((pq_0 / PSFC) \exp\{a_2(T2 - a_3) / (T2 - a_4)\}), \quad (2.1)$$

where $pq_0 = 379.91$, $a_2 = 17.27$, $a_3 = 273.16$ and $a_4 = 35.86$. The values obtained by equation (2.1) are normed to 1 and referred to as percents.

Finally, the corresponding 3 hourly verifying observations of 10 m wind speed for the same time period 1 June 2017 - 30 January 2018 measured in 37 monitoring stations around Valparaíso and Santiago city (see Figure 1b) were downloaded from the Dirección Meteorológica de Chile (<http://www.meteochile.gob>) and the National System for Air Quality (<https://sinca.mma.gob.cl/>). The stations have different altitudes represented in meters from 0 to 3000 m, see Figure 1 (b).

2.2. Preliminary data analysis

Consider first the dependence of the forecast error of an individual member of the WRF ensemble forecast on the location of the monitoring station. In Figure 2 the

box-plots of forecast errors corresponding to different stations are given, arranged in ascending order of station altitude. The median error at all stations is close to 0; however, as expected, stations above 2000 m located in the mountain zone of Santiago de Chile exhibit the highest forecast errors (last two boxplots in Figure 2), which is nicely in line with the results of [25].

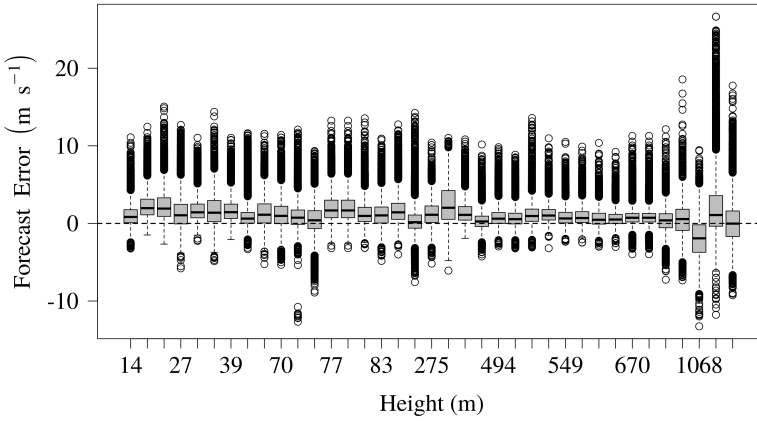


Figure 2. Forecast error at each station sorted by the altitude in meters.

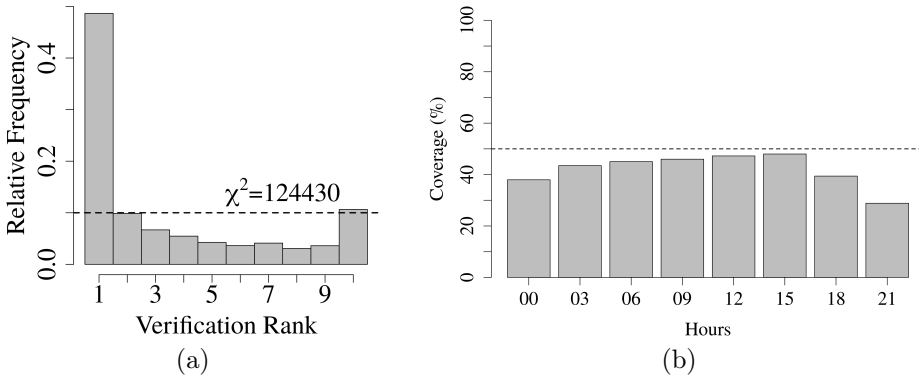


Figure 3. (a) Verification rank histogram for the total period and (b) the percentage of observed values included in the range of the ensemble forecasts at each hour for the all period.

Further, to get an idea about the calibration, Figures 3a,b show the verification rank histogram of raw wind speed ensemble forecasts and the coverage for the different observation times, respectively. The verification rank is the rank of the validating observation with respect to the corresponding ensemble members and in

the case of proper calibration it should be uniformly distributed (see e.g. Section 9.7.1 [42]), whereas the coverage is the proportion of observed values included in the range of the forecasts. Since, the resulted histogram doesn't follow a uniform distribution as confirmed by the high value of the χ^2 test statistic, a great difference between the observed and the expected frequencies is supposed to exist. In addition, there is no observation hour when the coverage exceeds 50% with the lowest value of 41.99% at 21 UTC (see Figure 3b), which proportions are far from the nominal 80% coverage of a calibrated 9-member ensemble forecast.

These preliminary results indicate that the ensemble forecasts are biased and not properly calibrated calling for some form of statistical post-processing.

3. Methodology

As wind speed data are characterized by non-negative values and the observations do not follow a symmetric law, they cannot be described by a normal distribution as temperature or air pressure. For this reason, to model wind speed the use of skewed and non-negative distributions, such as a truncated normal [1, 41], log-normal [2] or gamma [38] distribution are proposed.

Some post-processing approaches to calibrate wind speed forecasts and the tools to assess the forecast skill of the models are described below.

3.1. EMOS using truncated normal distribution

The Ensemble Model Output Statistics (EMOS) or non-homogeneous regression approach, proposed by [13], is one of the most used parametric post-processing techniques. EMOS models for various weather variables differ in the predictive distribution family and in the link functions connecting the parameters of the predictive distribution to the ensemble members. Following [41], as parametric family we consider a truncated normal (TN) distribution $\mathcal{N}_0(\mu, \sigma)$ with location μ , scale $\sigma > 0$ and cut-off equal at 0, defined by probability density function (PDF)

$$f(x | \mu, \sigma) := \begin{cases} \frac{1}{\sigma} \phi((x - \mu)/\sigma) / \Phi(\mu/\sigma), & \text{if } x \geq 0, \\ 0, & \text{otherwise,} \end{cases} \quad (3.1)$$

[41], where ϕ and Φ are the PDF and the cumulative distribution function (CDF) of a standard normal distribution, respectively.

The TN EMOS predictive distribution considering 9 WRF ensemble members is defined as

$$\mathcal{N}_0(a_0 + a_1 f_1 + \dots + a_9 f_9, b_0 + b_1 S^2), \quad \text{where } S^2 := \frac{1}{8} \sum_{k=1}^9 (f_k - \bar{f})^2,$$

with \bar{f} denoting the ensemble mean. Location parameters $a_0, a_1, \dots, a_9 \in \mathbb{R}$, $a_1, \dots, a_9 \geq 0$ and scale parameters $0 \leq b_0, b_1 \in \mathbb{R}$ are estimated over the training

data consisting of ensemble members and verifying observations from the preceding n days (rolling training period) by optimizing the mean of a certain proper verification score (see [41] for more details), which in our case is the continuous ranked probability score (CRPS) described in detail in Section 3.2.

3.2. Diagnostics

[11] defines the aim of statistical post-processing as maximization of the sharpness of the predictive distribution subject to calibration, where the latter expresses a statistical consistency between forecasts and observations, whereas the former indicates the forecast accuracy.

For doing a simultaneous evaluation of calibration and sharpness, [12] suggests the use of the continuous ranked probability score (CRPS), which for a given CDF $F(y)$ and observation x is defined as

$$\text{CRPS}(F, x) := \int_{-\infty}^{\infty} (F(y) - \mathbb{1}_{\{y \geq x\}})^2 dy = \mathbb{E}|X - x| - \frac{1}{2}\mathbb{E}|X - X'|,$$

where $\mathbb{1}_{\{y \geq x\}}$ denotes the indicator function which is 1 if $y \geq x$ and 0 otherwise, while X and X' are independent random variables with CDF F and finite first moment. For wind speed, similar to observations and forecasts, this score is expressed in m s^{-1} and it is a negatively oriented scoring rule, that is the smaller the better. For comparing predictive performance of different probabilistic forecasts one usually considers the mean CRPS over all forecasts and observations of the verification data denoted by $\overline{\text{CRPS}}$.

In addition, to assess the relative improvement of a forecast with respect to a given reference forecast, one can calculate the continuous ranked probability skill score (CRPS.S) [12],

$$\text{CRPS.S} = 1 - \frac{\overline{\text{CRPS}}}{\overline{\text{CRPS}}_{\text{ref}}},$$

where $\overline{\text{CRPS}}_{\text{ref}}$ denotes the mean CRPS of the reference forecast over the verification data.

Further, calibration can also be investigated by examining the coverage of the $(1 - \alpha)100\%$ central prediction interval with $\alpha \in (0, 1)$, i.e. by calculating the proportion of validating observations located between the lower and upper $\alpha/2$ quantiles of the predictive distribution. In the case of proper calibration the coverage should be around $(1 - \alpha)100\%$, and in order to provide a fair comparison with the raw ensemble one usually chooses the value of α to match the nominal coverage of the raw ensemble (80% for the 9-member ensemble).

The predictive performance of point forecasts can be assessed by considering the root mean of the squared error $SE(x, y) = (x - y)^2$ (RMSE) and mean of the absolute error $AE(x, y) = |x - y|$ (MAE) based on the forecast y and the observation x [10, 30] taken over all forecast cases in the verification data.

Finally, the probability integral transform (PIT) histograms (see e.g. [42]) might be used for a visual perception of the improvement in calibration compared with the raw ensemble. The PIT is the value of the predictive CDF evaluated at the validating observation and for a calibrated forecast PIT has to follow a uniform law on the $[0, 1]$ interval. Hence, the PIT histogram is the continuous counterpart of the verification rank histogram of the raw ensemble.

3.3. Quantile Regression Forests

As an alternative to parametric post-processing, [39] and [34] recently applied the Quantile Regression Forest (QRF) model for calibrating ensemble forecasts. This model was originally introduced by [26] as an extension of the random forest theory [4], by presenting an algorithm for computing the estimated distribution of the variable of interest, in our case the wind speed. The algorithm consists of an iterative process which splits the training data and every split minimizes the sum of the variance of the response variable. One of the disadvantages of this method is that the process of growing trees might lead to overfitting as mentioned in [22]. However, [39] suggested to solve this problem by tuning the number of trees.

Different predictors can be used in the implementation of the QRF (see for example [39] and [34]); here we consider two cases. For the first one the only predictors are the nine wind speed forecasts from the WRF model. In the second case this set is extended by the mean, standard deviation, minimum and maximum of some variables presented in Table 1 (U10, V10, T2, PSFC, and RH) in addition to the orographic variance (VAR), land use (LU), HGT, and the observed altitude (Alt_st), forming a total of 24 covariates. In particular, our implementation is based on the R package `quantregForest`.

The QRF model also allows to determine the importance of the predictor p_j by the random permutation method introduced in the context of random forests by [4]. The importance is computed by the mean CRPS of the difference between the forecast F conditional to the permuted predictor and the unpermuted features, i.e.

$$\text{Imp}(p_j) = \frac{1}{ST} \sum_{s=1}^S \sum_{t=1}^T (\text{CRPS}(F | X_{s,t}^{p_j}, y_{s,t}) - \text{CRPS}(F | X_{s,t}, y_{s,t})), \quad (3.2)$$

where $X_{s,t}^{p_j}$ denotes the vector of predictors at time t and station s for the permuted predictor (see [34] for more details). The higher the value of $\text{Imp}(p_j)$, the more important the predictor p_j .

3.4. Spatial selection of training data

For selecting the geographical composition of the training data for post-processing methods, [41] defines the local and regional approaches. By regional or global we mean that forecast/observation pairs of all stations from the training period are used to estimate the parameters of a given parametric model or perform a non-parametric calibration, while the local approach uses only the information of the

observation site at hand. Although the local approach in general results in better predictive performance, it requires longer training periods to avoid numerical stability issues [21]. Hence, we focus on regional estimation and on the novel semi-local approach proposed by [21]. Semi-local modeling takes the advantages of regional and local forecasting by clustering the observation stations based on climatological characteristics and the distribution of forecast errors of the training data and performing a global estimation within each cluster. Clustering is performed with the help of a k -means algorithm [14] and clusters may vary as the training window slides.

4. Results

In order to exclude the effect of natural daily variation in wind speed, calibration approaches described in Sections 3.1 and 3.3 were run separately for each forecast hour using an optimal training period length and both regional and semi-local approaches.

4.1. Selection of the training data

Selection of an appropriate set of training data is necessary for successful calibration. This selection procedure includes the choice of the length of the rolling training period and the geographical composition of the training set.

The optimal length of the training period is obtained by verifying the forecasts against observations for different training periods with the help of various scoring rules [12]. We investigated the mean CRPS and nominal coverage of the regional EMOS predictive distribution for the time period from 29 September 2017 to 30 January 2018 separately at each forecast hour using training periods of length 55, 60, 65, . . . , 120 days. Based on the corresponding figures of the mean CRPS and nominal coverage plotted against the training period length (not shown) we decided to choose a training period of length 65 days for calibrating wind speed forecasts of the WRF simulations. This length of the training period leaves 179 calendar days between 5 August 2017 and 30 January 2018 for forecast verification.

As mentioned in Section 3.4, EMOS and QRF modeling were performed using regional and clustering-based semi-local training. In the latter approach stations were grouped into 3 clusters using 24 features, where half of the features were obtained as equidistant quantiles of the climatological CDF, whereas the other half as equidistant quantiles of the empirical CDF of the forecast error over the training period [21]. Note that each verification day and forecast hour had an individual clustering of the 37 monitoring stations.

4.2. Comparison of the post-processed forecasts

EMOS and QRF calibration of WRF ensemble forecasts is performed using the optimal 65 day rolling training period and regional and semi-local approaches to

spatial selection of training data. In what follows EMOS_C and QRF_C will denote the semi-local EMOS and QRF methods, respectively, in order to distinguish them from the corresponding regional approaches. Note that EMOS model (3.1) has 12 parameters to be estimated from the training data.

Further, as mentioned in Section 3.3, QRF method is implemented in two different ways. In the first case, referred to as QRF, we use just the nine wind speed forecasts from the WRF model, whereas in the second case (QRF_M), this set is extended by several other variables (see Section 3.3) resulting in a total of 24 covariates. Both cases were tested with different arguments and we decided to make use of the model with 300 trees and a minimum size of 5 for terminal leaves, since these arguments provided smaller scores. Further, the implementations of QRF and QRF_M differ from each other in the number of variables randomly sampled as candidates at each split; one for QRF and three for QRF_M were the best options.

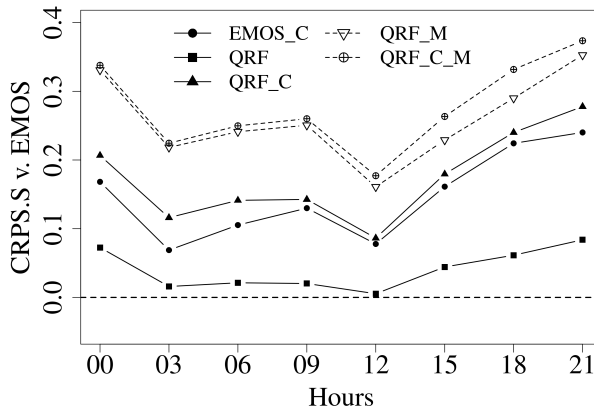


Figure 4. Mean CRPS vs. EMOS by hours for all stations.

Consider first Figure 4 showing the CRPS.S values with respect to the regional EMOS approach as function of the forecast hour. The use of semi-local estimation in EMOS modeling improves the calibration at each hour and the same applies for QRF modeling. QRF models perform slightly better than the corresponding EMOS approaches and the best QRF forecasts are obtained by adding other features as predictors to the regression model (QRF_M and QRF_C_M). Although the (QRF_M and QRF_C_M) provided better predictions than all the other methods (see Figure 4), the results could be further improved by adding new covariates, for example using the wind speed forecasts instead of the U10 and V10 components. Note that the ranking of the different methods is completely consistent, the different graphs do not cross.

Table 2. Overall scores for the different models computed in the study.

Models	CRPS	MAE	RMSE	Coverage
Ensemble	1.1715	1.4470	1.9949	43.95
EMOS	0.6078	0.8333	1.2443	82.12
EMOS_C	0.5108	0.7121	1.0361	80.29
QRF	0.5794	0.7968	1.1827	89.15
QRF_C	0.4939	0.6867	0.9817	88.18
QRF_M	0.4441	0.6143	0.9021	90.69
QRF_C_M	0.4318	0.5992	0.8781	89.65

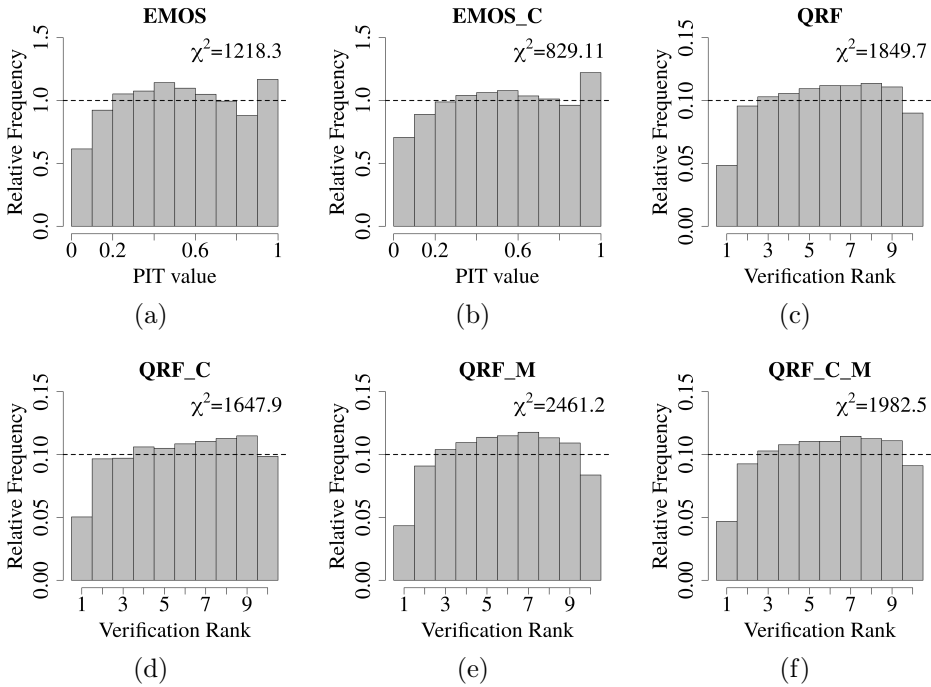


Figure 5. PIT histograms of post-processed forecasts (a) EMOS and (b) EMOS-C, and verification rank histogram of: (c) QRF, (d) QRF-C, (e) QRF-M, (f) QRF-C-M.

A similar ranking of the post-processing methods can be derived from the overall scores of Table 2. All post-processing approaches outperform the raw WRF ensemble in all scores with a wide margin, and the lowest CRPS, MAE and RMSE belong to QRF_M and QRF_C_M combined with slightly high coverage values.

Finally, as mentioned in Section 3.2, PIT and verification rank histograms allow us to visualize the improvement in calibration, and the goodness of fit to the cor-

responding uniform distribution can be quantified by the value of the χ^2 statistic (the smaller the better). According to Figure 5, all post-processing methods can successfully correct the underdispersion of the raw WRF ensemble forecast (see Figure 3a) turning it into a slight overdispersion indicated by the hump shape of the histograms. However, none of PIT histograms looks close to uniformity, probably due to the small sample size [40]. EMOS forecast are slightly biased and from the competing QRF approaches QRF-C seems to have the best calibration with $\chi^2 = 1647.9$.

4.3. Importance features results

Additionally to the comparison of the post-processing methods, the QRF model allows to determine the importance of each predictor by considering the CRPS as verification score using equation (3.2). Figure 6 shows that the observed altitude (Alt-st) and the terrain height from WRF model are the most important variables in the QRF-M. The zonal near-surface wind component (U10) seems to be more important than the meridional wind component (V10), and the surface pressure (PSFC), orographic variance (VAR) and the deviation of surface temperature (T2) are also included in the first ten most important features.

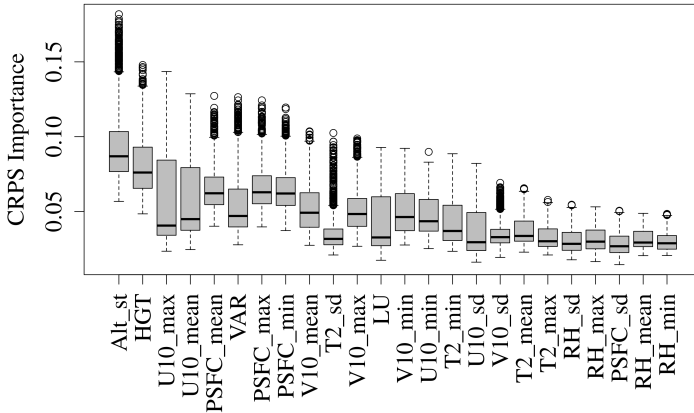


Figure 6. Importance of the forecasts considering the CRPS score.

5. Conclusions

In this work some parametric and non-parametric post-processing methods for calibrating 9-member 3 hourly WRF wind speed ensemble forecasts are investigated. The WRF ensemble forecasts were generated by different planetary boundary layers and land-surface model parametrizations in order to model wind speed at 37 monitoring stations around Valparaíso and Santiago city for the period between 1 June 2017 and 30 January 2018. In order to choose the optimal training pe-

riod length (65 days in this study) a regional EMOS model has been tested with training periods of different lengths. EMOS and QRF modeling is performed both regionally and using a clustering-based semi-local approach, the different forecast hours are treated separately in order to exclude the natural daily variation in wind speed.

Compared with raw WRF ensemble forecasts, all post-processing approaches result in a substantial improvement in calibration of probabilistic and accuracy of point forecasts. From the competing approaches to calibration, the semi-local QRF model considering other weather variables from WRF simulations exhibits the best overall predictive performance.

The importance of the predictors for QRF using a permutation method is also investigated, where from the additional covariates the altitude of the station occurs to be the most important. These results are crucial in choosing the parametrization for the WRF model in order to improve wind predictions in Chile.

As a next step we are planning to explore other machine learning methods in addition to different parametrizations. Further, it would be very interesting to evaluate the performance of EMOS models with additional predictors using the boosting approach proposed by [27].

Acknowledgments. Mailiu Díaz is grateful for the support of the National Commission for Scientific and Technological Research (CONICYT) of Chile under Grant No. 21150227. Orietta Nicolis and Julio César Marín are partially supported by the Interdisciplinary Center of Atmospheric and Astro-Statistical Studies. Powered@NLHPC: This research was partially supported by the supercomputing infrastructure of the National Laboratory for High Performing Computer (NLHPC) (ECM-02). Sándor Baran was supported by the National Research, Development and Innovation Office under Grant No. NN125679.

References

- [1] S. BARAN: *Probabilistic wind speed forecasting using Bayesian model averaging with truncated normal components*, Computational Statistics and Data Analysis 75 (2014), pp. 227–238, DOI: <https://doi.org/10.1016/j.csda.2014.02.013>.
- [2] S. BARAN, S. LERCH: *Log-normal distribution based EMOS models for probabilistic wind speed forecasting*, Quarterly Journal of the Royal Meteorological Society 141 (2015), pp. 2289–2299, DOI: <https://doi.org/10.1002/qj.2521>.
- [3] D. BOLTON: *The computation of equivalent potential temperature*, Monthly Weather Review 108 (1980), pp. 1046–1053, DOI: [https://doi.org/10.1175/1520-0493\(1980\)108<1046:TCOEPT>2.0.CO;2](https://doi.org/10.1175/1520-0493(1980)108<1046:TCOEPT>2.0.CO;2).
- [4] L. BREIMAN: *Random Forests*, Machine Learning 45.1 (2001), pp. 5–32, ISSN: 1573-0565, DOI: <https://doi.org/10.1023/A:1010933404324>.

- [5] F. CHEN, J. DUDHIA: *Coupling an Advanced Land Surface-Hydrology Model with the Penn State-NCAR MM5 Modeling System. Part I: Model Implementation and Sensitivity*, Monthly Weather Review 129.4 (2001), pp. 569–585, DOI: [https://doi.org/10.1175/1520-0493\(2001\)129<0569:CAALSH>2.0.CO;2](https://doi.org/10.1175/1520-0493(2001)129<0569:CAALSH>2.0.CO;2).
- [6] A. M. CÓRDOVA, J. ARÉVALO, J. C. MARÍN, D. BAUMGARDNER, G. B. RAGA, D. POZO, C. A. OCHOA, R. RONDANELLI: *On the transport of urban pollution in an Andean mountain valley*, Aerosol & Air Quality Research 16 (2016), pp. 593–605, DOI: <https://doi.org/10.4209/aaqr.2015.05.0371>.
- [7] M. DÍAZ, O. NICOLIS, J. MARÍN, S. BARÁN: *Statistical post-processing of ensemble forecasts of temperature in Santiago de Chile*, Meteorological Applications 27.1 (2019), DOI: <https://doi.org/10.1002/met.1818>.
- [8] J. DUDHIA: *A Multi-layer Soil Temperature Model for MM5. The Sixth PSU/NCAR Mesoscale Model Users' Workshop*, in: July 1996.
- [9] Y.-M. G., J. GIRONÁS, M. Caneo, R. DELGADO, R. GARREAUD: *Using the Weather Research and Forecasting (WRF) Model for Precipitation Forecasting in an Andean Region with Complex Topography*, Atmosphere 9.8 (2018), p. 304, DOI: <https://doi.org/10.3390/atmos9080304>.
- [10] T. GNEITING: *Making and evaluating point forecasts*, Journal of the American Statistical Association 106.494 (2011), pp. 746–762.
- [11] T. GNEITING, F. BALABDAOUI, A. E. RAFTERY: *Probabilistic forecasts, calibration and sharpness*, Journal of the Royal Statistical Society: Series B 69 (2007), pp. 243–268.
- [12] T. GNEITING, A. E. RAFTERY: *Strictly proper scoring rules, prediction, and estimation*, Journal of the American Statistical Association 102.477 (2007), pp. 359–378, DOI: <https://doi.org/10.1198/016214506000001437>.
- [13] T. GNEITING, A. E. RAFTERY, A. H. WESTVELD, T. GOLDMAN: *Calibrated Probabilistic Forecasting Using Ensemble Model Output Statistics and Minimum CRPS Estimation*, Monthly Weather Review 133.5 (2005), pp. 1098–1118, DOI: <https://doi.org/10.1175/MWR2904.1>.
- [14] J. A. HARTIGAN, M. A. WONG: *Algorithm AS 136: A K-means clustering algorithm*, Applied Statistics 28 (1979), pp. 100–108, DOI: <https://doi.org/10.2307/2346830>.
- [15] S.-Y. HONG, J. DUDHIA, S.-H. CHEN: *A Revised Approach to Ice Microphysical Processes for the Bulk Parameterization of Clouds and Precipitation*, Monthly Weather Review 132.1 (2004), pp. 103–120, DOI: [https://doi.org/10.1175/1520-0493\(2004\)132<0103:ARATIM>2.0.CO;2](https://doi.org/10.1175/1520-0493(2004)132<0103:ARATIM>2.0.CO;2).
- [16] S.-Y. HONG, Y. NOH, J. DUDHIA: *A New Vertical Diffusion Package with an Explicit Treatment of Entrainment Processes*, Monthly Weather Review 134.9 (2006), pp. 2318–2341, DOI: <https://doi.org/10.1175/MWR3199.1>.
- [17] M. J. IACONO, J. S. DELAMERE, E. J. MLAWER, M. W. SHEPARD, S. A. CLOUGH, W. D. COLLIN: *Radiative forcing by long-lived greenhouse gases: Calculations with the AER radiative transfer models*, Journal of Geophysical Research: Atmospheres 113.D13 (2008), DOI: <https://doi.org/10.1029/2008JD009944>.
- [18] Z. I. JANJIĆ: *The Step-Mountain Eta Coordinate Model: Further Developments of the Convection, Viscous Sublayer, and Turbulence Closure Schemes*, Monthly Weather Review 122.5 (1994), pp. 927–945, DOI: [https://doi.org/10.1175/1520-0493\(1994\)122<0927:TSMECM>2.0.CO;2](https://doi.org/10.1175/1520-0493(1994)122<0927:TSMECM>2.0.CO;2).
- [19] P. A. JIMÉNEZ, J. DUDHIA: *Improving the Representation of Resolved and Unresolved Topographic Effects on Surface Wind in the WRF Model*, Journal of Applied Meteorology and Climatology 51 (2012), pp. 300–316, DOI: <https://doi.org/10.1175/JAMC-D-11-084.1>.

- [20] J. S. KAIN: *The Kain-Fritsch Convective Parameterization: An Update*, Journal of Applied Meteorology 43.1 (2004), pp. 170–181, DOI: [https://doi.org/10.1175/1520-0450\(2004\)043<0170:TKCPAU>2.0.CO;2](https://doi.org/10.1175/1520-0450(2004)043<0170:TKCPAU>2.0.CO;2).
- [21] S. LERCH, S. BARAN: *Similarity-based semi-local estimation of EMOS models*, Journal of the Royal Statistical Society: Series C 66 (2017), pp. 29–51.
- [22] Y. LIN, Y. JEON: *Random Forests and Adaptive Nearest Neighbors*, Journal of the American Statistical Association 101 (2006), pp. 578–590.
- [23] J. C. MARÍN, D. POZO, E. MLAWER, D. TURNER, M. CURÉ: *Dynamics of local circulations in mountainous terrain during the RHUBC-II project*, Monthly Weather Review 141.10 (2013), pp. 3641–3656, DOI: <https://doi.org/10.1175/MWR-D-12-00245.1>.
- [24] C. MATTAR, D. BORVARÁN: *Offshore wind power simulation by using WRF in the central coast of Chile*, Renewable Energy 94 (2016), pp. 22–31, DOI: <https://doi.org/10.1016/j.renene.2016.03.005>.
- [25] A. MAZZEO, N. HUNEEUS, C. ORDÓÑEZ, A. ORFANOZ-CHEUQUELAF, L. MENUT, S. MAILLER, M. VALARI, H. DENIER VAN DER GON, L. GALLARDO, R. MUÑOZ, R. DONOSO, M. GALLEGUILLOS, M. OSSES, S. TOLVETT: *Impact of residential combustion and transport emissions on air pollution in Santiago during winter*, Atmospheric Environment 190 (2018), pp. 195–208, DOI: <https://doi.org/10.1016/j.atmosenv.2018.06.043>.
- [26] N. MEINSHAUSEN: *Quantile Regression Forests*, Journal of Machine Learning Research 7 (2006), pp. 983–999, ISSN: 1532-4435.
- [27] J. W. MESSNER, G. J. MAYR, A. ZEILEIS: *Nonhomogeneous boosting for predictor selection in ensemble postprocessing*, Monthly Weather Review 145.1 (2017), pp. 137–147, DOI: <https://doi.org/10.1175/MWR-D-16-0088.1>.
- [28] R. MUÑOZ, M. FALVEY, M. ARANCIBIA, V. ASTUDILLO, J. ELGUETA, M. IBARRA, C. SANTANA, C. VÁSQUEZ: *Wind energy exploration over the Atacama Desert: a numerical model-guided observational program*, Bulletin of the American Meteorological Society 99 (2018), pp. 2079–2092, DOI: <https://doi.org/10.1175/BAMS-D-17-0019.1>.
- [29] M. NAKANISHI, H. NIINO: *An Improved Mellor–Yamada Level-3 Model: Its Numerical Stability and Application to a Regional Prediction of Advection Fog*, Boundary-Layer Meteorology 119.2 (2006), pp. 397–407, DOI: <https://doi.org/10.1007/s10546-005-9030-8>.
- [30] P. PINSON, R. HAGEDORN: *Verification of the ECMWF ensemble forecasts of wind speed against analyses and observations*, Meteorological Applications 19 (2012), pp. 484–500.
- [31] J. E. PLEIM, A. XIU: *Development of a Land Surface Model. Part II: Data Assimilation*, Journal of Applied Meteorology 42.12 (2003), pp. 1811–1822, DOI: [https://doi.org/10.1175/1520-0450\(2003\)042<1811:DOALSM>2.0.CO;2](https://doi.org/10.1175/1520-0450(2003)042<1811:DOALSM>2.0.CO;2).
- [32] J. G. POWERS, J. G. KLEMP, W. S. SKAMAROCK, C. A. DAVIS, J. DUDHIA, D. O. GILL, J. L. COEN, D. J. GOCHIS, R. AHMADOV, S. E. PECKHAM, G. A. GRELL, J. MICHALAKES, S. TRAHAN, S. G. BENJAMIN, C. R. ALEXANDER, G. J. DIMEGO, W. WANG, C. S. SCHWARTZ, G. S. ROMINE, Z. LIU, C. SNYDER, F. CHEN, M. J. BARLAGE, W. YU, M. G. DUDA: *The Weather Research and Forecasting Model: overview, system efforts, and future directions*. Bulletin of the American Meteorological Society 98 (2017), pp. 1717–1737.
- [33] D. POZO, J. C. MARÍN, G. B. RAGA, J. AREVALO, D. BAUMGARDNER, A. M. CÓRDOVA, J. MORA: *Synoptic and local circulations associated with events of high particulate pollution in Valparaíso, Chile*, Atmospheric Environment 196 (2018), pp. 164–178, DOI: <https://doi.org/10.1016/j.atmosenv.2018.10.006>.

- [34] S. RASP, S. LERCH: *Neural networks for post-processing ensemble weather forecasts*, Monthly Weather Review 146.11 (2018), pp. 3885–3900, DOI: <https://doi.org/10.1175/MWR-D-18-0187.1>.
- [35] J. J. RUIZ, C. SAULO, J. NOGUÉS-PAEGLE: *WRF model sensitivity to choice of parameterization over South America: Validation against surface variables*, Monthly Weather Review 138.8 (2010), pp. 3342–3355, DOI: <https://doi.org/10.1175/2010MWR3358.1>.
- [36] M. SAIDE P. E. ADN MENA-CARRASCO, S. TOLVETT, P. HERNANDEZ, G. CARMICHAEL: *Air quality forecasting for winter-time PM 2.5 episodes occurring in multiple cities in central and southern Chile*, Journal of Geophysical Research: Atmospheres 121.1 (2016), pp. 558–575, DOI: <https://doi.org/10.1002/2015JD023949>.
- [37] W. SKAMAROCK, J. KLEMP, J. DUDHIA, D. GILL, D. BARKER, W. WANG, J. POWERS: *A Description of the Advanced Research WRF Version 3*, 27 (Jan. 2008), pp. 3–27.
- [38] J. M. SLOUGHTER, T. GNEITING, A. E. RAFTERY: *Probabilistic wind speed forecasting using ensembles and Bayesian model averaging*, Journal of the American Statistical Association 105 (2010), pp. 25–35, DOI: <https://doi.org/10.1198/jasa.2009.ap08615>.
- [39] M. TAILLARDAT, O. MESTRE, M. ZAMO, P. NAVEAU: *Calibrated Ensemble Forecasts Using Quantile Regression Forests and Ensemble Model Output Statistics*, Monthly Weather Review 144.6 (2016), pp. 2375–2393, DOI: <https://doi.org/10.1175/MWR-D-15-0260.1>.
- [40] T. L. THORARINSDOTTIR, N. SCHUHEN: *Chapter 6 - Verification: Assessment of Calibration and Accuracy*, in: Statistical Postprocessing of Ensemble Forecasts, Elsevier, 2018, pp. 155–186, DOI: <https://doi.org/10.1016/B978-0-12-812372-0.00006-6>.
- [41] T. L. THORARINSDOTTIR, T. GNEITING: *Probabilistic forecasts of wind speed: ensemble model output statistics by using heteroscedastic censored regression*, Journal of the Royal Statistical Society Series A 173.2 (2010), pp. 371–388, DOI: <https://doi.org/10.1111/j.1467-985X.2009.00616.x>.
- [42] D. S. WILKS: *Statistical Methods in the Atmospheric Sciences*, Amsterdam: 4th ed., Elsevier, 2019.

Trees classification based on Fourier coefficients of the sapflow density flux*

Dmitry Efrosinin^{ab}, Irina Kochetkova^{bc}, Natalia Stepanova^d,
Alexey Yaroslavtsev^{be}, Konstantin Samouylov^{bc},
Riccardo Valentini^{fb}

^aJohannes Kepler University Linz, Austria
dmitry.efrosinin@jku.at

^bPeoples' Friendship University of Russia (RUDN University), Russia
gudkova-ia@rudn.ru, yaroslavtsev-am@rudn.ru, samuylov-ke@rudn.ru

^cInstitute of Informatics Problems, Federal Research Center
“Computer Science and Control” of RAS, Russia

^dV.A. Trapeznikov Institute of Control Sciences of RAS, Russia
natalia0410@rambler.ru

^eLAMP, Russian Timiryazev State Agrarian University, Russia,
^fTuscia University, Viterbo, Italy
rik@unitus.it

Submitted: December 22, 2020

Accepted: March 8, 2021

Published online: May 18, 2021

Abstract

In this paper we study the possibility to use the artificial neural networks for trees classification based on real and approximated values of the sap flow density flux describing water transport in trees. The data sets were generated by means of a new tree monitoring system TreeTalker[©]. The Fourier series-based model is used for fitting the data sets with periodic patterns. The multivariate regression model defines the functional dependencies between sap flow density and temperature time series. The paper shows that Fourier

*The work was supported by the Russian Science Foundation, project 19-77-30012 (recipients I. Kochetkova, A. Yaroslavtsev, R. Valentini). This paper has been supported by the RUDN University Strategic Academic Leadership Program (recipients D. Efrosinin, K. Samouylov).

coefficients can be successfully used as elements of the feature vectors required to solve different classification problems. Here we train multilayer neural networks to classify the trees according to different types of classes. The quality of the developed model for prediction and classification is verified by numerous numerical examples.

Keywords: TreeTalker monitoring system, Fourier coefficients, neural network, classification of trees

AMS Subject Classification: 65C60, 62M10

1. Introduction

In the last decade, monitoring systems, which can be treated as a part of smart technologies and are used for generating large amounts of data sets from a network of sensors, have evolved rapidly. This work is a continuation of a previous survey related to a new sensor tree monitoring system TreeTalker[©] (TT) [12]. The TT is a system used for real-time ecological forest plantation monitoring made with the concept of the Internet of Things (IoT). It is responsible for collecting data from all sensors, which are fixed to trees, and transforming the analog signal to meaningful variables. With this system a database was created which is expected to be published shortly. It includes, among other things like temperature, humidity of the air and wood, spectral characteristics of the canopy, radial growth of the trunk, and data from accelerometer about 3D position of trunk, a large amount of information on the sap density flux describing water transport process in different tree species that also differ in age, health status, metric characteristics, etc.

We report in this paper our first experiments carried out on data sets extracted by the TT monitoring system as well as on the estimated values of the density flux and dedicated to trees classification. Classification is a very common use case of a machine learning. Artificial neural networks [6, 9] (NN) are parts of a supervised machine learning which are most popular in different problems of data classification, pattern recognition, regression, clustering, time series forecasting. Some of applications of the NN to the real data sets can be found, e.g. in [1, 4]. We study the possibility to use NN for classifying the trees of different species within the same age group and visual-tree-assessment (VTA) score, the same species but with different age groups and/or VTA scores as well as with trunk diameters. As classification features we use Fourier coefficients obtained by fitting the truncated Fourier series to the sapflow density flux data sets. This paper also shows that in addition to the Fourier coefficients obtained directly for the sap density function, coefficients estimated using multivariate regression on air temperature data can also be successfully used to classify trees. Thus, it is possible to simulate the sap flow process of a particular tree species. In the long term, this approach, which incorporates data generated by the TT with the proposed Fourier coefficient estimation method, can be used to trace ecological anomalies in the health status of an individual tree or entire forest area.

The rest of the paper is organized as follows. In Section 2 we describe the data

sets and methods. In Sections 3 we study the problem of trees classification which is based on Fourier coefficients estimated in observable time period. Final remarks are given in Section 4.

2. Data sets and methods

In most cases, data is collected hourly, but in Russian conditions, 1.5 hours are recommended. The performance of TreeTalkers is being tested at several sites all over Eurasia continent from Spain to China, with a wide variety of tree species, climate, topography, and land use with multiple tests of device reliability in terms of sensors operational limits of the sensors, data transmission, and battery effectiveness. In May 2019, 60 TT sensors were installed on different species of trees growing in different areas, belonging to different age groups with varying VTA scores summarized in Table 1. Data from all devices was collected till the end of November 2019 and stored on a remote web server. All data for basic variables was 3 sigma filtered, runaways were eliminated and gap filled with linear 4D interpolation for gaps smaller than 4 measurements. Sapwood area data was combined with TT data, and individual tree sap flux was calculated utilizing R software. While there are several papers where modern modeling techniques implemented for predicting evapotranspiration of planted areas with environmental data [7, 11, 14], there are very limited amount of papers about individual tree sap flow modeling [10]. Taking into account growing trend of IoT devices used in environmental monitoring [3, 13] modeling of one of the main physiological tree characteristics can be of great interest. This metric is in many cases a very contrasting reflection of the degree to which individual qualitative factors influence the state of the tree. As an example, consider Figure 1, which shows the sap density functions for two tree species: *Salix alba* and *Acer platanoides*. In first case we have three classes (III,2), (IV,2) and (IV,3), where trees differ in age group and VTA factor. In second case trees belong to the same age group but differ in VTA value according to four classes, (VI,1), (VI,2), (VI,3) and (VI,4). As can be seen, the functions $y_{flux,t}$ have a significant difference, depending on the respective class. Thus, it can be expected that this characteristic can be successfully applied for classification.

The data sets for the the air temperature (*tair*) and the sapflow density flux (*flux*) are represented respectively in form of time series $\vec{y}_{tair} = (y_{tair,1}, y_{tair,2}, \dots)$ and $\vec{y}_{flux} = (y_{flux,1}, y_{flux,2}, \dots)$ and with time-ordered sequence of observations. These time series are characterized by fluctuations which exhibit a periodic nature with a cycle length T . Each cycle includes mostly $N = 16$ measurements that are made at equally spaced time intervals $\Delta t = 1.5h$. The total time within a cycle is then $T = N\Delta t = 24h$. Since the fluctuations may have different amplitudes and shapes within each period, the data sets can not be treated as pure periodic ones, i.e. in general case $y_{tair,t_0} \neq y_{tair,t_0+kT}$ and $y_{flux,t_0} \neq y_{flux,t_0+kT}$ for all $k \in \mathbb{N}$. The data preprocessing step includes the denoising by locally data smoothing. We use for that a low pass filter which passes signals with a frequency lower than a selected cutoff frequency $\omega_c = 0.9$. Smaller values of ω_c result in greater smoothing.

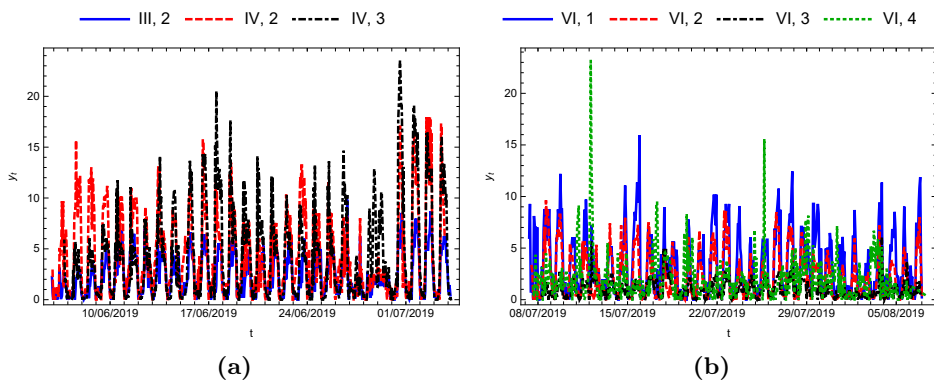


Figure 1. Sapflow density flux y_t *Salix alba* (a) and *Acer platanoides* (b).

Table 1. Key elements of the database.

Sorts	Area	Age	VTA score
<i>Acer fraxinifolium</i>	MS-7	IV,IV	2,3
<i>Acer platanoides</i>	OL-1,OL-2,OL-3,MS-6	III,IV,VI	1,2,3,4
<i>Aesculus flava</i>	OL-3	VI	3
<i>Betula pendula</i>	OL-1,MS-4,MS-6	IV,VI	1,2,3
<i>Carpinus betulus</i>	OL-4	IV	2
<i>Fraxinus excelsior</i>	OL-3	VI	3
<i>Fraxinus ornus</i>	OL-3	VI	2
<i>Juglans mandshurica</i>	OL-3	VI	3
<i>Junglas cinerea</i>	OL-3	VI	2
<i>Larix decidua</i>	MS-6	IV,VI	2
<i>Larix sibirica</i>	OL-2,OL-3,MS-6	V,VI	2,3,4
<i>Malus domestica</i>	OL-3	IV	3
<i>Picea abies</i>	OL-1,MS-4	IV,V,VI	1,2,3
<i>Pinus sylvestris</i>	OL-1	III,IV,VI	2,3
<i>Populus nigra</i>	OL-3,MS-7	VI	2,3
<i>Populus tremula</i>	OL-1	III,VI	1,2
<i>Prunus avium</i>	OL-3	VI	2
<i>Pyrus commutis</i>	OL-3	VI	2
<i>Quercus Rrobur</i>	OL-2,OL-3,MS-4	VI	2,3,4
<i>Robinia pseudoacacia</i>	OL-3	IV	2
<i>Salix alba</i>	OL-3,MS-5	III,IV,VI	2,3
<i>Tilia cordata</i>	OL-1,OL-2,OL-3,MS-4,MS-6	III,IV,VI	1,2,3,4

A truncated Fourier series can be used to find approximations for periodic functions of the air temperature $f_{\text{tair}}(t)$ and the sapflow density $f_{\text{flux}}(t)$ with a fundamental

period T that passes through all of the points,

$$\begin{aligned} f_{\text{tair}}(t) &\approx a_0 + \sum_{n=1}^m \left[a_n \cos\left(\frac{2\pi nt}{T}\right) + b_n \sin\left(\frac{2\pi nt}{T}\right) \right], \\ f_{\text{flux}}(t) &\approx \alpha_0 + \sum_{n=1}^m \left[\alpha_n \cos\left(\frac{2\pi nt}{T}\right) + \beta_n \sin\left(\frac{2\pi nt}{T}\right) \right]. \end{aligned} \quad (2.1)$$

The coefficients a_n , b_n and α_n , β_n can not be explicitly derived since the functions $f_{\text{tair}}(t)$ and $f_{\text{flux}}(t)$ are not available in explicit form and hence they must be estimated. We have only data $\vec{y}_{\text{tair}} = (y_{\text{tair},1}, y_{\text{tair},2}, \dots, y_{\text{tair},n_s})'$ and $\vec{y}_{\text{flux}} = (y_{\text{flux},1}, y_{\text{flux},2}, \dots, y_{\text{flux},n_s})'$ generated by the sensors. The known periodic patterns of the approximated functions $f_{\text{tair}}(t)$ and $f_{\text{flux}}(t)$ are expressed through vectors of parameters $\vec{a} = (a_0, a_1, \dots, a_m, b_1, \dots, b_m)'$ and $\vec{\alpha} = (\alpha_0, \alpha_1, \dots, \alpha_m, \beta_1, \dots, \beta_m)'$. These parameters are estimated using the method of the linear least squares

$$\sum_{t=(i-1)T}^{iT} (y_{\text{tair},t} - f_{\text{tair}}(t))^2 \Rightarrow \min_{\vec{a}}, \quad \sum_{t=(i-1)T}^{iT} (y_{\text{flux},t} - f_{\text{flux}}(t))^2 \Rightarrow \min_{\vec{\alpha}}, \quad 1 \leq i \leq n_p,$$

where $n_p = \frac{n_s}{N}$ is a number of cycles of length T within the observations with a total sample size n_s .

For trees classification the feature vectors $(\alpha_{i,0}, \alpha_{i,1}, \dots, \alpha_{i,m}, \beta_{i,1}, \dots, \beta_{i,m})$ of the Fourier coefficients (2.1) in an observable period $1 \leq i \leq n_p$ are used. Recall that in the previous paper [5] we presented a method for predicting the density flux during the day based on data on air temperature during the observed cycle. For this purpose, Fourier series and a multivariate regression model were used, establishing the functional relationship between the respective Fourier coefficients for temperature data sets and density flux values,

$$\begin{aligned} \alpha_{i,n} &= \theta_{0,n} + \theta_{1,n} a_{i,0} + \sum_{j=1}^m [\theta_{j+1,n} a_{i,j} + \theta_{m+j+1,n} b_{i,j}], \quad 0 \leq n \leq m, \\ \beta_{i,n} &= \theta_{0,n+m} + \theta_{1,n+m} a_{i,0} + \sum_{j=1}^m [\theta_{j+1,n+m} a_{i,j} + \theta_{m+j+1,n+m} b_{i,j}], \quad 1 \leq n \leq m, \end{aligned} \quad (2.2)$$

where $1 \leq i \leq n_p$, $\vec{\theta}_k = (\theta_{0,k}, \theta_{1,k}, \dots, \theta_{2m+1,k})'$, $0 \leq k \leq 2m$, denotes the vector of parameters of the multidimensional regression model. We discuss here the results of experiments carried out on data sets extracted by the TT monitoring system as well as on the estimated values of the density flux and dedicated to trees classification. We study the possibility to use artificial neural networks to classify the trees of the same species but with different age groups and visual-tree-assessment (VTA) scores. As classification features we use a predicted Fourier coefficients of the sap flow density flux approximation function. In the long term, this approach which incorporates data generated by the TT with the proposed Fourier coefficient estimation method can be used to determine the anomalous state of a tree or generally monitor forest ecology.

As was mentioned above, as features for trees classification we use the sets of vectors S , consisting of eleven original coefficients of the truncated Fourier series fitted to the density flux function $y_{\text{flux},t}$. Moreover, the classifier will be applied also to the sets \hat{S} for the predicted coefficients of the function $\hat{y}_{\text{flux},t}$ by using the multiple regression (2.2) for the Fourier coefficients of the air temperature data $y_{\text{air},t}$. The data sets for the classification problem were prepared in form of the set of mappings,

$$S = \{(\alpha_{i,0}, \alpha_{i,1}, \dots, \alpha_{i,m}, \beta_{i,1}, \dots, \beta_{i,m}) \rightarrow \text{Class } N : 1 \leq i \leq n_p\},$$

$$\hat{S} = \{(\hat{\alpha}_{i,0}, \hat{\alpha}_{i,1}, \dots, \hat{\alpha}_{i,m}, \hat{\beta}_{i,1}, \dots, \hat{\beta}_{i,m}) \rightarrow \text{Class } N : 1 \leq i \leq n_p\},$$

where $m = 5$ and n_p is a number of observable periods. 70% of samples S and \hat{S} is referred to as training data and the rest – as validation data. The data were chosen so that the sample in each class was more or less balanced, i.e. the sample size in each class did not differ significantly. The multilayer neural network is used for the data classification. It can be formally defined as a function $f : \vec{\alpha} \rightarrow \vec{y}$, which maps an input vector $\vec{\alpha}$ of dimension $2m + 1$ to an estimate output $\vec{y} \in \mathbb{R}^{N_c}$ of the class number $N = 1, \dots, N_c$. The network is decomposed into 6 layers as illustrated in Figure 2, each of which represents a different function mapping vectors to vectors. The successive layers are: a linear layer with an output vector of size k , a nonlinear elementwise activation layer, other three linear layers with output vectors of size k , and a nonlinear normalization layer.

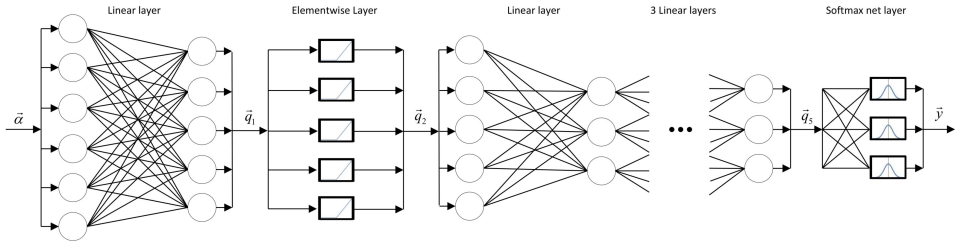


Figure 2. Architecture of the neural network.

The first layer is an affine transformation

$$\vec{q}_1 = W_1 \vec{\alpha} + \vec{b}_1,$$

where $\vec{q}_1 \in \mathbb{R}^{2m+1}$ is the output vector, $W \in \mathbb{R}^{2m+1 \times k=30}$ is the weight matrix, $\vec{b}_1 \in \mathbb{R}^{2m+1}$ is the bias vector. The rows in W_1 are interpreted as features that are relevant for differentiating between corresponding classes. Consequently, $W_1 \vec{\alpha}$ is a projection of the input $\vec{\alpha}$ onto these features. The second layer is an elementwise activation layer which is defined by the nonlinear function $\vec{q}_2 = \max(0, \vec{q}_1)$ setting negative entries of q_1 to zero and uses only positive entries. The next three layers are another affine transformations,

$$\vec{q}_i = W_i \vec{q}_{i-1} + \vec{b}_i,$$

where $\vec{q}_i \in \mathbb{R}^k$, $W_i \in \mathbb{R}^{k \times k}$, and $b_i \in \mathbb{R}^k$, $i = 3, 4, 5$. The last layer is the normalization layer $\vec{y} = \text{softmax}(\vec{q}_5)$, which componentwise is of the form

$$y_N = \frac{e^{q_{5N}}}{\sum_N e^{q_{5N}}}, N = 1, \dots, N_c.$$

The last layer normalizes the output vector \vec{y} with the aim to get the values between 0 and 1. The output \vec{y} can be treated as a probability distribution vector, where the N th element y_N represents the likelihood that \vec{a} belongs to class N .

The neural networks in our experiments are trained by the ADAM (adaptive moment estimation method) [8] which is a modification of stochastic gradient descent (SGD). The neural network toolbox in *Mathematica*[®] of the Wolfram Research is used. We verify the classifier which should be accurate enough to be used to predict new output from verification data. The algorithm was ran many times on samples and networks with different sizes. In all cases the results were quite positive and indicate the potential of machine learning methodology for trees classification problem based on the estimated Fourier coefficients.

To follow the classification progress, we summarize the results in form of a confusion matrix evaluated on the trained and verification data. with these matrices it is possible to observe the relations between the classifier outputs and the true ones. Each row of these matrices represents the instances in a predicted value while each column represents the instances in an actual value. Different statistical measures of the performance of a binary classification, such as the overall accuracy (ACC), sensitivity (true positive rate – TPR), specificity (true negative rate – TNR) as well as F-1 Scores which is the harmonic mean of precision and sensitivity. For more details about these measures, refer to [2]. Note that in multi-class classification problem we calculate the F-1 Score per class in a one-vs-rest manner, i.e. we estimate successful occurrence of the class as if there are individual classifiers for each class.

3. Experiments

Five main examples are discussed in this section.

Example 3.1. In this example, we test the feasibility of using the data to classify tree varieties within the same age group and VTA score. Data for four tree varieties such as *Acer platanoides*, *Betula pendula*, *Salix albe* and *Tilla cordata* in the age group IV and with the VTA score 2 were selected for 4-class classification problem as is shown in Table 2.

The Figure 3 shows the confusion matrices evaluated using data set S and \hat{S} for real and estimated Fourier coefficients of the data flux density function $y_{\text{flux},t}$. Obviously, the matrices are diagonally dominant and the frequencies of correctly recognized classes are almost identical. Four statistical quantities described above, which are used to represent some aspect of a classification quality, are summarized in Table 3. The quality of the classification is quite high, the overall accuracy is over

than 85%. The quality metrics per class take also the high values. Moreover, the classification of trees according to the estimated Fourier coefficients exhibits slightly reduced values for quality parameters, but the difference is quite insignificant.

Table 2. Classes within the same age group IV and VTA score 2.

Class N	Sort	Age group	VTA score
1	Acer platanoides	IV	2
2	Betula pendula	IV	2
3	Salix albe	IV	2
4	Tilia cordata	IV	2

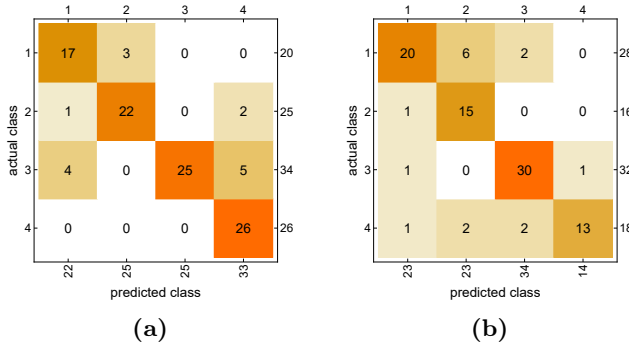


Figure 3. Confusion matrices for sorts classification based on S (a) and \hat{S} (b) data sets.

Table 3. Classification performance.

Data \ Metric	ACC	TPR	TNR	F-1 Scores
S	0.8571	1 \rightarrow 0.8500	1 \rightarrow 0.9412	1 \rightarrow 0.8095
		2 \rightarrow 0.8800	2 \rightarrow 0.9625	2 \rightarrow 0.8800
		3 \rightarrow 0.7353	3 \rightarrow 1.000	3 \rightarrow 0.8474
		4 \rightarrow 1.000	4 \rightarrow 0.9114	4 \rightarrow 0.8814
\hat{S}	0.8298	1 \rightarrow 0.7143	1 \rightarrow 0.9955	1 \rightarrow 0.7843
		2 \rightarrow 0.9375	2 \rightarrow 0.8974	2 \rightarrow 0.7692
		3 \rightarrow 0.9375	3 \rightarrow 0.9355	3 \rightarrow 0.9091
		4 \rightarrow 0.7222	4 \rightarrow 0.9864	4 \rightarrow 0.8125

Based on the available samples, it can thus be stated that the sap flow process varies considerably among the different tree varieties. We believe that by obtaining an appropriate trained neural network for each tree variety of a certain age group and

VTA score, it is feasible to recognize anomalies in the growth process of a particular tree, which in turn will make it possible to produce an environmental health map of forest plantations in urban parks or large forest areas outside of cities.

The experiments carried out comparing different tree varieties in terms of sap density values show the possibility to use not only the real values obtained directly by the TT monitoring system, but also their estimates obtained by multivariate linear regression as a functional relationship between air temperature and sap density values. This allows considerable savings in the purchase and installation of a large number of sensors, as a sensor network of a limited number of devices installed on different types of trees will be sufficient to cover large areas.

Example 3.2. Consider data sets with n_p observable periods for *Salix alba*. We divide the data set into three subgroups according to Table 4.

Table 4. Classes of *Salix alba*.

Class N	Age group	VTA score
1	IV	2
2	IV	3
3	III	2

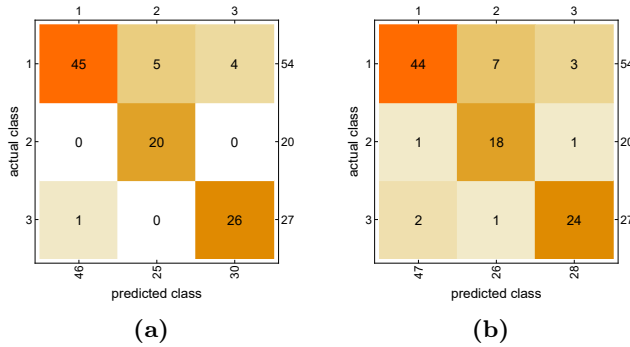


Figure 4. Confusion matrices for classification of *Salix alba* based on $y_{flux,t}$ (a) and $\hat{y}_{flux,t}$ (b).

The trees of this species can belong to different age groups and have different VTA scores. Figure 4 illustrates two confusion matrices which are obviously diagonally dominant. As we can see, factors such as age group and VTA have a significant influence on the values of the density flux function. As can be seen in Table 5, the classification accuracy reaches more than 90% and the overall accuracy is almost indistinguishable from the qualitative characteristics for each class. The results of experiments were quite positive and indicate the potential of machine learning methodology for trees classification problem based on the Fourier coefficients for the fitted density flux data.

Table 5. Classification performance.

Data \ Metric	ACC	TPR	TNR	F-1 Scores
<i>Salix albe</i> , S	0.9009	1 \rightarrow 0.8333 2 \rightarrow 1.000 3 \rightarrow 0.9629	1 \rightarrow 0.9787 2 \rightarrow 0.9382 3 \rightarrow 0.9459	1 \rightarrow 0.9000 2 \rightarrow 0.8889 3 \rightarrow 0.9123
<i>Salix albe</i> , \hat{S}	0.8514	1 \rightarrow 0.8148 2 \rightarrow 0.9000 3 \rightarrow 0.9231	1 \rightarrow 0.9348 2 \rightarrow 0.9125 3 \rightarrow 0.9459	1 \rightarrow 0.8713 2 \rightarrow 0.8000 3 \rightarrow 0.8889

Example 3.3. Next we study the possibility to classify the trees of the same species according to different age groups but with the equal VTA scores. The presented experiment includes the gathered data for *Tilia cordata* and the task is to provide a classification according to the classes in Table 6.

Table 6. Classes of *Tilia cordata*.

Class N	Age group	VTA score
1	III	3
2	IV	3
3	VI	3

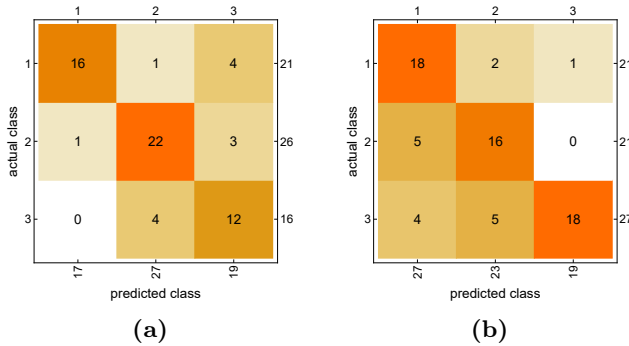


Figure 5. Confusion matrices for classification of *Tilia cordata* based on real $y_{\text{flux},t}$ (a) and estimated $\hat{y}_{\text{flux},t}$ (b) density flux function.

The Figure 5 shows two confusion matrices in a 3-class classification problem. As we see here, the matrices are also diagonally dominated but nevertheless there are non-zero false positive and false negative elements. The overall accuracy together with other quality characteristics per class are summarized in Table 7. In this example we obtained over 79% accuracy for trees classification. The age classification of other tree species yielded fairly similar results. Hence a the sapflow

density can be treated as a characteristic for determining the age of a tree. This result can also be considered encouraging given the high noise content of the raw data, erroneous measurements, and missing values within individual classes. The use of Fourier coefficients derived from air temperature for classification can also be considered acceptable, although of course the quality is slightly degraded and in average is near 74%. During the experiments, we also noticed that when the VTA score is increased, the trees are more accurately classified according to the age group. Finally, it can be noticed that the higher the age of the trees, the more closely the sap density function takes on values. Classification then becomes in this case a more difficult task. This can be seen from the low values of the classification quality characteristics for classes 2 and 3 in Table 7.

Table 7. Classification performance.

Data \ Metric	ACC	TPR	TNR	F-1 Scores
$y_{\text{flux},t}$	0.7937	1 \rightarrow 0.8421 2 \rightarrow 0.8302 3 \rightarrow 0.6857	1 \rightarrow 0.7619 2 \rightarrow 0.8461 3 \rightarrow 0.7500	1 \rightarrow 0.9762 2 \rightarrow 0.8649 3 \rightarrow 0.8511
$\hat{y}_{\text{flux},t}$	0.7430	1 \rightarrow 0.8571 2 \rightarrow 0.7619 3 \rightarrow 0.6429	1 \rightarrow 0.8163 2 \rightarrow 0.8367 3 \rightarrow 0.9762	1 \rightarrow 0.7500 2 \rightarrow 0.7111 3 \rightarrow 0.7659

Example 3.4. Now we will fix the age group and try to classify the trees by VTA scores only. Consider data sets for *Acer platanoides*. The data were divided into four subgroups according to Table 8.

Table 8. Classes of *Acer platanoides*.

Class N	Age group	VTA score
1	VI	1
2	VI	2
3	VI	3
4	VI	4

The confusion matrices in Figure 6, although diagonally dominant, contain many non-zero elements outside the main diagonal. The reason is, that there was not much variation in the density flux data in each group. Therefore, we consider the classification accuracy of more than 75% as a very good result, taking into account that VTA is still a somewhat subjective characteristic. Table 9 shows that some classes are better recognized than others. This is not surprising, as it is obvious that in addition to age group and VTA there are other factors that influence the value of juice density, such as trunk diameter. In the following example we investigate the task of classification according to this characteristic.

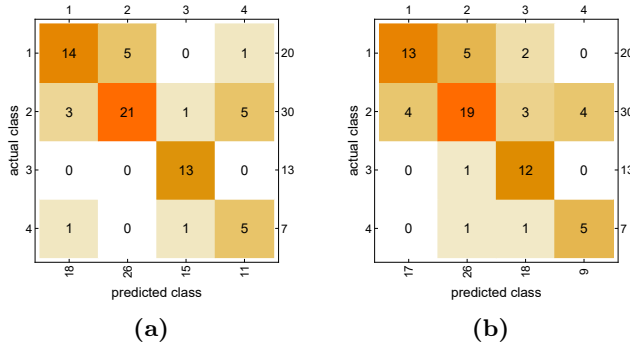


Figure 6. Confusion matrices for classification of *Acer platanoides* based on data set S (a) and \hat{S} (b).

Table 9. Classification performance.

Data \ Metric	ACC	TPR	TNR	F-1 Scores
<i>Acer platanoides</i> , S	0.7571	1 \rightarrow 0.7000 2 \rightarrow 0.7000 3 \rightarrow 1.000 4 \rightarrow 0.7143	1 \rightarrow 0.9200 2 \rightarrow 0.8750 3 \rightarrow 0.9649 4 \rightarrow 0.9048	1 \rightarrow 0.7368 2 \rightarrow 0.7500 3 \rightarrow 0.9286 4 \rightarrow 0.5556
<i>Acer platanoides</i> , \hat{S}	0.7000	1 \rightarrow 0.6500 2 \rightarrow 0.6333 3 \rightarrow 0.9230 4 \rightarrow 0.7143	1 \rightarrow 0.9200 2 \rightarrow 0.8333 3 \rightarrow 0.8947 4 \rightarrow 0.9365	1 \rightarrow 0.7027 2 \rightarrow 0.6786 3 \rightarrow 0.7742 4 \rightarrow 0.5039

Example 3.5. In this example we try to classify the trees by trunk diameter based on the density flux information with fixed factors of the age group and the VTA score. Two tree species are selected for the illustration: *Betula pendula* and *Tilia cordata*.

Table 10. Classes of *Betula pendula* (a) and *Tilia cordata* (b).

Class N	Diam	Age	VTA	Class N	Diam	Age	VTA
1	25.46	VI	1	1	37.87	VI	2
2	26.73	VI	1	2	48.76	VI	2
3	30.24	VI	1	3	51.24	VI	2

(a) (b)

Three different classes for each species are enumerated respectively in Table 10. Here we see that the quality of classification by trunk diameter is slightly higher than by age group, although these two factors have a large positive correlation for

almost all the tree species under consideration. As features we use here only the Fourier coefficients of data sets of type S obtained by fitting the truncated Fourier series to the density flux data sets. But we expect that the results will be similar to the case of the estimated Fourier coefficients. The results of classification are illustrated as usual in form of the confusion matrices in Figure 7 and in Table 11 of performance measures.

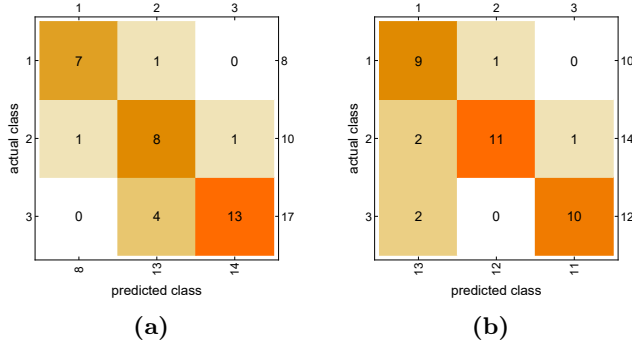


Figure 7. Confusion matrices for classification of *Betula pendula* (a) and *Tilia cordata* based on real $y_{\text{flux},t}$ density flux function.

Table 11. Classification performance.

Data \ Metric	ACC	TPR	TNR	F-1 Scores
<i>Betula pendula, S</i>	0.8000	1 \rightarrow 0.8750	1 \rightarrow 0.9629	1 \rightarrow 0.8750
		2 \rightarrow 0.8000	2 \rightarrow 0.8000	2 \rightarrow 0.6956
		3 \rightarrow 0.7647	3 \rightarrow 0.9444	3 \rightarrow 0.8387
<i>Tilia cordata, S</i>	0.8333	1 \rightarrow 0.9000	1 \rightarrow 0.8462	1 \rightarrow 0.7816
		2 \rightarrow 0.7857	2 \rightarrow 0.9545	2 \rightarrow 0.8461
		3 \rightarrow 0.8333	3 \rightarrow 0.9583	3 \rightarrow 0.8677

Here we see that the quality of classification by trunk diameter is more than 80% which is slightly higher than the classification by the age group, although these two factors have a large positive correlation for almost all the tree species under consideration.

4. Conclusion

On the basis of the proposed experiments, it can be noticed that the temperature observations can be mapped to the values of the sap flow density flux through the corresponding Fourier coefficients which is resulting in high quality predictions. Moreover, the estimated coefficients for the function approximating the sap

flow density have a good potential to be used as feature vector in trees classification tasks even within the same species. From this we can draw a conclusion about the perspective to use the TreeTalker equipment together with the proposed mathematical approach for solving problems of trees monitoring and anomaly state recognition. Moreover, if a tree's sapflow density pattern does not match what a healthy tree with similar characteristics should have, this can be seen as an indirect sign of problems with soil, groundwater or the general environment. As new data become available, we plan to continue our research on tree classification based on the monitoring system. We will also take into account the reviewer's suggestion related to the use of alternative classifiers and a comparative analysis of classification quality.

References

- [1] L. AHRENS, J. AHRENS, H. SCHOTTEN: *A machine-learning phase classification scheme for anomaly detection in signals with periodic characteristics*, Journal of Advances in Signal 27 (2019), p. 23, DOI: <https://doi.org/10.1186/s13634-019-0619-3>.
- [2] D. G. ALTMAN, J. M. BLAND: *Statistics Notes: Diagnostic tests 1: sensitivity and specificity*, BMJ 308.6943 (1994), p. 1552, DOI: <https://doi.org/10.1136/bmj.308.6943.1552>.
- [3] A. BOURSANIS, M. DIAMANTOULAKIS, A. LIOPA-TSAKALIDI, P. BAROUCAS, G. SALAHAS, S. GOUDOS: *Internet of Things (IoT) and Agricultural Unmanned Aerial Vehicles (UAVs) in Smart Farming: A Comprehensive Review*, Internet of Things 100187 (2020).
- [4] B. COLVERT, E. KANSO, E. ALSALMAN: *Classifying vortex wakes using neural networks*, Bioinspiration and Biomimetics 13.2 (2017), pp. 1–11, DOI: <https://doi.org/10.1088/1748-3190/aaa787>.
- [5] D. EFROSININ, I. KOCHETKOVA, N. STEPANOVA, A. YAROVSLAVTSEV, K. SAMOUYLOV, R. VALENTINI: *The Fourier Series Model for Predicting Sapflow Density Flux based on TreeTalker Monitoring System*, in: LNCS, NEW2AN 2020 (to be published), St. Petersburg, Russia: Springer, 2020.
- [6] C. GERSHENSON: *Artificial Neural Networks for Beginners*, 2003, arXiv: [cs/0308031](https://arxiv.org/abs/cs/0308031) [cs.NE].
- [7] F. JUNLIANG, W. YUE, F. ZHANG, H. CAI, X. WANG, X.-A. LU, Y. XIANG: *Evaluation of SVM, ELM and four tree-based ensemble models for predicting daily reference evapotranspiration using limited meteorological data in different climates of China*, Agricultural and Forest Meteorology 263 (2018).
- [8] D. P. KINGma, J. BA: *Adam: A Method for Stochastic Optimization*, 2014, arXiv: [1412.6980](https://arxiv.org/abs/1412.6980) [cs.LG].
- [9] S. RUSSELL, P. NORVIG: *Artificial Intelligence: A Modern Approach*, 3rd, USA: Prentice Hall Press, 2009, ISBN: 0136042597.
- [10] J. SIQUEIRA, T. PAC, J. SILVESTRE, F. SANTOS, A. FALCAO, L. PEREIRA: *Generating fuzzy rules by learning from olive tree transpiration measurement – An algorithm to au-tomatize Granier sap flow data analysis*, Computers and Electronics in Agriculture 101 (2014).
- [11] D. TANG, Y. FENG, W. HAO, N. CUI: *Evaluation of artificial intelligence models for actual crop evapotranspiration modeling in mulched and non-mulched maize croplands*, Computers and Electronics in Agriculture 152 (2018).

- [12] R. VALENTINI, L. MARCHESINI, D. GIANELLE, G. SALA, A. YAROVSLAVTSEV, V. VASENEV, S. CASTALDI: *New Tree Monitoring Systems: From Industry 4.0 to Nature 4.0*. *Annals of Silvicultural Research* 43.2 (2019), pp. 84–88, DOI: <http://dx.doi.org/10.12899/asr-1847>.
- [13] G. XU, Y. SHI, X. SUN, W. SHEN: *Internet of things in marine environment monitoring: A review*, *Sensors* 19.7 (2019).
- [14] S. YAMAC, M. TODOROVIC: *Estimation of daily potato crop evapotranspiration using three different machine learning algorithms and four scenarios of available meteorological data*, *Agricultural Water Management* 228 (2020).

Ensemble noisy label detection on MNIST*

István Fazekas^a, Attila Barta^b, László Fórián^b

^aUniversity of Debrecen, Department of Applied Mathematics and Probability Theory
fazekas.istvan@inf.unideb.hu

^bUniversity of Debrecen, Doctoral School of Informatics
barta.attila@inf.unideb.hu
forian.laszlo@inf.unideb.hu

Submitted: January 18, 2021

Accepted: March 29, 2021

Published online: May 18, 2021

Abstract

In this paper machine learning methods are studied for classification data containing some misleading items. We use ensembles of known noise correction methods for preprocessing the training set. Preprocessing can be either relabeling or deleting items detected to have noisy labels. After preprocessing, usual convolutional networks are applied to the data. With preprocessing, the performance of very accurate convolutional networks can be further improved.

Keywords: Label noise, deep learning, classification

AMS Subject Classification: 68T07

1. Introduction

In recent years, deep neural networks have reached very impressive performance in the task of image classification. However, these models require very large datasets with labeled training examples, and such datasets are not always available. The labeling process is often very expensive, or it is very difficult even for experts in a particular field. That is what can lead to the use of databases with label noise, which contain incorrectly labeled instances. Therefore, it is important to examine

*This work was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund.

training on this kind of datasets. According to a widely accepted assumption, deep networks learn consistent, simple patterns in the beginning, and then it is followed by the learning of the harder examples with possibly incorrect labels [2]. So treating the label noise in the training set can lead to a better generalization ability instead of its overfitting to the wrong examples. A lot of studies address the noisy label problems, for example, [1] is an extensive survey of a broad range of the existing methods.

In this work we investigate the possibilities of improving a classifier (which is an ensemble of deep neural networks) by handling the label noise in the training dataset. We classify with an ensemble of convolutional neural networks (CNNs). At the start, we train that ensemble on the original training dataset. Then we are going to apply a label noise cleansing technique on the training data. Finally, we take a CNN ensemble with the same structure as our original CNN ensemble, and train it on the new dataset gained after treating the label noise. We evaluate and compare the performance of the ensemble classifiers and draw conclusions. As our main goal is to study label correcting neural networks for preprocessing purpose, so here we use only simple ensemble building methods. The effect of more sophisticated voting systems (see, e.g. [7]) on our two-phase classification method can be studied later.

We conduct experiments on the MNIST dataset [6]. MNIST is a database of handwritten digits, it consists of images with 28×28 grayscale pixels. The size of the training set is 60 000 examples and the test set has 10 000 samples. However, MNIST contains some misleading items. To support it, some examples that might be considered as mislabeled images can be seen on Figure 1.

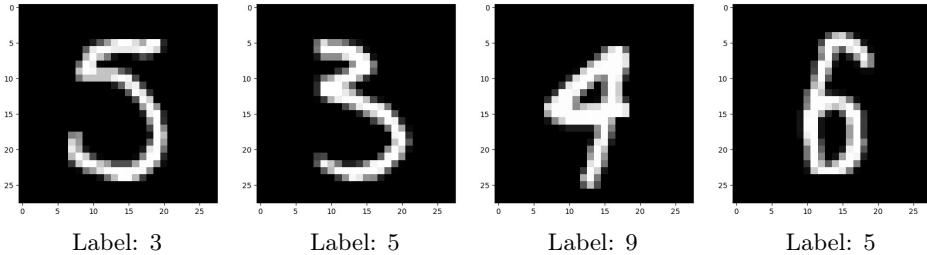


Figure 1. Some misleading images in the MNIST dataset.

We shall consider the misleading items as inaccurately labelled ones so we can apply some known methods elaborated to handle noisy labels.

2. Methods to detect and correct label noise

In the beginning, let us define some notations. We denote vectors with bold (e.g. \mathbf{x}) letters and matrices with capital (e.g. X) letters. Specifically, $\mathbf{1}$ corresponds to a vector of all-ones. In the c -dimensional space, the term of hard-label and soft-

label spaces are used, they are denoted by $\mathcal{H} = \{\mathbf{y} : \mathbf{y} \in \{0, 1\}^c, \mathbf{1}^\top \mathbf{y} = 1\}$ and $\mathcal{S} = \{\mathbf{y} : \mathbf{y} \in [0, 1]^c, \mathbf{1}^\top \mathbf{y} = 1\}$, respectively. It is easy to see that \mathcal{H} contains one-hot vectors, and the elements of \mathcal{S} are discrete probability distributions.

In an image classification problem with c classes, we have a training set of n images: $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ with the corresponding ground-truth

$$Y^{GT} = \{\mathbf{y}_1^{GT}, \mathbf{y}_2^{GT}, \dots, \mathbf{y}_n^{GT}\}$$

labels. $\mathbf{y}_i^{GT} \in \mathcal{H}$ represents the class of \mathbf{x}_i with a 1 in the coordinate corresponding to that class.

Using a neural network with the cross entropy loss function, the model can be trained by minimizing

$$\mathcal{L} = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^c \mathbf{y}_{ij}^{GT} \log s_j(\theta, \mathbf{x}_i),$$

where θ is the set of the network parameters, \mathbf{y}_{ij}^{GT} is the j -th element of \mathbf{y}_i^{GT} and s_j is the j -th element of the network's softmax output. If the clean labels are given, we only have to minimize this function with respect to θ .

However, in this noisy label setting, the ground-truth labels are not known, only

$$Y = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n\}$$

is given, which is the set of noisy labels. But the goal is the same: our task is to train the model to predict the true labels.

2.1. A joint optimization framework

The first technique we have used is the framework in Tanaka et al. [8]. The authors of that paper suggest a two-stage approach. The noise correction is made in the first phase by jointly optimizing the weights of a neural network and the labels of the training data. During this joint optimization process they train a classifier and correct the wrong labels at the same time. It is made possible by repeating alternating steps of updating the network parameters and the training labels. In the early stages, the training goes in the usual way, but a high learning rate is used, because it prevents the learning of noisy labels. When the classifier has achieved a reasonable accuracy, they start the repetition of the two alternating steps mentioned before. The first is the well known update of the network weights by the stochastic gradient descent method. In the other step, they update the labels. And from now on, a more complex loss function is used, two regularization terms are added to the classification loss to prevent certain anomalies. Once this label correction is done, the authors start the training over in the second step with the recently obtained new labels and without the two regularization terms of the loss function. The new labels are considered as clean and the trained network is considered as accurate.

In [8] the noisy label problem is formulated as the joint optimization of the network parameters and the labels:

$$\min_{\theta, Y} \mathcal{L}(\theta, Y|X).$$

The loss function is

$$\mathcal{L}(\theta, Y|X) = \mathcal{L}_c(\theta, Y|X) + \alpha \mathcal{L}_p(\theta|X) + \beta \mathcal{L}_e(\theta|X), \quad (2.1)$$

where $\mathcal{L}_p(\theta|X)$, $\mathcal{L}_e(\theta|X)$ are the regularization losses, and α, β are hyper-parameters. The $\mathcal{L}_c(\theta, Y|X)$ classification loss is made with the Kullback-Leibler divergence of the labels and the softmax outputs:

$$\mathcal{L}_c(\theta, Y|X) = \frac{1}{n} \sum_{i=1}^n D_{KL}(\mathbf{y}_i || \mathbf{s}(\theta, \mathbf{x}_i)),$$

where

$$D_{KL}(\mathbf{y}_i || \mathbf{s}(\theta, \mathbf{x}_i)) = \sum_{j=1}^c y_{ij} \log \left(\frac{y_{ij}}{s_j(\theta, \mathbf{x}_i)} \right).$$

[8] introduces two possible ways to update the labels at the end of the epochs: the hard-label method and the soft-label method. In the first case, the new labels are one-hot vectors, too. A $y \in \mathcal{H}$ is updated in the following way:

$$y_{ij} = \begin{cases} 1, & \text{if } j = \arg \max_k s_k(\theta, \mathbf{x}_i), \\ 0, & \text{otherwise.} \end{cases}$$

In the second case, the new labels are the softmax outputs:

$$\mathbf{y}_i = \mathbf{s}(\theta, \mathbf{x}_i).$$

Tanaka et al. [8] experienced that the soft-label method performed better than the hard-label method. In experiments, sudden changes in the labels were prevented by the use of a relatively high momentum (0.9), and the new soft labels were not single softmax outputs, but the average of softmax outputs in the last 10 epochs.

Finally, we describe regularization terms in the loss function of [8]. The regularization loss $\mathcal{L}_p(\theta|X)$ is needed to prevent the assignment of all labels to a single class. If we only minimize $\mathcal{L}_c(\theta, Y|X)$ with respect to θ and Y , this is a trivial solution. To solve this issue, we use the prior distribution p of the classes in the entire training set. We do not let the distribution of the updated labels be much different from p , so we introduce the Kullback-Leibler divergence of p and $\bar{s}(\theta, X)$ as a loss function term:

$$\mathcal{L}_p = \sum_{j=1}^c p_j \log \frac{p_j}{\bar{s}_j(\theta, X)}.$$

$\bar{s}(\theta, X)$ is approximated by calculating the mean of softmax outputs over each mini-batch.

After the start of label updating in the case of soft labels, it might happen that the network output is the same as the soft label for most of the training examples, and it stops the learning process. That is why the entropy loss $\mathcal{L}_e(\theta|X)$ is needed.

$$\mathcal{L}_e(\theta|X) = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^c s_j(\theta, \mathbf{x}_i) \log s_j(\theta, \mathbf{x}_i)$$

This entropy term forces the probability distribution of the soft labels to concentrate to a single class.

This noise correcting and training process needs a background network. To this end, the PreAct ResNet [5] was used in [8]. It is a modification of the famous ResNet network [4]. Residual Networks give a simple yet groundbreaking solution to the vanishing gradient problem. They use identity shortcuts, which let the data skip one or more layers. Obviously, the error back-propagation is the point where these models can really take advantage of these shortcuts. The pre-activation residual blocks [5] let the gradients flow throughout the PreAct ResNet even more easily. Such networks may have hundreds of layers and researchers consider them more accurate than ResNets.

It is important to note that this framework does not depend on the background network structure, it can be used with any background network. From now on, this technique will be referred to as Tanaka's method.

2.2. Probabilistic end-to-end noise correction

The second method we have used is introduced in [10]. This framework is called PENCIL (probabilistic end-to-end noise correction in labels). This technique uses soft labels, too. So the labels of the images are not fixed categorical values, but distributions among all possible labels. Similarly to [8], the labels are updated iteratively during the training of a classifier. But those updates are made with back-propagation instead of the moving average of softmax outputs.

For every image \mathbf{x}_i , a label distribution $\mathbf{y}_i^d \in \mathcal{S}$ is maintained and updated. These distributions are the estimations of the \mathbf{y}_i^{GT} clean labels. The $\mathbf{y}_i^d \in \mathcal{S}$ values are initialized based on the given noisy labels \mathbf{y}_i . The initialization goes in the following way. An additional label $\tilde{\mathbf{y}}_i$ is used to assist \mathbf{y}_i^d . $\tilde{\mathbf{y}}_i$ is initialized by multiplying the given \mathbf{y}_i with a large constant:

$$\tilde{\mathbf{y}}_i = K\mathbf{y}_i.$$

(K is the same value for all i , in [10] K is 10). $\tilde{\mathbf{y}}_i$ is then transformed into a probability distribution with softmax. This will be the value of \mathbf{y}_i^d :

$$\mathbf{y}_i^d = \text{softmax}(\tilde{\mathbf{y}}_i).$$

The loss function terms are also similar to [8], but there are important differences. The authors showed that Kullback-Leibler divergence with interchanged

arguments is more suitable for this noise correction task than the classic form. So here the classification loss \mathcal{L}_c is the following:

$$\mathcal{L}_c = \frac{1}{n} \sum_{i=1}^n D_{KL}(\mathbf{s}(\theta, \mathbf{x}_i) || \mathbf{y}_i^d),$$

where

$$D_{KL}(\mathbf{s}(\theta, \mathbf{x}_i) || \mathbf{y}_i^d) = \sum_{j=1}^c s_j(\theta, \mathbf{x}_i) \log \left(\frac{s_j(\theta, \mathbf{x}_i)}{y_{ij}^d} \right).$$

As we have seen before, we have to prevent the assignment of all instances to a single class if we begin the update of the labels. It should not be allowed for the estimated label distribution \mathbf{y}_i^d to be much different from the original \mathbf{y}_i . Therefore, in [10] a cross entropy loss term is introduced between label distribution and the noisy label. It is called compatibility loss:

$$\mathcal{L}_o(\mathbf{Y}, \mathbf{Y}^d) = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^c y_{ij} \log y_{ij}^d,$$

where \mathbf{Y} is the set of given noisy labels, and \mathbf{Y}^d is the set of the estimated labels.

There is one more issue to prevent during the label correction: if $s(\theta, \mathbf{x}_i)$ is equal to \mathbf{y}^d , it stops the training and label updating process, so the softmax outputs need to be forced to concentrate on a single class. An entropy loss is suitable for this requirement:

$$\mathcal{L}_e(s(\theta, \mathbf{x})) = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^c s_j(\theta, \mathbf{x}_i) \log s_j(\theta, \mathbf{x}_i).$$

This term is exactly the same as \mathcal{L}_e in [8].

The PENCIL loss function is a weighted sum of these terms:

$$\mathcal{L} = \frac{1}{c} \mathcal{L}_c(s(\theta, \mathbf{x}), \mathbf{Y}^d) + \alpha \mathcal{L}_o(\mathbf{Y}, \mathbf{Y}^d) + \frac{\beta}{c} \mathcal{L}_e(s(\theta, \mathbf{x})), \quad (2.2)$$

where α and β are two hyperparameters.

The training with PENCIL begins with a fixed, high learning rate, because it helps not to overfit to noisy labels. In the next stage, the label correction starts with the (2.2) loss function. \mathbf{y}^d is updated by updating $\tilde{\mathbf{y}}$. The advantage of this labeling is that $\tilde{\mathbf{y}}$ can be updated freely without any constraint while \mathbf{y}^d is always a probability distribution. It is important to note that a very large learning rate is needed to update $\tilde{\mathbf{y}}$. Finally, the network is fine-tuned with only the classification loss.

In the paper [10] the PreAct ResNet is used as a background network, but the PENCIL framework can also be used with any neural network.

3. Our experiments

Tanaka et al. [8] made experiments on CIFAR-10 with synthetic label noise, and a real-world dataset, in which almost 40 percent of the labels are wrong [9]. Yi and Wu [10] have also conducted experiments with synthetic label noise and real-world datasets.

In our work, we use a preprocessed dataset without adding synthetic label noise. However, we do not treat it as a perfectly clean training set. We suppose the existence of a certain, but not too large amount of label noise in MNIST. As mentioned before, we train an ensemble of CNN classifiers before and after the label noise cleansing. We perform this correction with the first phase of the method seen in section 2.1, and the technique in 2.2. To further enhance this procedure, we have also used an ensemble for this label noise cleansing, too. Our goal is to examine its effect on the dataset, the learning process, and the accuracy.

We implemented our experiments with the Python-based deep learning framework Tensorflow and the Keras library. Our first task was to implement the loss functions (2.1) and (2.2). Then we built a custom training loop for both frameworks to update the labels. In the case of PENCIL, the main task was to write the code for the backpropagation to update $\tilde{\mathbf{y}}$. Tensorflow’s Gradient Tape made it easy for us. For the joint optimization framework (Tanaka’s method) we used the average of the last 10 epoch’s output for the updating.

The CNN ensembles are built up with very accurate convolutional neural networks [3]. A Keras summary of such a CNN can be seen in the appendix. We used this CNN with structure in Table 8 as the background network of the label noise cleansing frameworks, too.

3.1. Comparison of the two cleansing frameworks

In order to get a better result we increased our dataset with 60 000 augmented images, where the augmentation was a little amount of random shift and rotation. In our experiments the first step was to train a base-model with the cross entropy loss function with a high learning rate ($lr = 1$). After 20 epochs we saved our model and used it for both frameworks as a base-model. As a second step, we continued with the label changing phase of the two methods. Table 1 shows the parameters we used and how they worked on the different frameworks in this stage. In the last two columns, we show how many labels were detected as incorrect on the whole dataset and in parentheses we show only the detected labels from the original dataset. The last column corresponds to an ensemble of a PENCIL and a Tanaka network, which was made by taking the average of the softmax outputs.

We also wanted to examine how differently the detection works in the case of these two frameworks. Table 2 contains the number of images, which were classified into the same new class according to the pairwise intersections of Tanaka, PENCIL and the ensemble of them. The table shows that different methods detect mostly the same label noise. Figure 2 shows some seemingly mislabeled training images,

which were found by both frameworks. The subcaptions contain the new labels and the highest scores generated by the cleansing methods. The Tanaka process seems to be more confident than PENCIL according to the higher peak scores. It can be generally concluded on the whole set of detected instances.

Table 1. Noisy label detection of ensembles on the training set with 120 000 samples.

Framework	α	β	lr	λ	epochs	Detected labels	Ensemble
PENCIL	0.05	0.6	0.1	600	30	71 (28)	75 (36)
Tanaka	1.1	0.6	0.05	-	20	116 (59)	

Table 2. The number of identically detected images.

	PENCIL	Tanaka
Tanaka	54 (24)	-
Ensemble	55 (24)	74 (36)

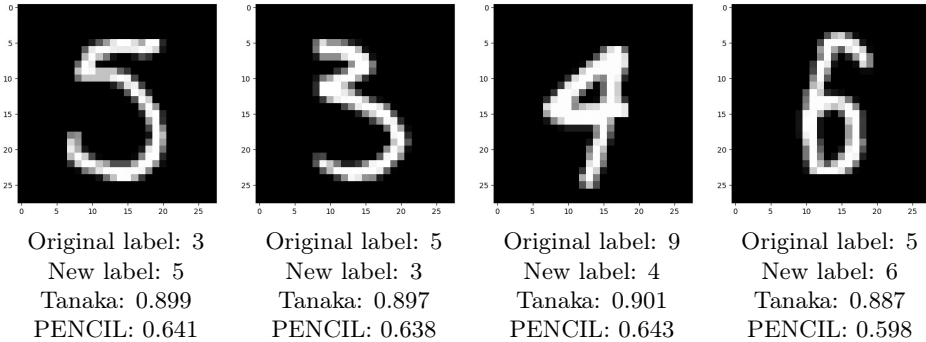


Figure 2. Some detected images with new labels and the corresponding scores of Tanaka and PENCIL.

Then in this experiment we wanted to investigate how the exclusion of the detected labeled inputs affects the goodness of the CNN. Here we use a single very accurate CNN with structure in Table 8. In Table 3 we show that how the different training datasets performed with the same weight initialized models. The table contains the performance of our CNN model trained on the original augmented dataset and on its cleaned versions. Of course, they were evaluated on the test dataset. In Table 3, column 'Original' contains the results of the CNN without cleaning the training set. Columns 'Tanaka' and 'PENCIL' contain the result of the CNN after cleaning with the Tanaka and PENCIL methods, respectively. 'Ensemble' means that the cleaning was made by an ensemble of a Tanaka and a

PENCIL network, with weighting the same way as the previous case. The * symbol denotes the cases when we deleted the detected items only from the artificially created part of the training dataset. We repeated each experiment 30 times and in each case with a learning rate of 0.1 and a momentum of 0.2 for 30 epochs. In Table 3 we show the best, the worst and the mean of the 30 repetitions.

Table 3. Training with cleaned labels.

	Original	Tanaka	Tanaka*	PENCIL	PENCIL*	Ensemble	Ensemble*
Max	99.70%	99.67%	99.64%	99.68%	99.65%	99.64%	99.60%
Mean	99.57%	99.56%	99.55%	99.49%	99.40%	99.57%	99.44%
Min	99.30%	99.40%	99.34%	99.35%	99.36%	99.37%	99.34%

We can see that the result depends on the cleaning method. We can also see that removing the misleading items makes the classification more stable in the following sense. The minimum is higher and the range is smaller after the use of each cleaning method than for the original raw dataset. For the original dataset the minimum is 99.30% and the range is $99.70 - 99.40 = 0.40\%$ while for Tanaka the minimum is 99.40% and the range is 0.27%. However, such simple cleaning techniques do not improve the average and the maximal performance.

3.2. Possibilities of improving a CNN ensemble classifier

Our final goal was to examine the opportunities of making an already accurate CNN ensemble classifier even better. An ensemble of 3 convolutional neural networks was trained before and after label noise cleansing. In our ensemble, 3 CNNs with structure in Table 8 were used. For fair comparison, these networks were initialized with the same weights in each case. All of them were trained with a learning rate of 0.1 and a momentum of 0.2 for 30 epochs.

The treating of the label noise was carried out in the following way: 3 networks were trained with both frameworks. We took the average of the label estimations of the 6 networks and applied the NumPy argmax function. These were the new labels corresponding to the training examples.

In the following experiment the training data was slightly increased with 24 000 augmented images, so the training set consisted of 84 000 examples in this setting. For both noise cleansing frameworks, the training of the models began with 20 epochs using the cross entropy loss function and a momentum of 0.3. In the second phase, the momentum was set to 0.5 for the networks with Tanaka’s method and 0.1 for both of PENCIL’s optimizers, because of the nature of these techniques. The other parameters can be seen in Table 4. Of course, the number of epochs in this table means the number of epochs in this phase. In the ‘Detected labels’ column, there is the number of labels detected as noisy by the ensemble of the 3 networks corresponding to the methods. In parentheses, the number of noisy labels are shown in the original dataset. The last column contains the results of ensembling all the 6 networks.

Table 4. Noisy label detection of ensembles on the training set with 84 000 samples.

Framework	α	β	lr	λ	epochs	Detected labels	Ensemble
PENCIL	0.08	0.5	0.2	550	25	52 (27)	36 (24)
Tanaka	1.1	0.6	0.04	-	20	56 (32)	

In the next part of the experiment, the different options of label noise handling are investigated. Table 5 contains the test performance of the CNN ensembles trained on this augmented dataset with the original labels, relabeling and deletion. These results correspond to 20 runs in each case with the same 20×3 weight set initialization. The first and third rows show the best and weakest performances, while the second line contains the mean of the 20 test accuracies.

Table 5. Performance of the CNN ensemble with different noise handling options.

	CNN ensemble before	CNN ensemble after relabeling	CNN ensemble after deletion
Max	99.68%	99.71%	99.72%
Mean	99.663%	99.675%	99.673%
Min	99.58%	99.61%	99.60%

Finally, we wanted to investigate the opportunities of this CNN ensemble improvement by using only the original 60 000 samples. In this setting, the parameters of the noise detecting ensemble are the same as before and this table corresponds to 3 PENCIL and 3 Tanaka networks, too. The number of labels detected as wrong are visible below and the performance of the CNN ensembles are shown in Table 7, in the same way as in Table 5.

Table 6. Noisy label detection of ensembles on the original MNIST dataset.

Framework	Detected labels	Ensemble
PENCIL	29	21
Tanaka	24	

Table 7. Performance of the CNN ensemble with different noise handling options on the original MNIST.

	CNN ensemble before	CNN ensemble after relabeling	CNN ensemble after deletion
Max	99.67%	99.69%	99.71%
Mean	99.647%	99.654%	99.658%
Min	99.58%	99.57%	99.59%

4. Conclusions

Machine learning methods developed for classification data with label noise can be applied to handle datasets containing some misleading items. These machine learning methods can be used for preprocessing the training data. After preprocessing any usual classification tool can be applied. Deleting some misleading items in the preprocessing phase is more promising than relabeling them. With preprocessing we can further improve the performance of very accurate convolutional networks, too. For preprocessing, ensembles of different noise correction methods (like the method of Tanaka et al. [8] and PENCIL of [10]) are promising. However, we have to be careful with relabeling and with removal of relabeled items, too. Relabeling means adding information to the training set artificially. If we relabel too many images, it may happen that we mislead the classifier with those modified labels. The removal of the relabeled examples is also dangerous: it can cause information loss that degrades the performance of our models. (With high noise rates, it is obviously a wrong choice.) We can improve a classifier with those operations only if we find the right amount of training data to relabel or delete.

Acknowledgements. The authors are indebted to the referees for their valuable suggestions.

References

- [1] G. ALGAN, I. ULUSOY: *Image classification with deep learning in the presence of noisy labels: A survey*, Knowledge-Based Systems 215 (2021), p. 106771, ISSN: 0950-7051, DOI: <https://doi.org/10.1016/j.knsys.2021.106771>, URL: <https://www.sciencedirect.com/science/article/pii/S0950705121000344>.
- [2] D. ARPIT, S. JASTRZEBSKI, N. BALLAS, D. KRUEGER, E. BENGIO, M. S. KANWAL, T. MAHARAJ, A. FISCHER, A. COURVILLE, Y. BENGIO, S. LACOSTE-JULIEN: *A Closer Look at Memorization in Deep Networks*, in: ed. by D. PRECUP, Y. W. TEH, vol. 70, Proceedings of Machine Learning Research, International Convention Centre, Sydney, Australia: PMLR, Aug. 2017, pp. 233–242, URL: <http://proceedings.mlr.press/v70/arpit17a.html>.
- [3] C. DEOTTE: *How to choose CNN Architecture MNIST*, 2018, URL: <https://www.kaggle.com/cdeotte/how-to-choose-cnn-architecture-mnist>.
- [4] K. HE, X. ZHANG, S. REN, J. SUN: *Deep Residual Learning for Image Recognition*, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778, DOI: <https://doi.org/10.1109/CVPR.2016.90>.
- [5] K. HE, X. ZHANG, S. REN, J. SUN: *Identity Mappings in Deep Residual Networks*, in: Computer Vision – ECCV 2016, ed. by B. LEIBE, J. MATAS, N. SEBE, M. WELLING, Cham: Springer International Publishing, 2016, pp. 630–645, ISBN: 978-3-319-46493-0, DOI: https://doi.org/10.1007/978-3-319-46493-0_38.
- [6] Y. LECUN, C. CORTES, C. BURGES: *MNIST handwritten digit database*, ATT Labs [Online]. Available: <http://yann.lecun.com/exdb/mnist> 2 (2010).
- [7] T. TAJTI: *New voting functions for neural network algorithms*, Annales Mathematicae et Informaticae 52 (2020), pp. 229–242, ISSN: 1787-6117, DOI: <https://doi.org/10.33039/ami.2020.10.003>.

-
- [8] D. TANAKA, D. IKAMI, T. YAMASAKI, K. AIZAWA: *Joint Optimization Framework for Learning With Noisy Labels*, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018, pp. 5552–5560, DOI: <https://doi.org/10.1109/CVPR.2018.00582>.
- [9] TONG XIAO, TIAN XIA, YI YANG, CHANG HUANG, XIAOGANG WANG: *Learning from massive noisy labeled data for image classification*, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 2691–2699, DOI: <https://doi.org/10.1109/CVPR.2015.7298885>.
- [10] K. YI, J. WU: *Probabilistic End-To-End Noise Correction for Learning With Noisy Labels*, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

Appendix

Table 8. The Keras summary of the CNN that we used.

```

Model: "sequential"
-----
Layer (type)                Output Shape                Param #
-----
conv2d (Conv2D)              (None, 26, 26, 32)         320
-----
batch_normalization (BatchNormalization) (None, 26, 26, 32)         128
-----
conv2d_1 (Conv2D)            (None, 24, 24, 32)         9248
-----
batch_normalization_1 (BatchNormalization) (None, 24, 24, 32)         128
-----
conv2d_2 (Conv2D)            (None, 12, 12, 32)         25632
-----
batch_normalization_2 (BatchNormalization) (None, 12, 12, 32)         128
-----
dropout (Dropout)            (None, 12, 12, 32)         0
-----
conv2d_3 (Conv2D)            (None, 10, 10, 64)         8496
-----
batch_normalization_3 (BatchNormalization) (None, 10, 10, 64)         256
-----
conv2d_4 (Conv2D)            (None, 8, 8, 64)           36928
-----
batch_normalization_4 (BatchNormalization) (None, 8, 8, 64)           256
-----
conv2d_5 (Conv2D)            (None, 4, 4, 64)           102464
-----
batch_normalization_5 (BatchNormalization) (None, 4, 4, 64)           256
-----
dropout_1 (Dropout)          (None, 4, 4, 64)           0
-----
flatten (Flatten)            (None, 1024)                0
-----
dense (Dense)                 (None, 128)                  131200
-----
batch_normalization_6 (BatchNormalization) (None, 128)                  512
-----
dropout_2 (Dropout)          (None, 128)                  0
-----
dense_1 (Dense)               (None, 10)                   1290
=====
Total params: 327,242
Trainable params: 326,410
Non-trainable params: 832

```


How well can screening sensitivity and sojourn time be estimated*

Ayman Hijazy^{ab}, András Zempléni^a

^aDepartment of Probability Theory and Statistics, Eötvös Loránd University
aymanhijazy@caesar.elte.hu

^bFaculty of Informatics, University of Debrecen
andras.zempleni@ttk.elte.hu

Submitted: December 22, 2020

Accepted: March 8, 2021

Published online: May 18, 2021

Abstract

Chronic disease progression models are governed by two main parameters: preclinical intensity and sojourn time. The estimation of these parameters helps in optimizing screening programs (with an additional parameter: sensitivity of the screens), and we examine their effect in improving survival. Multiple approaches exist for estimating these parameters. However, these models are based on strong underlying assumptions. Our main aim is to investigate the effect of these assumptions. For this purpose, we developed a simulator to mimic a breast cancer screening program while directly observing the exact onset and the sojourn time of the disease. We then examine the performance of the model under different parameterizations and investigate the effects of different models on the sensitivity, the inter-screening intervals and misspecification of the used parametric distributions. Our results indicate a strong correlation among the estimated parameters. Besides, the underlying assumptions have a strong effect on the overall performance of the model. These findings shed a light on the seemingly discrepant results obtained by different authors using the same data sets but different assumptions.

Keywords: Disease progression, likelihood, screening sensitivity, sojourn time

*The project has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.2-16-2017-00015).

1. Introduction

Statistical modeling of natural disease progression aids in understanding its dynamics and forecasting its incidence rates. This allows better prevention and treatment plans which improves survival. However, in many cases, some data is not observable, as some diseases have an asymptomatic phase in which the patient does not know he has the sickness yet.

In the model proposed by Shen and Zelen [12], the natural progression of a disease is regarded as a three state model (see Figure 1): individuals progress from a disease free state S_f to the preclinical state S_p , when the disease has become onset but is still asymptomatic, i.e. the person has the disease but it has not shown any symptoms. The final state of the disease from this point of view is when it manifests itself through clinical symptoms, thus it is called the clinical state S_c .

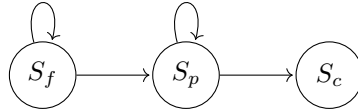


Figure 1. Progression in the three state model.

The flow in the process is governed by the preclinical intensity and the sojourn time. The preclinical intensity is the probability of moving from the disease free state to the preclinical one during $(t, t + dt)$. Equivalently, it is the waiting time in the disease free state S_f . The sojourn time is defined as the amount of time spent in the preclinical state S_p , in other words it is the time needed for the disease to show itself by means of clinical symptoms. However, directly observing sojourn time is not feasible as the exact time of onset is unknown. The sojourn time is then estimated through modelling, mostly by assuming it is a random variable with a specified distribution, see e.g. [16, 17].

Early detection methods such as screening allows discovering the disease before any symptoms appear. Screening sensitivity, defined as the probability of detection given that the patient is in S_p , is crucial in determining the efficiency of the screening program.

The parameters of interest in such a process are the preclinical intensity, the sojourn time and screening sensitivity. The estimation of these parameters is essential to optimize screening intervals and to correct lead time bias, that is defined as the apparent increase in survival due to early detection by means of screening.

In this paper we aim to investigate the identifiability of the parameters governing the process in different setups. For that purpose, a simulator is developed to record the exact onset and sojourn times of patients. This allows us to assess the accuracy of the estimators by comparing the estimated values to the real ones.

The theoretical basis of disease progression models under periodic screening was set by Zelen and Feinleib [17] and Prorok [10]. Later, Shen and Zelen [12] intro-

duced two models to estimate the parameters governing disease progression. The first describes stable diseases that are assumed to have incidence and prevalence independent of time or age. The other incorporates time dependence of incidence and prevalence to the model (these cases are called non stable diseases). We investigate the common cases, when incidence is age-dependent.

Wu et al. [16] further extended the results by allowing both the transition probability from the disease-free state and the sensitivity to be age-dependent. They assume that the sojourn time follows a loglogistic distribution, the preclinical intensity has a lognormal distribution and the sensitivity is age-dependent, the age-dependence is incorporated by assuming that the sensitivity has a logistic function form.

Generally, these models are built by deriving the probabilities of cases being detected by screening or symptoms, this allows the forming a likelihood function from which parameters can be estimated by classical methods such as maximization of the likelihood function, a least squares approach or a Bayesian one. However, many questions can be raised about the effects of the assumptions one makes when modelling such a scenario.

The paper is organized as follows: we lay the model foundations in Section 2. Next, we setup the simulations in Section 3, we then present our simulation based on results in Section 4. We then show the reasons behind the discrepancy of estimates in the literature in Section 5. Finally, we summarize our findings in Section 6.

2. The model

We will use the generalized model proposed by Wu et al. [16] in this paper. Now, to lay the setup of the model, we suppose an individual becomes onset at a random time X where X is a random variable with a (possibly) defective density $f_X(x)$. Introduce $r \leq 1$ as the lifetime risk, which is the probability of a person to get the disease i.e. $\int_0^\infty f_X(x) dx = r$. After a case becomes onset, we suppose that it stays in the preclinical state for a random amount of time Y (called sojourn time) independent of X , where Y is a random variable with pdf $f_Y(y)$ and survivor function $Q_Y(y)$. Let $Z = X + Y$ denote the time of diagnosis. As X and Y are assumed independent, the density of Z in the absence of screening is given by the convolution of X and Y , namely: $f_Z(z) = \int_0^z f_X(x)f_Y(z-x) dx$.

If no screens are organized, i.e. the patient only knows that he has the sickness when symptoms are exhibited, then one can only observe the time of diagnosis Z and the likelihood function would simply be the product of the densities $f_Z(z_i)$. The identifiability of the parameters in such a setup is a serious concern, for instance if both X and Y are normally distributed, there are infinitely many parameters which can generate the same distributions. We currently investigate the theoretical aspects of identifiability which we will publish in a separate paper.

Suppose now that a screening program consisting of K screens is organized for a population which is stratified by age at the first screen t_1 where $t_1 = t_{\min}, \dots, t_{\max}$.

Let us define Δ as the inter-screening time and assume that all participants are disease free at age t_0 before the first screen (we assume $t_0 = 0$). Denote by $t_i = t_1 + (i - 1)\Delta$ the age at the i^{th} screen and by (t_{i-1}, t_i) the i^{th} screening interval. Denote the sensitivity of a screen by $\xi(t)$ and suppose it is a parametric function of the age at screening. The aim is to build a likelihood function using the probabilities of detection by screens and by showing symptoms.

Under this setup, the probability of detection at the first screen for those aged t_1 at the first screen denoted by D_{1,t_1} is given by cases which have moved to S_p in (t_0, t_1) and stayed there till they are screened positively. Therefore:

$$D_{1,t_1} = \xi(t_1) \int_{t_0}^{t_1} f_X(x) Q_Y(t_1 - x) dx.$$

In order to determine the probability of detection at the k^{th} screen, let us discretize the timeline into intervals of the form (t_{i-1}, t_i) . Denote by $D_{k,t_1}^{(i)}$ the contribution of cases which have moved to S_p in the i^{th} screening interval to the probability of detection at the k^{th} screen for $i = 1, \dots, K$. That is given by cases which have been falsely screened negative in all the previous screens and they did not show symptoms before the k^{th} screen when they were finally screened positively. Hence:

$$D_{k,t_1}^{(i)} = \begin{cases} \xi(t_k) \left[(1 - \xi(t_i)) \cdots (1 - \xi(t_{k-1})) \right] \int_{t_{i-1}}^{t_i} f_X(x) Q_Y(t_k - x) dx, & \text{if } i < k, \\ \xi(t_k) \int_{t_{k-1}}^{t_k} f_X(x) Q_Y(t_k - x) dx, & \text{if } i = k. \end{cases} \quad (2.1)$$

Hence, the probability of detection at the k^{th} screen is given by the sum of contributions:

$$D_{k,t_1} = \sum_{i=1}^k D_{k,t_1}^{(i)}.$$

A similar approach is used to determine the probability of showing symptoms in the k^{th} screening interval. Denote by $f_{Z_{t_1}}^{(i,k)}$ the contribution of cases which have moved to S_p in (t_{i-1}, t_i) to the probability of showing symptoms between $(z, z + dz)$ where $t_{k-1} < z < t_k$. Therefore:

$$f_{Z_{t_1}}^{(i,k)}(z) = \begin{cases} \int_{t_{k-1}}^z f_X(x) f_Y(z - x) dx, & \text{if } k = i, \\ \int_{t_{i-1}}^{t_i} f_X(x) f_Y(z - x) dx \prod_{j=i}^{k-1} (1 - \xi(t_j)), & \text{if } k > i. \end{cases} \quad (2.2)$$

Hence, the probability of a case to show symptoms between z and $z + dz$ for $t_k < z < t_{k+1}$ is given by:

$$f_{Z_{t_1}}^k(z) = \sum_{i=1}^k f_{Z_{t_1}}^{(i,k)}(z). \quad (2.3)$$

The probability for a case to show symptoms between t_{k-1} and t_k for individuals aged t_1 at the first screen denoted by I_{k,t_1} is given by integrating Equation (2.3), i.e.

$$I_{k,t_1} = \int_{t_{k-1}}^{t_k} f_{Z_{t_1}}^k(z) dz.$$

For a screening program consisting of K screens and participants age at first screen ranging between $t_{\min}, \dots, t_{\max}$, Wu et al. [16] used the count data $(n_{k,t_1}, s_{k,t_1}, r_{k,t_1})$ to form a likelihood function similar to a multinomial distribution, where n_{k,t_1} is the number of participants in screen k who were aged t_1 at program entry, s_{k,t_1} is the number of screen detected cases on screen k from those aged t_1 at screen entry and r_{k,t_1} is the number of symptomatic cases in the k^{th} screening interval from those aged t_1 at program entry. The likelihood is of the form:

$$L_1 = \prod_{t_1=t_{\min}}^{t_{\max}} \prod_{k=1}^K I_{k,t_1}^{r_{k,t_1}} D_{k,t_1}^{s_{k,t_1}} (1 - D_{k,t_1} - I_{k,t_1})^{n_{k,t_1} - s_{k,t_1} - r_{k,t_1}}.$$

Our important suggestion is that we propose to incorporate the exact dates of diagnosis of symptomatic patients (z_i) in the likelihood function if they are available, as they carry important information. Then the likelihood is of the form:

$$L_2 = \prod_{t_1=t_{\min}}^{t_{\max}} \prod_{k=1}^K \left[D_{k,t_1}^{s_{k,t_1}} (1 - D_{k,t_1} - I_{k,t_1})^{n_{k,t_1} - s_{k,t_1} - r_{k,t_1}} \prod_{i=1}^{r_{k,t_1}} f_{Z_{t_1}}^k(z_i) \right].$$

After specifying the parametric distributions of the preclinical and the sojourn time, we obtain the maximum likelihood estimates through nonlinear minimization of the negative log-likelihood. The variances of the parameter estimators can be approximated using the observed Fisher information matrix. We expect this to be more accurate for larger sample sizes.

3. Simulation setup

In order to investigate the identifiability of the parameters, we simulated disease progression data mimicking a breast cancer screening program using different onset and sojourn time distributions. The aims are: checking the identifiability of the parameters, examining the improvement in the model performance if the exact date of the diagnosis of symptomatic cases is incorporated, studying the effect of the length of the inter-screening time and see the effects of incorrect specifications, namely if the sensitivity is falsely assumed constant or if the sojourn time distribution is misspecified.

Breast cancer's screening sensitivity is known to be increasing with age ([11]). Wu et al [16] choose to model this age-dependence via a logistic function with parameters b_0 and b_1 . As a result, the sensitivity at age t is then given by:

$$\xi(t) = \frac{1}{\exp(-b_0 - b_1(t - \bar{t}))}.$$

Adopting the age-dependent sensitivity, we also chose the lognormal distribution $LN(\mu, s^2)$ for the onset time. This is realistic since the transition probabilities of breast cancer to the preclinical state were estimated by Lee and Zelen [8] using age-specific incidence rates. Wu et al [16] plotted the probabilities and found them to be right skewed with a heavy tail, so the lognormal distribution was chosen for having similar properties. It is also noted that the estimates of the onset distribution parameters may depend on the model choice, so different estimates exist [9]. We simulated our data using $\mu = 3.971$ and $s = 2.267$, these values lead to an average age of transition of around 54 years and a standard deviation of 15 years.

Using a lognormal preclinical intensity and an age-dependent sensitivity, we simulate progression data based on an exponential sojourn time with $\lambda = 1/2.5$ and a gamma sojourn time with shape $\alpha = 6.25$ and rate $\beta = 2.5$ both resulting in a mean sojourn time of 2.5 years and a unit variance (gamma case). For $t_1 = 40, \dots, 65$ years, we simulate 2 data sets for each distribution, one with $N_{1,t_1} = 10\,000$ and the other of size $N_{1,t_1} = 100\,000$ in each cohort, this would help us study the asymptotic performance of the model. The model is then run on each of the data sets, with and without including the exact date of diagnosis.

We also run a simplified simulation mimicking the model used by Duffy et al [3] in which both the onset and the sojourn times are exponentially distributed with parameters λ_1 and λ_2 while assuming that the sensitivity is constant. From a mathematical point of view, this model is interesting as the natural progression (without screening) in the chain is time homogeneous. The defined parameters in the simulations are $\xi = 0.75$, $\lambda_1 = 1/55$ and $\lambda_2 = 1/2.5$, resulting in an average onset age of 55 years and a mean sojourn time of 2.5 years. We also simulate two datasets ($N_{1,t_1} = 10\,000$ and $N_{1,t_1} = 100\,000$).

In order to test the goodness of fit, we will use Pearson's chi-squared test, which measures the distance between the observed and the expected counts. However, the chi-squared distance is just an illustrative measure, used only for showing the magnitude of the differences. Asymptotically, this distance is χ^2 distributed with $K \cdot (t_{\max} - t_{\min}) - v$ degrees of freedom, where v is the number of parameters, but we experienced large deviations for the small sample sizes due to the large number of classes.

In order to establish confidence regions for the parameters, we will use the likelihood ratio statistic, which assesses the goodness of fit of two competing statistical models based on the ratio of their likelihoods. The likelihood ratio statistic can be expressed as a function of the difference between the loglikelihoods $LR = 2(l(\hat{\theta}) - l(\theta))$, where $l(\hat{\theta})$ is the value of the loglikelihood at the maximum.

The finite sample distributions of likelihood-ratio tests are generally unknown. However, under the null hypothesis ($\theta = \theta_0$), LR converges in distribution to a χ^2 -distribution (by Wilks' theorem [15]). That allows defining the asymptotic confidence region $C(\theta)$ as:

$$C(\theta) = \{\theta : 2(l(\hat{\theta}) - l(\theta)) < \chi_{0.95}^2(v)\}.$$

The simulation and the nonlinear minimization of the negative loglikelihood are carried out using the statistical software **R**. However, since the integrals in

Equation (2.1) and (2.2) usually do not have a closed form, the integration has to be carried out numerically. This can be computationally expensive, especially as we include the date of diagnosis. For that reason, we use the package **Rcpp** [4] to carry out the numerical integration in **C++** using the package **CUBA** by Hahn [5]. The negative of the log likelihood is then minimized using the *optim* function, that is carried out using the “L-BFGS-B” algorithm [1]. In each scenario, we present the actual parameters, the estimates based on the count and full models for both data sets, the negative loglikelihood at the maximum (L_{\max}) and the likelihood based on the actual parameters (L_{actual}).

4. Results

4.1. Exponential sojourn time

4.1.1. Exponentially distributed onset

Let us start with the results of the simplest case, in which a constant sensitivity is assumed along with exponential X and Y . The results are presented in Table 1, it is clear that the model does not perform well for a small sample size.

In the first block of Table 1, the results for the small data set are presented, we noticed that the estimates for both the count based and full model are similar but not accurate at all. The sensitivity and the average onset age are highly overestimated and the mean sojourn time is underestimated.

Table 1. Estimates of the sensitivity, onset and sojourn time parameters for exponentially distributed X and Y .

		-Loglikelihood		Sensitivity	Onset	Sojourn time
		Maximum	Actual	ξ	$1/\lambda_1$	$1/\lambda_2$
Actual				0.75	55	2.5
$\Delta = 1$ $N_{t_1} = 10\ 000$	Count data	27 957.9	27 973.9	0.964	67.694	2.110
	Full data	27 950.4	27 975.9	1.000	59.443	2.081
$\Delta = 1$ $N_{t_1} = 100\ 000$	Count data	240 181.8	240 184.2	0.779	54.408	2.431
	Full data	239 864.6	239 866.7	0.778	54.381	2.431

Increasing the sample size to $N_{t_1} = 100\ 000$ (second block of Table 1), we observed a significant improvement in the accuracy of the models. The results of the count based model and the full model are almost identical, estimates for $1/\lambda_1$ and $1/\lambda_2$ are accurate.

When we studied the profile likelihood of the onset, it became clear that multiple parameters can maximize the likelihood and that the confidence region is vast. This can be seen in Figure 2, where we plot the negative loglikelihood fixing ξ and λ_2 to the estimated values and varying λ_1 . The confidence region for the average onset age $1/\lambda_2$ is [49.31;71.67] for the small data set and [48.7;61.07] for the large one.

The reason behind this large region is the exponential onset, that is very dense near 0 and decays quickly. Since we only start observing patients older than $t_{\min}=40$ years old and follow them up for 10 years, there is no information about the densest interval $(0, t_{\min})$, it is difficult for the model to estimate λ_1 for a small sample size. The dissimilarity to the actual density within the observation period is not detectable. Increasing the sample size allows better estimation of the parameters although the confidence region is still sizable. In this scenario, one can think of the disease progression as a flow process with the parameters controlling the rate of flow between states, accordingly, there are different flow rates which generate the same output.

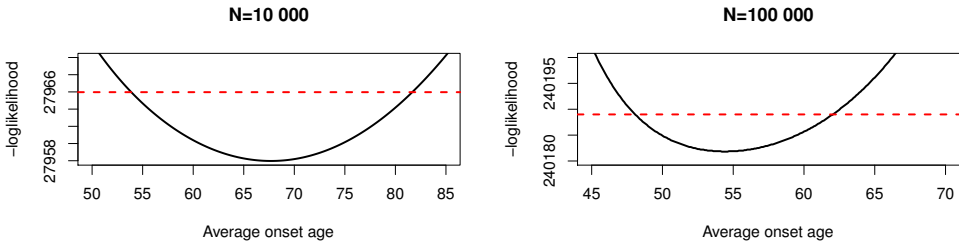


Figure 2. Profile likelihood of the average onset for the small data (left) and the large one (right), the red line is the critical threshold for the likelihood based confidence region.

Another observation is the very strong negative correlation between the sensitivity and the sojourn time estimators (the correlation measured using the observed Fisher information matrix between b_0 and ξ is around -0.8). Although they are assumed independent in the model, screening acts as a censoring mechanism, once a case is detected, the rest of its sojourn time cannot be observed. What happens then is that the model preserves a good fit in one of two ways, the first is by returning a high sensitivity estimate and a low sojourn time meaning that cases stay a short time in the preclinical state but participation in a screen leads to detection with a high probability. The second is by combining a high sojourn time estimate with a low sensitivity, meaning that cases will stay for a longer time in the preclinical state, therefore having multiple chances to participate in a screen, with screens having a low probability of detection. We observed this negative correlation in all of our parameterizations.

4.1.2. Lognormal onset

The results for a lognormally distributed onset time and an exponentially distributed sojourn time are presented in Table 2, plots for the sensitivity and the sojourn time are presented in the top part of Figure 3.

For ($N_{t_1} = 10\,000$), the sensitivity and onset parameters are substantially biased when using the count data, while using the full model results in more accurate estimates. Increasing the number of participants to 100 000 (second block), re

noticed a slight improvement in the performance of the count based model and a significant improvement when using the full model.

Table 2. Estimates of the sensitivity (b_0, b_1) , onset (μ, s) and sojourn time (λ) parameters for a lognormal X and an exponential Y .

		-Loglikelihood		Sensitivity		Preclinical intensity		Sojourn time
		Maximum	Actual	b_0	b_1	μ	s	λ
Actual				1.4	0.05	3.971	0.267	2.5
$\Delta = 1$	Count data	71 777.0	71 786.6	1.971	0.081	3.969	0.253	2.236
$N_{t_1} = 10\ 000$	Full data	71 647.6	71 652.7	1.529	0.061	3.969	0.257	2.423
$\Delta = 1$	Count data	712 825.7	712 851.5	1.540	0.050	3.972	0.260	2.428
$N_{t_1} = 100\ 000$	Full data	711 386.6	711 408.4	1.437	0.047	3.972	0.261	2.486

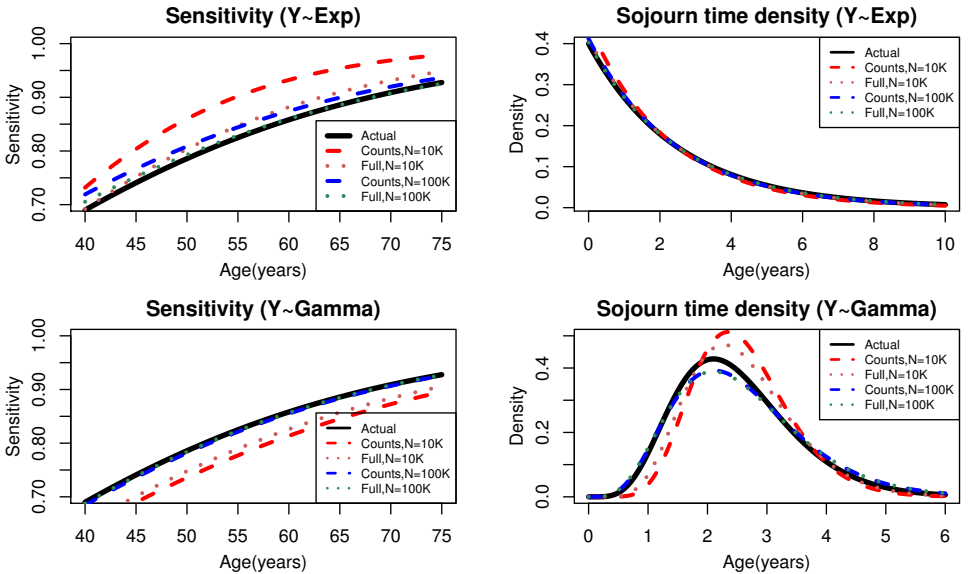


Figure 3. Sensitivity and the sojourn time density for lognormal X and exponential Y (top), lognormal X and gamma Y (bottom).

In order to evaluate the performance of the model and create reliable confidence intervals for the estimators for the small dataset, we ran the simulator 50 times and estimated the parameters based on both models. We also calculated the likelihood based confidence regions. The resulting confidence intervals are displayed in Table 3. We noticed that the intervals based on the full model are tighter than those of the count based ones. Besides, the likelihood-based confidence intervals for the sensitivity parameters b_0 and b_1 are larger than those based on the simulation. That is not the situation for the mean sojourn time intervals, where the likelihood-based intervals are tighter. The strong negative correlation between the sojourn time and the sensitivity creates a multi-centered confidence region for the

sojourn time. Since the likelihood based intervals are built around one center, they appear tighter than they actually are.

Table 3. Likelihood based and simulation based confidence intervals for the count based and the full models.

	Count based model		Full model	
	Simulations	Likelihood	Simulations	Likelihood
b_0	[1.425; 1.925]	[1.612; 2.418]	[1.346; 1.783]	[1.296; 1.786]
b_1	[0.0294; 0.0808]	[0.0219; 0.134]	[0.0314; 0.0672]	[0.0221; 0.101]
$1/\lambda$	[2.200; 2.663]	[2.095; 2.392]	[2.298; 2.590]	[2.185; 2.302]

4.2. Gamma sojourn time

For a lognormal onset and a gamma distributed sojourn time, the estimates are presented in Table 4, plots for the sensitivity and the sojourn time are shown in the bottom part of Figure 3.

For the small data set, the sensitivity estimates are biased under both models (bottom left part of Figure 3). Besides, estimates of the sojourn time parameters for both models are strange at first glance. However, these parameters result in acceptable estimates of the mean sojourn time, although the sojourn time variance is substantially underestimated in both cases.

Table 4. Estimates of the sensitivity (b_0, b_1), onset (μ, s) and sojourn time (λ) parameters for a lognormal X and gamma Y .

	Actual	-Loglikelihood		Sensitivity		Preclinical	intensity	Sojourn time		E(Y)	V(Y)
		Maximum	Actual	b_0	b_1	μ	s	α	β		
$\Delta = 1$	Count data	68 069.8	68 077.4	1.114	0.045	3.970	0.261	10.252	3.940	2.602	0.661
$N_{t_1} = 10\ 000$	Full data	68 040.3	68 047.3	1.181	0.047	3.970	0.261	8.407	3.259	2.580	0.792
$\Delta = 1$	Count data	676 971.1	676 999.6	1.375	0.050	3.973	0.264	5.457	2.116	2.579	1.219
$N_{t_1} = 100\ 000$	Full data	676 564.9	676 595.3	1.388	0.050	3.973	0.264	5.344	2.072	2.579	1.244

Moving on to the second block (larger dataset), both models perform well and their results are very close. However, the estimates of the sojourn time variance are still biased, the plots in Figure 3 show that estimated sojourn time density based on the full model is very close to the actual one although a slight bias is observed.

In general, the model seems to perform well in this case, with some slight bias in the sensitivity and the variance of the sojourn time. However, to test the reliability of our confidence sets, we fixed all the parameters to their estimated values ($N_{t_1} = 100\ 000$) and calculated the profile likelihood for different α and β . The contour plot can be seen in Figure 4, which clearly shows that the likelihood-based confidence region (black region) contains a substantial part of the line $\alpha = 2.5\beta$.

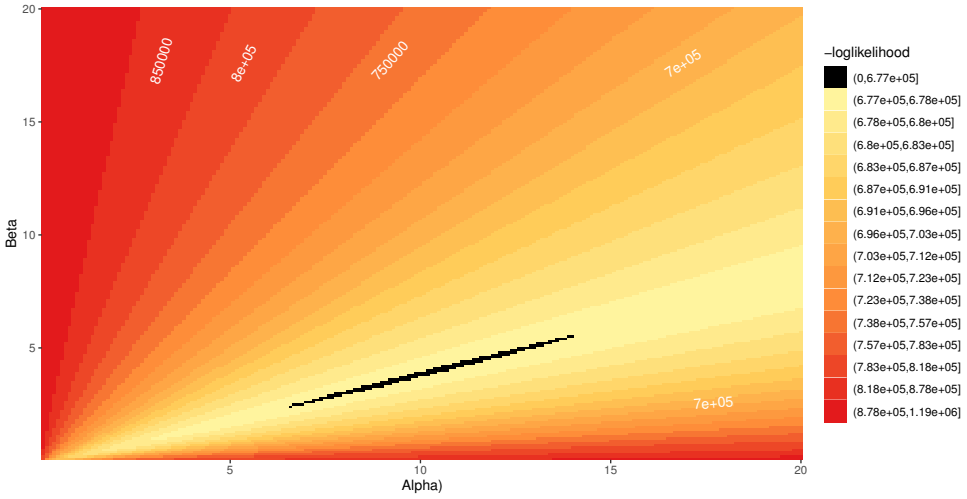


Figure 4. Contour plot of the loglikelihood ($N_{t_1} = 100000$) using the estimated sensitivity and onset parameters and varying α and β . The black region represents the likelihood based confidence region.

The figure shows that the likelihood is near constant in the neighborhood of the line $\alpha = 2.5\beta$. In this neighborhood, the expected value of the gamma distributed mean sojourn time is almost constant ($\alpha/\beta = 2.5$), however, there is a great variation of the variance (α/β^2), which does not affect the likelihood, this essentially means that the variance could be much larger or smaller and still fall in the likelihood based confidence region, with the noise in the data determining where the center of that region is. The model is then not able to estimate the variance of the sojourn time under this setup.

To further test the ability of the model to estimate the sojourn time variance, we generated a dataset ($N_{t_1} = 100\,000$, $K = 10$) based on $\alpha = 100$ and $\beta = 10$ resulting in a mean sojourn time of 10 years and a variance of 1. In this case, fitting a constant sojourn time of 10 years results in a $-\log$ likelihood of 928 675.1, almost identical to the $-\log$ likelihood based on the original parameters (928 674.3). This means that for large enough parameters α_0 and β_0 where $\alpha_0/\beta_0 = 10$, the model retains a good fit regardless of the variance, showing that there are infinitely many parameters maximizing the likelihood. Hence, the model is not able to estimate the variance of the sojourn time when Δ is too small since the tail of the sojourn time cannot be observed due to screening.

A larger inter-screening interval means that there are fewer opportunities for an individual to participate in a screening exam, thus it will lead to a high number of clinically detected (interval) cases and therefore more information about the sojourn time tail, we investigate this question in the next subsection.

4.3. Larger inter-screening time

In order to study the effect of a larger inter-screening time, we used the same disease progression data of size 100 000 and ran a screening program consisting of 5 screens with 2 years between each screen. The results are presented in Table 5. The models perform well in general, the estimates are more accurate than the case where the inter-screening time is one year.

Table 5. Estimates of the process governing parameters for an inter-screening time of 2 years.

		Exponential onset and sojourn time $\chi^2_{0.95}(247) = 284.66$									
		L_{max}	L_{actual}	ξ	$1/\lambda_1$	$1/\lambda_2$					
$N_{t_1} = 100\ 000$	$\Delta = 2$ Count data	215 680.9	215 681.9	0.751	55.948	2.532					
	Full data	222 598.4	222 599.3	0.752	55.972	2.528					
		Lognormal onset and exponential sojourn time $\chi^2_{0.95}(245) = 282.51$									
		L_{max}	L_{actual}	b_0	b_1	μ	s	$1/\lambda$			
$N_{t_1} = 100\ 000$	$\Delta = 2$ Count data	636 568.5	636 587.1	1.556	0.048	3.973	0.264	2.469			
	Full data	658 494.9	658 512.1	1.431	0.045	3.972	0.265	2.542			
		Lognormal onset and gamma sojourn time $\chi^2_{0.95}(244) = 281.44$									
		L_{max}	L_{actual}	b_0	b_1	μ	s	α	β	$E(Y)$	$V(Y)$
$N_{t_1} = 100\ 000$	$\Delta = 2$ Count data	60 993.9	62 151.2	1.403	0.055	3.973	0.267	5.282	2.047	2.58	1.261
	Full data	606 678.4	618 104.1	1.381	0.053	3.973	0.267	5.506	2.132	2.582	1.211

4.4. Misspecifications

Since it is not possible to observe neither the exact onset nor the sojourn time of breast cancer, there is a possibility that one may falsely assume the sensitivity to be constant or model the process with an incorrect distribution. In order to investigate the performance of the model under these false assumptions, we first force the sensitivity to be constant by fixing $b_1 = 0$.

To investigate the performance of the model when one misspecifies the distribution of the sojourn time, the model is run on the data generated by a known distribution, while using an incorrect distribution to model the sojourn time.

4.4.1. Constant sensitivity

Let us first use a constant sensitivity ($b_1 = 0$) and fit the count based and the full model on the data ($N_{t_1} = 100\ 000$ and $\Delta = 1$) generated by a lognormal onset combined with an exponential and gamma sojourn times. The results can be seen in Table 6.

It seems that forcing the sensitivity to be constant does not have a large effect on the estimates for an exponentially distributed sojourn time. However, the χ^2 -distance in both cases does not fall in the acceptance region.

In the second block of the table, where the sojourn time is gamma distributed, a bias can be observed in the variance of the sojourn time but the full model still performs quite well regardless of the false assumption. We also noticed that the

χ^2 -distance for both models is extremely large and is way outside the acceptance region.

Table 6. Estimates of the sensitivity (b_0), onset (μ, s) and sojourn time (λ) parameters when forcing a constant sensitivity ($b_1 = 0$).

		Distance	Sensitivity	Onset		Sojourn time			
		χ^2	b_0	μ	s	λ			
Actual			1.4	3.971	0.267	2.5			
Exponential	Count data	629.847	1.615	3.974	0.258	2.415			
	Full data	630.543	1.515	3.974	0.259	2.472			
		χ^2	b_0	μ	s	α	β	E(Y)	V(Y)
Actual			1.4	3.971	0.267	6.25	2.5	2.5	1
Gamma	Count data	1200.211	1.246	3.975	0.262	8.376	3.228	2.595	0.804
	Full data	1220.863	1.372	3.975	0.262	6.273	2.438	2.572	1.055

4.4.2. Incorrect distribution

In order to investigate the effect of misspecifying the sojourn time distribution, we used an exponential distribution to model the data generated by a gamma sojourn time ($\Delta = 1$ and $N_{t_1} = 100\ 000$). The results are displayed in Table 7.

Table 7. Estimates of the key parameters when misspecifying the sojourn time distribution.

Fitting an exponential sojourn time $\chi^2_{0.95}(497) = 549.97$							
$Y \sim$		χ^2	b_0	b_1	μ	s	$1/\lambda$
Gamma	Count data	8946.79	2.546	0.154	4.006	0.296	3.632
	Full data	9048.17	2.093	0.119	4.008	0.3	3.827

When an exponential distribution is fitted to data generated by a gamma sojourn time, the model does not perform well, the χ^2 -distances are enormous for both models. The estimates for the mean sojourn time are very high and both sensitivity and preclinical intensity parameters are also highly overestimated. This is highly problematic as the exponential distribution is the most used one in the literature.

The likely reason behind the high mean sojourn time estimate is the inability of the exponential distribution, which is a one parameter family, to fit the shape of a two parameter distribution (gamma). We also noticed that in this case, the count based model has a slightly better mean sojourn time estimate than that of the full model, since adding the exact time of diagnosis forces the exponential distribution to fit a density with a peak leading to worse results. The multicorrelation explains the estimates for the sensitivity and the onset, the model adjusts by increasing the sensitivity of screens and onset age to compensate for the inability of the exponential distribution to fit the data.

5. Consequences for previous results

The estimates of the mean sojourn time and the sensitivity in some famous clinical trials are shown in Table 8. One immediately notices the discrepancy between the estimates, there are completely different estimates based on the same data set but using different assumptions. We aim to discuss the reasons behind this inconsistency.

Table 8. Sojourn time and sensitivity estimates (M: mammography, P: physical exam) for some clinical trials.

Trial	Mean sojourn time	Sensitivity
Health Insurance Plan of greater New York (HIP) [13]	2.5	M:0.39 P:0.47
Edinburgh [13]	4.3	M:0.63, P:0.40
Canadian National Breast Screening Study (CNBSS1) [13]	1.9	M:0.61, P:0.59
Canadian National Breast Screening Study (CNBSS2) [13]	3.1	M:0.66, P:0.39
Canadian National Breast Screening Study (CNBSS1) [2]	2.55	0.7
Canadian National Breast Screening Study (CNBSS2) [2]	3.15	0.77
Norwegian Breast Cancer Screening Program for the age group [50,59] [14]	6.1	0.58
Norwegian Breast Cancer Screening Program for the age group [60,69] [14]	7.9	0.73

Chen et al. [2] used a stable disease approach and used the gamma distribution to model the sojourn time of breast cancer. They applied their model on the CNBSS data. They modeled the 40–49-year-old and 50–59-year-old cohorts separately. The sensitivity is assumed to be constant, we have shown that forcing a constant sensitivity barely affects the rest of the parameters. However, assuming the onset to be independent of age is not likely to hold true.

In the approach used by Wu et al [16], they used constraints on the sojourn time, the preclinical intensity, as well as the sensitivity when maximizing the likelihood. In other words, they run MCMC simulation on a bounded area to find a maximum, which could force a convergence to a local maximum. They also introduced using a loglogistic sojourn time to model the sojourn time, which has similar shape to the lognormal distribution but has heavier tails, it also has desirable survival rate properties.

To check the model performance under a loglogistic sojourn time, we ran the simulator based on a scale $\alpha = 2.336$ and a shape $\beta = 4.951$, to generate a data set of size $N_{t_1} = 100\,000$, the defined values lead to a mean sojourn time of 2.5 years and unit variance. After running the count based and the full model, we noticed that the estimates are generally accurate and the performance of the model is similar to the gamma sojourn time case. That being said, the variance of the sojourn time is also hard to estimate in this case.

Furthermore, we also used the loglogistic sojourn time to model the data based on the exponential and gamma distributions. For the exponential data, the count based model performed well, with acceptable estimates. However, the full model fails to estimate the parameters (estimated mean sojourn time of 3.41 years), this is caused by the inability of the loglogistic distribution to fit the exponential shape.

On the other hand, when fitting the model to data generated by a gamma sojourn time, the results are almost indistinguishable to actually fitting a gamma sojourn time. Even the likelihood based on the full models are almost identical, with a -loglikelihood of 676 562.9 when fitting a gamma distribution and 676 564.9 when fitting a loglogistic one. This means that there is no difference between the fit of the two distributions and one is not able to differentiate between them.

Regarding the conflicting results of the CNBSS1 studies, [13] estimated the sensitivity for Mammography(M) and physical examination(P) independently, their mean sojourn time estimate for the CNBSS1 trial is 1.9 years, significantly lower than the estimate of [2] of 2.55 years, the multicorrelation and different sojourn time distributions is possibly the reason behind the difference in the estimates.

A two parameters (entry–exit) Markov chain model is used by [3], assuming that the incidence rate λ_1 and the rate of transition from the preclinical state to the clinical one λ_2 are both constants. When this method is applied to the data from the Swedish two-county study of breast cancer screening in the age group 70-74, the resulting estimate for the mean sojourn time is 2.3 years. Although the model is very flexible in the sense that symptomatic data is not needed, we have shown that the parameters are not identifiable in this setup.

Weedon-Fekjaer et al. used a weighted non-linear least-square regression estimates based on a three step Markov chain model, then performed sensitivity analysis to determine the possible impact of opportunistic screening between regular screening rounds. Mean sojourn time and sensitivity were estimated by non-linear least square regression, using number of cancer cases at screening and in the interval between screening examinations. Mean sojourn time was estimated as 6.1 (95% confidence interval [CI] 5.1-7.0) years for women aged 50-59 years, and 7.9 years (95% CI 6.0-7.9) years for those aged 60-69 years, sensitivity was estimated as 58% (95% CI 52-64 %) and 73 % (67-78 %), respectively. We suspect that the high sojourn time estimate is a consequence of the choice of the sojourn time distribution, as we have shown earlier, using the exponential distribution to model a sojourn time having a different distribution results in a very high sojourn time estimate. Their findings also suggest that sensitivity is lower than in other programs as well as a higher mean sojourn time, but we believe it to be a direct consequence of the correlation between the two parameters.

6. Summary

Summing up our findings, we can state that the current models are very sensitive to the underlying assumptions. One should take great care of using such an approach, and multiple trials with different models are needed before in order to get reliable results. One way to solve this problem might be to include more information in the model to stabilize the results such as tumor growth shape and tumor size [6, 7].

Under an exponential onset and sojourn time, the parameters are not identifiable for a small sample, the acceptance region is sizable and data before the

first screen is needed to stabilize the results. On the other hand, under a lognormal onset and an exponential sojourn time, the model performs much better and estimates are generally accurate. Overall, the model performs well in this case, we noticed that the full model performs much better than the count-based one. Nonetheless, it would be wiser to apply the gamma model for the sojourn time, as it is much more flexible and it can be reduced to the exponential distribution in case the shape estimate is close to one.

The performance of the model is satisfactory for a gamma sojourn time, however estimates of the variance of the sojourn time are quite biased. A larger inter-screening interval improves the variance estimate since it allows observing the tail of the sojourn time before censoring (screening). But of course medical considerations might be more important in practice.

We also observed a high correlation between the parameters under all parameterizations. Consequently, the obtained variances of the estimators are not as reliable as we might think. On the other hand, including the exact date of diagnosis leads to more accurate estimates for a small sample size and a more compact acceptance region. We recommend applying both the count and the full model, and if they give inconsistent results, then misspecification might be the reason for this. The likelihood based on the full model is much more sensitive to small shifts in the parameters, since it will be magnified through the product of the likelihood of symptomatic cases. That is not the case with the count-based model.

Higher inter-screening intervals result in less accurate estimates for the sensitivity but better sojourn time variance estimates. We also noticed that the χ^2 distance of the count-based model was always smaller than that of the full one, although the latter allows for better estimates. Since maximizing the count-based likelihood is equivalent to minimizing the χ^2 -distance, this is a sign of over-fitting. Nonetheless, the χ^2 -distance can serve as good indicator for misspecification or incorrect assumptions.

References

- [1] R. BYRD, P. LU, J. NOCEDAL, C. ZHU: *A limited memory algorithm for bound constrained optimization*, SIAM Journal of Scientific Computing 16 (1995), pp. 1190–1208, issn: 1064-8275, DOI: <https://doi.org/10.1137/0916069>.
- [2] Y. CHEN, G. BROCK, D. WU: *Estimating key parameters in periodic breast cancer screening—Application to the Canadian National Breast Screening Study data*, Cancer Epidemiology 34.4 (2010), pp. 429–433, issn: 1877-7821, DOI: <https://doi.org/10.1016/j.canep.2010.04.001>.
- [3] S. W. DUFFY, H.-H. CHEN, L. TABAR, N. E. DAY: *Estimation of mean sojourn time in breast cancer screening using a Markov chain model of both entry to and exit from the preclinical detectable phase*, Statistics in Medicine 14.14 (1995), pp. 1531–1543, DOI: <https://doi.org/10.1002/sim.4780141404>.
- [4] D. EDELBUETTEL, J. J. BALAMUTA: *Extending extitR with extitC++: A Brief Introduction to extitRcpp*, PeerJ Preprints 5 (2017), e3188v1, issn: 2167-9843, DOI: <https://doi.org/10.7287/peerj.preprints.3188v1>.

- [5] T. HAHN: *Cuba—a library for multidimensional numerical integration*, Computer Physics Communications 168.2 (2005), pp. 78–95, ISSN: 0010-4655, DOI: <https://doi.org/10.1016/j.cpc.2005.01.010>.
- [6] A. HIJAZY, A. ZEMPLÉNI: *Gamma Process-Based Models for Disease Progression*, Methodol Comput Appl Probab (2020), DOI: <https://doi.org/10.1007/s11009-020-09771-4>.
- [7] A. HIJAZY, A. ZEMPLÉNI: *Optimal inspection for randomly triggered hidden deterioration processes*, Quality and Reliability Engineering International (2020), DOI: <https://doi.org/10.1002/qre.2707>.
- [8] S. J. LEE, M. ZELEN: *Scheduling Periodic Examinations for the Early Detection of Disease: Applications to Breast Cancer*, Journal of the American Statistical Association 93.444 (1998), pp. 1271–1281, DOI: <https://doi.org/10.1080/01621459.1998.10473788>.
- [9] G. PARMIGIANI, S. SKATES: *Estimating distribution of age of the onset of detectable asymptomatic cancer*, Mathematical and Computer Modelling 33.12 (2001), pp. 1347–1360, ISSN: 0895-7177, DOI: [https://doi.org/10.1016/S0895-7177\(00\)00320-4](https://doi.org/10.1016/S0895-7177(00)00320-4).
- [10] P. C. PROROK: *The theory of periodic screening II: doubly bounded recurrence times and mean lead time and detection probability estimation*, Advances in Applied Probability 8.3 (1976), pp. 460–476, DOI: <https://doi.org/10.2307/1426139>.
- [11] S. P. SHAPIRO S. VENET W., V. L.: *Periodic Screening for Breast Cancer. The Health Insurance Plan Project, 1963–1986, and its Sequelae*, 1988.
- [12] Y. SHEN, M. ZELEN: *Parametric estimation procedures for screening programmes: Stable and non stable disease models for multimodality case findings*, Biometrika 86.3 (1999), pp. 503–515.
- [13] Y. SHEN, M. ZELEN: *Screening Sensitivity and Sojourn Time From Breast Cancer Early Detection Clinical Trials: Mammograms and Physical Examinations*, Journal of Clinical Oncology 19.15 (2001), pp. 3490–3499, DOI: <https://doi.org/10.1200/JCO.2001.19.15.3490>.
- [14] H. WEEDON-FEKJÆR, L. J. VATTEN, O. O. AALEN, B. LINDQVIST, S. TRETTLI: *Estimating mean sojourn time and screening test sensitivity in breast cancer mammography screening: new results*, Journal of Medical Screening 12.4 (2005), pp. 172–178, DOI: <https://doi.org/10.1258/096914105775220732>.
- [15] S. S. WILKS: *The Large-Sample Distribution of the Likelihood Ratio for Testing Composite Hypotheses*, Ann. Math. Statist. 9.1 (1938), pp. 60–62, DOI: <https://doi.org/10.1214/aoms/1177732360>.
- [16] D. WU, G. L. ROSNER, L. BROEMELING: *MLE and Bayesian Inference of Age-Dependent Sensitivity and Transition Probability in Periodic Screening*, Biometrics 61.4 (2005), pp. 1056–1063, DOI: <https://doi.org/10.1111/j.1541-0420.2005.00361.x>.
- [17] M. ZELEN, M. FEINLEIB: *On the theory of screening for chronic diseases*, Biometrika 56.3 (1969), pp. 601–614, DOI: <https://doi.org/10.1093/biomet/56.3.601>.

Dealing with uncertainty: A rough-set-based approach with the background of classical logic*

Tamás Kádek, Tamás Mihálydeák

University of Debrecen, Faculty of Informatics

tamas.kadek@inf.unideb.hu

mihalydeak@unideb.hu

Submitted: December 22, 2020

Accepted: February 17, 2021

Published online: May 18, 2021

Abstract

The representative-based approximation has been widely studied in rough set theory. Hence, rough set approximations can be defined by the system of representatives, which plays a crucial role in set approximation. In the authors' previous research a possible use of the similarity-based rough set in first-order logic was investigated. Now our focus has changed to representative-based approximation systems. In this article the authors show a logical system relying on representative-based set approximation. In our approach a three-valued partial logic system is introduced. Based on the properties of the approximation space, our theorems prove that in some cases, there exists an efficient way to evaluate the first-order formulae.

Keywords: Rough set theory, set approximation, approximation-based logic system

AMS Subject Classification: 03E72

1. Introduction

Nowadays a huge amount of data appear in an information system and they have to be treated in order to get new information, to make decisions, etc. Behind the

*This work was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund.

data there are objects with (probably different) properties. Properties are handled in two steps: as attributes and the corresponding attribute values. In the real practice finite number of attributes and that of the corresponding attribute values can be used. Usually, there are more objects than combination of attribute values, therefore more than one objects are represented by the same attribute values, and so they are indiscernible relying on the background knowledge embedded in an information system. Indiscernible objects have to be treated in the same way.

Pawlak's original system of rough sets shows the consequences of indiscernibility [10–12]. In many practical cases not only indiscernible objects have to be treated in the same way, but objects with the same attribute values of some (and not all) attributes. This is one of the theoretical bases of the generalizations of Pawlak's original theory. In rough sets theory the objects to be treated in the same way belong to a base set. Informally in granular computing a granule contains objects which have to be treated in the same way. Granules play – as the most fundamental concept – a crucial role in granular computing, it means that granules (and not objects belonging to them) are in the focus of investigations.

Representatives are used for representing a whole group of objects. In a very general case to choose representatives (granules) is not a trivial problem. In the case of a system relying on an indiscernible relation, any object can represent the corresponding indiscernible set of objects. When a tolerance relation is used, then the method of correlation clustering gives a possibility to define representatives (see [1, 8]). Based on the different techniques to find representatives some generalization of the approximation space must be considered.

From the logical point of view, a natural question arises: is there any possibility to create a first-order logical system relying on representatives? If the answer is yes, then the consequence relation can be used in order to get (or check) new information. In this paper the authors define first-order logical semantics and show some of its important properties.

Logical systems based on rough sets are also widely studied [9], so it seems easy to predict the results, but the investigation should repeat when a new viewpoint appears. In this work we will use the most recent general definition of representative-based approximation space. The main goal is to define a logical system which uses only the representatives when a decision about a certain group of objects is made.

The structure of the paper is the following: at first, we will define the representative-based approximation system, where instead of base sets the extension of representatives is used. Then a one-argument first-order language is introduced with approximation-space-based semantics. We will show how to generate approximative interpretations from an existing classical one. Finally, the key properties of our system will be discovered with the help of a few theorems.

2. Representative-based approximation spaces

Definition 2.1. The triple $\langle U, R, \mathfrak{A} \rangle$ is a *representative-based approximation space* if

1. U is a nonempty set of objects,
2. $R = \{r_1, r_2, \dots, r_k\}$ where $k \geq 1$ is a set of representatives,
3. $\mathfrak{R} \subseteq R \times U$ is a relation.

Definition 2.2. Let r_i be a representative, i.e. $r_i \in R$. Then

$$\langle\langle r_i \rangle\rangle^{\langle U, R, \mathfrak{R} \rangle} = \{u : r_i \mathfrak{R} u\}$$

is the *extension* of r_i . We shortly will write $\langle\langle r_i \rangle\rangle$ if it does not cause any misunderstanding.

There exists a general agreement to restrict $\langle\langle r_i \rangle\rangle$ at least saying that it shall not be empty, but this constraint is now unnecessary. Although is straightforward that a representative with empty extension can not be useful during the approximation.

Definition 2.3. The *approximation pair* $\langle \iota, \mathbf{u} \rangle$ of the representative-based approximation space $\langle U, R, \mathfrak{R} \rangle$ is a pair of mappings $2^U \rightarrow 2^U$ defined as follows

$$\begin{aligned} \iota(S) &= \cup \{ \langle\langle r_i \rangle\rangle : r_i \in R \text{ and } \langle\langle r_i \rangle\rangle \subseteq S \}; \\ \mathbf{u}(S) &= \cup \{ \langle\langle r_i \rangle\rangle : r_i \in R \text{ and } \langle\langle r_i \rangle\rangle \cap S \neq \emptyset \}. \end{aligned}$$

From this point, the $\langle\langle r_i \rangle\rangle$ extensions of the representatives can be considered as base sets of a union-type approximation space [2].

Definition 2.4. Let $\langle U, R, \mathfrak{R} \rangle$ be a representative-based approximation space and $u \in U$. Then the *representative vector* of u (denoted by $[u]^{\langle U, R, \mathfrak{R} \rangle}$ or simply $[u]$ if it does not cause any misunderstanding) is the following:

$$\begin{aligned} [u]^{\langle U, R, \mathfrak{R} \rangle} &= \left\langle [u]_1^{\langle U, R, \mathfrak{R} \rangle}, \dots, [u]_k^{\langle U, R, \mathfrak{R} \rangle} \right\rangle \text{ where} \\ [u]_i^{\langle U, R, \mathfrak{R} \rangle} &= \begin{cases} 1 & \text{if } u \in \langle\langle r_i \rangle\rangle, \\ 0 & \text{otherwise,} \end{cases} \quad i = 1, \dots, k. \end{aligned}$$

Some common properties of the approximation space can be determined by analyzing the representative vectors:

$$\sigma(u) = \sum_{i=1}^k [u]_i$$

- if $\sigma(u) = 1$ for all $u \in U$, then the approximation space is based on a partition generated by \mathfrak{R} ;
- if $\sigma(u) = 0$ for some $u \in U$, then the approximation space is partial, because u is an object without any representative, and so $u \notin \iota(S)$ and $u \notin \mathbf{u}(S)$ for all $S \subseteq U$;
- if $\sigma(u) \geq 2$ for some $u \in U$, then the approximation space contains overlapping (not disjoint) extensions for some representatives. See more about covering systems relying on tolerance relations in [14] and about general covering systems in [13, 15].

3. One-argument first-order language

We begin the investigation with a simplified first-order language which allows one-argument predicate parameters only. The simplified language could be easily extended with other predicate parameters [7], and it is expressive enough for further investigations [3].

Definition 3.1. The ordered 4-tuple $\langle LC, Var, Pred, Form \rangle$ is a *one-argument first-order language* containing only one-argument predicate parameters if

1. $LC = \{\neg, \wedge, \vee, \supset, \exists, \forall, (,)\}$ is the set of logical constants;
2. $Var = \{x_1, x_2, \dots\}$ is a countably infinite set of variables;
3. $Pred$ is a nonempty set of one-argument predicate parameters;
4. LC, Var , and $Pred$ are pairwise disjoint;
5. the *set of formulae* denoted by $Form$ is defined inductively:
 - (a) if $P \in Pred$ and $x \in Var$, then $P(x) \in Form$ and is an *atomic formula*,
 - (b) if $A \in Form$, then $\neg A \in Form$,
 - (c) if $A, B \in Form$ and $\circ \in \{\wedge, \vee, \supset\}$, then $(A \circ B) \in Form$,
 - (d) if $A \in Form$ and $x \in Var$, then $\exists x A \in Form$ and $\forall x A \in Form$.

3.1. Interpretation

The conventional Aristotelian semantics of a one-argument first-order language is very widely known, hence it is not introduced here, only the interpretation of the language is recalled.

Definition 3.2. The pair $\langle U, \psi \rangle$ is an *interpretation* of the one-argument first-order language $\langle LC, Var, Pred, Form \rangle$ if

1. U is a nonempty set of objects,
2. ψ is a mapping $Pred \rightarrow 2^U$.

In the classical first-order logic, if $\langle U, \psi \rangle$ is an interpretation on a given U set of objects, and P is a one-argument predicate parameter of the language, then the semantic value of P is usually given as $\psi(P) \subseteq U$:

- $u \in \psi(P)$ means that u belongs to the positivity domain of P , or we can say that P is true on u ,
- $u \in U \setminus \psi(P)$ means that u belongs to the negativity domain of P , or we can say that P is false on u .

Next, we define the semantics of a one-argument first-order language with the help of a representative-based approximation space. The idea is to approximate the positivity and negativity domains adapting the solution explained in [6]. To do so, first we introduce the representative-based approximative interpretation.

Definition 3.3. The ordered 4-tuple $\langle U, R, \mathfrak{R}, \varrho \rangle$ is an *approximative interpretation* of the one-argument first-order language $\langle LC, Var, Pred, Form \rangle$ if

1. $\langle U, R, \mathfrak{R} \rangle$ is a representative-based approximation space,
2. ϱ is a mapping such that $\varrho(P) = \langle \varrho(P)_1, \dots, \varrho(P)_k \rangle$ for all $P \in Pred$, where
 - (a) $\varrho(P)_\ell \in \{-1, 0, 1\}$ ($\ell = 1, \dots, k$); and
 - (b) there is no $u \in U$ and $i, j \in \{1, \dots, k\}$ such that

$$[u]_i \cdot \varrho(P)_i = 1 \text{ and } [u]_j \cdot \varrho(P)_j = -1;$$

where k is the number of representatives, hence $R = \{r_1, \dots, r_k\}$.

The $\varrho(P)_i$ represents the relationship between the i th representative (r_i) and the semantic value of the one-argument predicate P :

- if $\varrho(P)_i = +1$, then r_i certainly belongs to the positivity domain of P ;
- if $\varrho(P)_i = -1$, then r_i certainly belongs to the negativity domain of P ;
- if $\varrho(P)_i = 0$, then we cannot decide whether r_i belongs to the positivity domain or not. We could say that r_i is in the boundary region.

The arithmetic product $[u]_i \cdot \varrho(P)_i$ is used to express the connection between an arbitrary object $u \in U$ and the semantic value of P with the help of the i th representative. Our definition excludes the contradiction when different representatives of u belong certainly to the positivity and negativity domain of P . Now we show a method to satisfy this condition with the help of an interpretation.

Definition 3.4. Let $\langle U, R, \mathfrak{R} \rangle$ be a representative-based approximation space, \mathcal{L} be a one-argument first-order language, and $\langle U, \psi \rangle$ be its interpretation. The

$$\varrho(P)_i = \begin{cases} 1 & \text{if } \langle\langle r_i \rangle\rangle \subseteq \psi(P), \\ -1 & \text{if } \langle\langle r_i \rangle\rangle \cap \psi(P) = \emptyset, \\ 0 & \text{otherwise;} \end{cases}$$

function is the *derived mapping* from ψ with respect to a given $\langle U, R, \mathfrak{R} \rangle$.

Theorem 3.5. Let $\langle U, R, \mathfrak{R} \rangle$ be a representative-based approximation space, \mathcal{L} be a one-argument first-order language, and $\langle U, \psi \rangle$ be its interpretation. If ϱ is the derived mapping from ψ with respect to $\langle U, R, \mathfrak{R} \rangle$, then there is no $u \in U$ and $i, j \in \{1, \dots, k\}$ such that $[u]_i \cdot \varrho(P)_i = 1$ and $[u]_j \cdot \varrho(P)_j = -1$.

Proof. If $[u]_i \cdot \varrho(P)_i = 1$ for some $u \in U$ and $i \in \{1, \dots, k\}$, then both $[u]_i = 1$ and $\varrho(P)_i = 1$. By definition, $[u]_i = 1$ when $u \in \langle\langle r_i \rangle\rangle$ and $\varrho(P)_i = 1$ when $\langle\langle r_i \rangle\rangle \subseteq \psi(P)$, so $u \in \psi(P)$. Indirectly supposing that there exists a $j \in \{1, \dots, k\}$ such that $[u]_j \cdot \varrho(P)_j = -1$, the following contradiction appears: $[u]_j = 1$, so $u \in \langle\langle r_j \rangle\rangle$, which means that $\langle\langle r_j \rangle\rangle \cap \psi(P) \neq \emptyset$, but $\varrho(P)_j = -1$, hence the previous intersection should be empty. \square

Corollary 3.6. *Let $\langle U, R, \mathfrak{R} \rangle$ be a representative-based approximation space, \mathcal{L} be a one-argument first-order language, $\langle U, \psi \rangle$ be its interpretation, and ϱ be the derived mapping from ψ . Then $\langle U, R, \mathfrak{R}, \varrho \rangle$ is an approximative interpretation.*

The value of $\varrho(P)_i$ – if it is derived from the $\langle U, \psi \rangle$ interpretation – shows the relationship between the positivity domain of P and extension $\langle\langle r_i \rangle\rangle$ of the i th representative:

- If $\varrho(P)_i = 1$, then all members of the extension of r_i (all objects represented by r_i) are in the positivity domain of P ; P is certainly true for all $u \in \langle\langle r_i \rangle\rangle$.
- If $\varrho(P)_i = -1$, then all members of the extension of r_i are in the negativity domain of P ; P is certainly false for all $u \in \langle\langle r_i \rangle\rangle$.
- If $\varrho(P)_i = 0$, then some members of the extension of r_i belong to the positivity domain, while others belong to the negativity domain.

3.2. Semantics

A widely used technique in rough set theory is to distinguish between optimistic and pessimistic approaches [7]. At this point it is crucial to analyze the information about objects, especially in the case when different representatives declare different facts about the positivity and negativity domain of a predicate.

The tables in Fig. 1 summarize the difference of four approaches. The heads of the tables contain the maximum and the rows contain the minimum of the set:

$$\Delta(P, u) = \{ \varrho(P)_i : i \in \{1, \dots, k\}, [u]_i = 1 \}$$

The bottom left corners are empty hence this kind of contradiction was not allowed in Definition 3.3. If $\Delta \neq \emptyset$, when u has at least one representative, then the following approaches appear:

1. *Optimistic approach:* we take the maximum of $\Delta(P, u)$, so if there exists at least one representative of u that belongs to the positivity domain of P , we will suppose that P is true on u .
2. *Pessimistic approach:* we take the minimum of $\Delta(P, u)$, so we suppose that P is true on u only if all the representatives of u belong to the positivity domain of P .

3. *Union-based approach*: we say that u belongs to the union of its representatives. This implies that if at least one representative belongs to the border, then we cannot say anything certain about u .
4. *Intersection-based approach*: we say that u belongs to the intersection of its representatives. This implies that uncertainty will appear only if all the representatives of u belong to the border.

$\Delta(P, u)$	1	0	-1
1	1		
0	1	0	
-1		0	-1

Optimistic Approach

$\Delta(P, u)$	1	0	-1
1	1		
0	0	0	
-1		-1	-1

Pessimistic Approach

$\Delta(P, u)$	1	0	-1
1	1		
0	0	0	
-1		0	-1

Union-Based Approach

$\Delta(P, u)$	1	0	-1
1	1		
0	1	0	
-1		-1	-1

Intersection-Based Approach

Figure 1. Managing contradicting information.

By respecting the set theoretic view of the extension of representatives (introduced in Definition 2.2 and also used later in Definition 3.4), it is a straightforward decision to adopt the intersection-based approach.

Definition 3.7. Let $\langle U, R, \mathfrak{R}, \varrho \rangle$ be an approximative interpretation. The function $v : Var \rightarrow U$ is an *assignment* relying on the approximative interpretation.

Definition 3.8. Let v be an assignment relying on the $\langle U, R, \mathfrak{R}, \varrho \rangle$ approximative interpretation. The assignment $v[x:u]$ denotes a *modified assignment* which is defined as follows:

$$v[x:u](y) = \begin{cases} u & \text{if } y = x, \\ v(y) & \text{otherwise.} \end{cases}$$

Note that we defined the assignment and the modified assignment exactly in the same way as it was introduced in the classical first-order logic. It helps us to compare the evaluation method later.

Definition 3.9. The semantic value of $P \in Pred$ is the following

$$U \rightarrow \{0, \frac{1}{2}, 1\} \cup \{2\}$$

function:

$$\llbracket P \rrbracket^{\langle U, R, \mathfrak{R}, \varrho \rangle} (u) = \begin{cases} 2 & \text{if } [u]_i = 0 \text{ for all } i \in \{1, \dots, k\} \\ 1 & \text{if } [u]_i \cdot \varrho(P)_i = 1 \text{ for some } i \in \{1, \dots, k\} \\ 0 & \text{if } [u]_i \cdot \varrho(P)_i = -1 \text{ for some } i \in \{1, \dots, k\} \\ 1/2 & \text{otherwise.} \end{cases}$$

As a consequence of the system's possible partiality, logic with truth value gap is used. The value 2 represents the lack of truth value.

Theorem 3.10. *Let $\langle LC, Var, Pred, Form \rangle$ be a one-argument first-order language and $\langle U, R, \mathfrak{R}, \varrho \rangle$ be its approximative interpretation relying on the representative-based approximation space $\langle U, R, \mathfrak{R} \rangle$ where ϱ is the derived mapping from ψ ; then $\llbracket P \rrbracket (u) = 1$ if and only if $u \in \mathfrak{I}(\psi(P))$ for all $u \in U$.*

Proof. Let us create the proof in two steps:

1. If $\llbracket P \rrbracket (u) = 1$ then there exists an $r_i \in R$ such that $[u]_i \cdot \varrho(P)_i = 1$ and so $u \in \langle\langle r_i \rangle\rangle$ (based on Definition 2.4) and $\langle\langle r_i \rangle\rangle \subseteq \psi(P)$ (based on Definition 3.4). When $\langle\langle r_i \rangle\rangle \subseteq \psi(P)$ then $\langle\langle r_i \rangle\rangle \subseteq \mathfrak{I}(\psi(P))$ and so $u \in \mathfrak{I}(\psi(P))$.

2. If $u \in \mathfrak{I}(\psi(P))$ then there exists an r_i such that $u \in \langle\langle r_i \rangle\rangle$ and $\langle\langle r_i \rangle\rangle \subseteq \psi(P)$ and so $[u]_i = 1$ and $\varrho(P)_i = 1$ therefore $\llbracket P \rrbracket (u) = 1$ hence $[u]_i \cdot \varrho(P)_i = 1$. \square

The idea to use a partial three-valued system appeared in [4, 7].

Definition 3.11. The semantic value of the formula $A \in Form$ using the interpretation $\langle U, R, \mathfrak{R}, \varrho \rangle$ is denoted by $\llbracket A \rrbracket^{\langle U, R, \mathfrak{R}, \varrho \rangle}$ or simply $\llbracket A \rrbracket_v$ and defined as follows:

$$\begin{aligned} \llbracket P(x) \rrbracket_v &= \llbracket P \rrbracket (v(x)) \\ \llbracket \neg A \rrbracket_v &= \begin{cases} 2 & \text{if } \llbracket A \rrbracket_v = 2 \\ 1 - \llbracket A \rrbracket_v & \text{otherwise;} \end{cases} \\ \llbracket (A \wedge B) \rrbracket_v &= \begin{cases} 2 & \text{if } \llbracket A \rrbracket_v = 2 \text{ or } \llbracket B \rrbracket_v = 2 \\ \min\{\llbracket A \rrbracket_v, \llbracket B \rrbracket_v\} & \text{otherwise;} \end{cases} \\ \llbracket (A \vee B) \rrbracket_v &= \begin{cases} 2 & \text{if } \llbracket A \rrbracket_v = 2 \text{ or } \llbracket B \rrbracket_v = 2 \\ \max\{\llbracket A \rrbracket_v, \llbracket B \rrbracket_v\} & \text{otherwise;} \end{cases} \\ \llbracket (A \supset B) \rrbracket_v &= \begin{cases} 2 & \text{if } \llbracket A \rrbracket_v = 2 \text{ or } \llbracket B \rrbracket_v = 2 \\ \max\{1 - \llbracket A \rrbracket_v, \llbracket B \rrbracket_v\} & \text{otherwise;} \end{cases} \end{aligned}$$

Let $\mathcal{V} = \{u : u \in U \text{ and } \llbracket A \rrbracket_{v[x;u]} \neq 2\}$.

$$\llbracket \exists x A \rrbracket_v = \begin{cases} 2 & \text{if } \mathcal{V} = \emptyset, \\ \max_{u \in \mathcal{V}} \{ \llbracket A \rrbracket_{v[x;u]} \} & \text{otherwise;} \end{cases}$$

$$\llbracket \forall x A \rrbracket_v = \begin{cases} 2 & \text{if } \mathcal{V} = \emptyset, \\ \min_{u \in \mathcal{V}} \{ \llbracket A \rrbracket_{v[x:u]} \} & \text{otherwise;} \end{cases}$$

Like in the classical case, \exists and \forall quantifiers are defined as the generalizations of \vee and \wedge , respectively.

4. Key properties of the approximation

Theorem 4.1. *Let $\mathcal{L} = \langle LC, Var, Pred, Form \rangle$ be a one-argument first-order language and $I = \langle U, R, \mathfrak{R}, \varrho \rangle$ be an approximative interpretation of \mathcal{L} . There exists an approximative interpretation $J = \langle U', R, \mathfrak{R}', \varrho \rangle$ such that*

$$|U'| \leq 2^k \text{ and } \llbracket A \rrbracket_v^I = \llbracket A \rrbracket_w^J \text{ for all } A \in Form$$

where $w(x) = \tau(v(x))$ for some mapping $\tau : U \rightarrow U'$.

Proof. We present a construction for such an interpretation $J = \langle U', R, \mathfrak{R}', \varrho \rangle$ and mapping τ :

$$\tau(u) = \sum_{i=1}^k 2^{k-1} [u]_i^{\langle U, R, \mathfrak{R} \rangle}$$

$$U' = \{ \tau(u) : u \in U \}$$

$$\mathfrak{R}' = \{ \langle r_i, \tau(u) \rangle : \langle r_i, u \rangle \in \mathfrak{R} \}.$$

It is clear that $U' \subseteq \{0, 1, \dots, 2^k - 1\}$ so the cardinality condition of U' is satisfied. Because of the definition of the \mathfrak{R}' relation,

$$[u]_i^{\langle U, R, \mathfrak{R} \rangle} = [\tau(u)]_i^{\langle U', R, \mathfrak{R}' \rangle} \text{ therefore } \llbracket P \rrbracket^I(v(x)) = \llbracket P \rrbracket^J(w(x))$$

so the theorem is proved for atomic formulae and can be proved for arbitrary formulae with the help of structural induction. \square

Corollary 4.2. *During the evaluation process of a quantified formula, instead of using all members of the set U , it is enough to consider 2^k objects only. It can dramatically increase the speed of quantified formulae evaluation and so it can reduce the computation time.*

4.1. Properties of the approximation on covering systems

Theorem 4.3. *The one-argument first-order language $\langle LC, Var, Pred, Form \rangle$ and its approximative interpretation $\langle U, R, \mathfrak{R}, \varrho \rangle$ generate a three-valued logic system without truth value gap if $\langle U, R, \mathfrak{R} \rangle$ is a covering approximation space.*

Proof. In case of an arbitrary atomic formula $P(x)$ and an arbitrary approximative interpretation $\langle U, R, \mathfrak{R}, \varrho \rangle$, truth value gap (2) can appear as the semantic value of $\llbracket P(x) \rrbracket_v$ only if $\sigma(v(x)) = 0$, but in a covering approximation space $\sigma(u) > 0$ for all $u \in U$. So the theorem is proved for atomic formulae and can be proved for arbitrary formulae with the help of structural induction. \square

Theorem 4.4. *Let $\langle LC, Var, Pred, Form \rangle$ be a one-argument first-order language and $\langle U, R, \mathfrak{R}, \varrho \rangle$ be its approximative interpretation relying on the representative-based covering approximation space $\langle U, R, \mathfrak{R} \rangle$ where ϱ is the derived mapping from ψ , and let v be an arbitrary assignment.*

$$\text{If } \llbracket A \rrbracket_v^{\langle U, R, \mathfrak{R}, \varrho \rangle} \in \{0, 1\} \text{ then } \llbracket A \rrbracket_v^{\langle U, R, \mathfrak{R}, \varrho \rangle} = |A|_v^{\langle U, \psi \rangle}.$$

Proof. First we will show, that the statement is true for any arbitrary atomic formula:

- If $\llbracket P(x) \rrbracket_v = 1$ then $[v(x)]_i \cdot \varrho(P)_i = 1$ for some $i \in \{1, \dots, k\}$.
 - $v(x) \in \langle\langle r_i \rangle\rangle$ because of $[v(x)]_i = 1$
(as a consequence of Definition 2.4);
 - $\langle\langle r_i \rangle\rangle \subseteq \psi(P)$ hence $\varrho(P)_i = 1$ and ϱ is derived from ψ
(see Definition 3.4);

therefore $v(x) \in \psi(P)$ so $|P(x)|_v = 1$.

- If $\llbracket P(x) \rrbracket_v = 0$ then $[v(x)]_i \cdot \varrho(P)_i = -1$ for some $i \in \{1, \dots, k\}$.
 - $v(x) \in \langle\langle r_i \rangle\rangle$ because of $[v(x)]_i = 1$;
 - $\langle\langle r_i \rangle\rangle \cap \psi(P) = \emptyset$ hence $\varrho(P)_i = -1$ and ϱ is derived from ψ ;

therefore $v(x) \notin \psi(P)$ so $|P(x)|_v = 0$.

Now the theorem can be proved by using structural induction which is trivial in case of zero order connectives and very similar in case of the quantifiers, therefore we focused on the existentially quantified expressions only. Supposing that the theorem is true for the formula A :

- $\llbracket \exists x A \rrbracket_v = 1$ guarantees that there exists a $v[x:u]$ modified assignment so that $\llbracket A \rrbracket_{v[x:u]} = 1$. As we have above supposed, $|A|_{v[x:u]} = 1$ so $|\exists x A|_v = 1$.
- $\llbracket \exists x A \rrbracket_v = 0$ implies that $\llbracket A \rrbracket_{v[x:u]} = 0$ for all $v[x:u]$ modified assignment. Therefore $|A|_{v[x:u]} = 0$ for all $u \in U$ so $|\exists x A|_v = 0$. Remember that $\llbracket A \rrbracket_{v[x:u]} \neq 2$ if the approximative interpretation relies on a covering approximation space as it was proved in Theorem 4.3. \square

5. Conclusion and future work

In this article we have successfully shown a possible semantic background of a one-argument first-order logical system based on a representative-based approximation. An approximative interpretation was introduced, which we can derive from a classical first-order interpretation easily. We have compared the classical and the approximation-based evaluation, and we have found that, at least in the case of a covering approximation space, we can predict the semantic value of a formula by using the approximation.

Thanks to the promising results shown in the theorems, we have the theoretical basis for further investigations. One possible direction is to analyze the logical laws and inference schemes of the first-order logic in case of different granule systems. The investigation will follow the methods presented in [3, 5]. We hope that the result of the planned research could be a calculus over a three-valued partial system relying on the representative-based approximation space.

References

- [1] L. ASZALÓS, T. MIHÁLYDEÁK: *Rough Clustering Generated by Correlation Clustering*, in: *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, Springer Berlin Heidelberg, 2013, pp. 315–324, DOI: <https://doi.org/10.1109/TKDE.2007.1061>.
- [2] Z. E. CSAJBÓK, T. MIHÁLYDEÁK: *General set approximation and its logical applications*, in: Jan. 2015, pp. 33–40, DOI: <https://doi.org/10.14794/ICAI.9.2014.1.33>.
- [3] T. KÁDEK, T. MIHÁLYDEÁK: *On (in)Validity of Aristotle's Syllogisms Relying on Rough Sets*, *Annals of Computer Science and Information Systems* 7 (2015), pp. 35–40.
- [4] T. MIHÁLYDEÁK: *First-Order Logic Based on Set Approximation: A Partial Three-Valued Approach*, in: 2014 IEEE 44th International Symposium on Multiple-Valued Logic, May 2014, pp. 132–137, DOI: <https://doi.org/10.1109/ISMVL.2014.31>.
- [5] T. MIHÁLYDEÁK: *Aristotle's Syllogisms in Logical Semantics Relying on Optimistic, Average and Pessimistic Membership Functions*, in: *Rough Sets and Current Trends in Computing: 9th International Conference, RSCTC 2014, Granada and Madrid, Spain, July 9-13, 2014. Proceedings*, ed. by C. CORNELIS, M. KRYSZKIEWICZ, D. ŚLĘZAK, E. M. RUIZ, R. BELLO, L. SHANG, Cham: Springer International Publishing, 2014, pp. 59–70, ISBN: 978-3-319-08644-6, DOI: http://dx.doi.org/10.1007/978-3-319-08644-6_6.
- [6] T. MIHÁLYDEÁK: *Logic on Similarity Based Rough Sets*, in: *Rough Sets*, ed. by H. S. NGUYEN, Q.-T. HA, T. LI, M. PRZYBYŁA-KASPEREK, Cham: Springer International Publishing, 2018, pp. 270–283, ISBN: 978-3-319-99368-3.
- [7] T. MIHÁLYDEÁK: *Partial First-Order Logical Semantics Based on Approximations of Sets*, *Non-classical Modal and Predicate Logics* (2011), ed. by M. V. PETR CINTULA SHIER JU, pp. 85–90.
- [8] D. NAGY, T. MIHÁLYDEÁK, L. ASZALÓS: *Similarity Based Rough Sets*, in: *Rough Sets: International Joint Conference, IJCRS 2017, Olsztyn, Poland, July 3–7, 2017, Proceedings, Part II*, ed. by L. POLKOWSKI, Y. YAO, P. ARTIEMJEW, D. CIUCCI, D. LIU, D. ŚLĘZAK, B. ZIEŁOSKO, Cham: Springer International Publishing, 2017, pp. 94–107, ISBN: 978-3-319-60840-2, DOI: https://doi.org/10.1007/978-3-319-60840-2_7.

- [9] P. PAGLIANI, M. CHAKRABORTY: *Logic and Rough Sets: An Overview*, in: *A Geometry of Approximation: Rough Set Theory: Logic, Algebra and Topology of Conceptual Patterns*, Dordrecht: Springer Netherlands, 2008, pp. 169–191, ISBN: 978-1-4020-8622-9, DOI: https://doi.org/10.1007/978-1-4020-8622-9_5.
- [10] Z. PAWLAK: *Rough Sets: Theoretical Aspects of Reasoning about Data*, Theory and Decision Library D: Springer Netherlands, 1991, ISBN: 9780792314721, URL: <https://books.google.hu/books?id=MJPLCqIniGsC>.
- [11] Z. PAWLAK: *Rough sets*, *International Journal of Computer & Information Sciences* 11.5 (1982), pp. 341–356, ISSN: 1573-7640, DOI: <http://dx.doi.org/10.1007/BF01001956>.
- [12] Z. PAWLAK, A. SKOWRON: *Rough sets and Boolean reasoning*, *Information sciences* 177.1 (2007), pp. 41–73.
- [13] Z. PAWLAK, A. SKOWRON: *Rudiments of rough sets*, *Information sciences* 177.1 (2007), pp. 3–27.
- [14] A. SKOWRON, J. STEPANIUK: *Tolerance approximation spaces*, *Fundamenta Informaticae* 27.2 (1996), pp. 245–253.
- [15] Y. YAO, B. YAO: *Covering based rough set approximations*, *Information Sciences* 200 (2012), pp. 91–107, ISSN: 0020-0255, DOI: <http://dx.doi.org/10.1016/j.ins.2012.02.065>, URL: <http://www.sciencedirect.com/science/article/pii/S0020025512001934>.

Multi dimensional analysis of sensor communication processes*

Mohamed Amine Korteby, Zoltán Gál, Péter Polgár

University of Debrecen, Faculty of Informatics

korteby.amine@inf.unideb.hu

gal.zoltan@inf.unideb.hu

polgarp@mailbox.unideb.hu

Submitted: December 10, 2020

Accepted: March 8, 2021

Published online: May 18, 2021

Abstract

The Internet of Things requires communication mechanism to be optimal not only from the data transfer but from the energy consumption point of view, too. One of the most analyzed types of the sensor network is Low Energy Adaptive Clustering Hierarchy (LEACH) system depending on the population density, algorithm of cluster head election, heterogeneity of the energy and physical position of the nodes, velocity of the sink node, data aggregation rate and size of the data frame. Complexity of the system has been analyzed based on status data series of 360 simulation cases. New family of wireless sensor network (WSN) system is proposed with name CB-LEACH, having better characteristics than the classical LEACH system. The service ability of sensor network and dependency properties was done with analytic technique based on Singular Value Decomposition (SVD). Using this method there were identified most important modes serving as basis to regenerate responses of the studied sensor systems. It was found that the number of significant modes is just six. The novelty of the paper is a proof of concept that SVD is a useful multidimensional tool which can be used for describing the behavior of the newly proposed CB-LEACH family of sensor network mechanisms.

*This work was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund. This paper was supported also by the FIKP-20428-3/2018/ FEKUTSTRAT project of the University of Debrecen, Hungary and by the QoS-HPC-IoT Laboratory.

Keywords: Wireless sensor networks, Low Energy Adaptive Clustering Hierarchy (LEACH), switching, cluster, classification analysis

AMS Subject Classification: 62H30, 68M10, 90B15, 62H25, 62H30

1. Introduction

Nodes in wireless sensor networks monitor an Area of Interest (AoI) and, depending on their function, data is transmitted directly or indirectly to a Base Station (BS) or Sink Node (SN) that is connected to a wired network for further processing. This forwarding process is accomplished through various routing mechanisms that are the focus of the related research. Because sensor devices have limited power resources, the efficiency of their energy consumption is crucial. At the same time, these sensors are heavily concentrated in physical areas that are difficult to access physically by human, so their power supply is virtually non-replaceable. Thus, the key to their operating time is the efficiency of their energy consumption.

Routing protocols play a key role in sending aggregated data, inducing careful handling of such tasks. A successful model of a WSN system is one that can strike a good compromise between the maximum amount of data collection and the minimum amount of energy consumption. In WSN routing mechanisms, clustering appears to be an important consideration as it provides efficient energy savings and data delivery at the network level. Hierarchical routing which includes clustering is proving to be a preferred method of arranging sensors [8, 16]. At the same time, the method increases scalability, reduces the amount of energy loss, and delay time, while providing good connectivity and load balancing with increased network life [1, 7].

LEACH is a hierarchical, cluster-based, energy-efficient routing mechanism. It extends the lifetime of the system with randomly selected Cluster Head (CH) elements that forward frames from cluster members to the SN after aggregation. This intermediate transmission step significantly reduces the energy used by the nodes that operate the radio channel, since the transmission consumption of the data frame is based on the power function of the distance with exponent $2b$. Here b is the path loss exponent, its value depends on the path propagation properties and most often $b \geq 2$.

The further structure of the paper is as follows: in the second chapter we discuss the architecture and energy effects of the classical LEACH mechanism. In the third chapter we introduce our newly developed system Cost Balanced LEACH (CB-LEACH) and we present the methods used for the analysis of multidimensional data sets. Chapter four examines the newly proposed WSN system based on a synthetic state data series generated using 360 different simulations. In the last chapter we summarize the results and formulate the directions for the possible continuation of the research work.

2. Architecture and functions of the LEACH mechanism

Routing protocols were divided into four schemes: Network Structure Scheme, Topology Based Scheme, Communication System Scheme, and Reliable Routing Schema. Besides, the network structure scheme can be broken down into two classes: Flat routing and Hierarchical routing, depending on the role of the sensor nodes. In Flat routing, the sensor nodes have similar roles and functionalities in the network. Used in small area networks, the principal issue of this type of routing is scalability. Flooding and Gossiping, Directed diffusion, Rumor, SPIN are the most popular Flat routing protocols.

Hierarchical routing delivers greater energy efficiency and reliability within its architecture, the entire network is partitioned into clusters and unique nodes called cluster heads (CH) based on certain criteria. CH manages the gathering and aggregation of data received from its neighborhood, then forwards the collected data to the Base Station (BS) while providing other services to other nodes which consumes more energy. Hence is required cluster rotation method, a common approach often used to manage the energy harvesting inside the cluster.

LEACH is a Time Division Multiple Access (TDMA) based Media Access Control (MAC) routing protocol, self-adaptive and self-organized and most widely known hierarchical routing protocol. Due to its capability to increase energy efficiency it improves the lifespan of sensor nodes by using randomized rotations of local cluster head functions between the nodes [11, 12, 14, 18].

LEACH protocol consists of several rounds with two phases in each round: Set-up Phase and Steady Phase (see Figure 1).

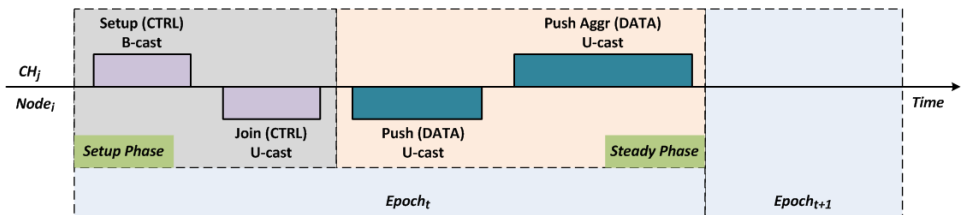


Figure 1. Mechanism for sending frames by epoch period.

In Set-up phase CH advertisement, cluster set-up, and TDMA scheduling are performed. During the CH advertisement, each node participates in CH election process based on the following equation:

$$T(n) = \begin{cases} \frac{p}{1-p \cdot \text{mod}(r, 1/p)}, & \text{if } n \in G, \\ 0, & \text{otherwise,} \end{cases} \quad (2.1)$$

where $T(n)$ is the threshold, n is the node index taking values in continuous interval $[0, 1]$ and set $\{1, 2, \dots, N\}$, respectively. Parameter p denotes the CH election

probability or the percentage of a node to become CH and is constant for a given simulation scenario. Function $\text{mod}(r, 1/p)$ is the remainder after dividing the current index r by the number $1/p$ and G is the set of sensor nodes that did not become CH in previous $1/p$ rounds. $Node_n$ acquires a cluster head function in the r^{th} epoch period if it drew $q_n < T(n)$. q_n is a random number $q_n \in [0, 1]$ drawn by $Node_n$ in the actual epoch. If $Node_n$ was CH in any of the last $1/p$ epochs, then $T(n) = 0$, so any $q_n > 0 = T(n)$, which means that it cannot regain a cluster head function in the actual epoch. With this rule and the random choice in the process, each node in the WSN is equal likely to become CH while achieving a homogeneous energy distribution among the CHs. The cluster head notifies all nodes in the system of its state with a control-type frame sent in a broadcast manner. Each of the nodes without CH function independently selects the most advantageous CH based on the field signal strength from the cluster head sources.

We refer round r to be an epoch. A node with index n becomes CH for the current round if the generated random value q_n of a sensor node is strictly less than the threshold $T(n)$. For a given round we have a fixed threshold and it is compared to the sensor node random values. If it is equal to 0, it means that n does not belong to G but belongs to the complementary set of nodes that were already elected as CH or died. Once a node is elected CH it cannot participate in the next $1/p$ round of CH election: i.e. if $p \in \{0.05, 0.10\}$ then the elected CH cannot be reelected in the next $1/p \in \{20, 10\}$ rounds respectively. This criterion is useful for the energy load balance inside the network since every node gets a better chance to become CH.

Based on this rule any of the nodes can become CH with similar probability, providing uniformity on AoI space the extra energy consumption of the CH function. The CHs announce the other nodes with radio channel broadcast about its new CH function. The ordinary nodes receive these signals and based on the intensity of the signal decide which cluster to become a member with. The signal intensity in practice depends on different environmental parameters but for the classical version of the LEACH, just the distance between the node and the CH is considered. The ordinary nodes send their responses to the most advantageous CH, becoming in this way members of that cluster. The CH schedules the communication inside of the cluster for the members during the actual epoch time.

In the epoch time second phase is named Steady Phase. During this period sensor end nodes send their data to the selected CH. The CH aggregates own frames with the transit traffic and send the aggregated data to the Sink Node. Thus, in the case of a WSN consisting of N sensor nodes, these two phases occur alternately in successive periods.

During the LEACH simulation, the strength of electromagnetic field is assumed to be exclusively distance-dependent, so the radio energy attenuation $A(d)$ per bit transmission is as follows:

$$A(d) = E_{Rx}(d)/E_{Tx}(d),$$

$$A(d) = \begin{cases} 1, & \text{if } d \in [0, \delta), \\ (\delta/d)^2, & \text{if } d \in [\delta, d_0), \\ (\delta/d_0)^2 \cdot (d_0/d)^{2b}, & \text{if } d \in [d_0, \infty), \end{cases} \quad (2.2)$$

where d is the distance between the sender and the receiver, E_{Rx} and E_{Tx} are the received and transmitted power, δ is the geometric parameter of the radio antenna (on the scale of centimeters), d_0 is the propagation distance threshold, b is the path loss exponent, where $b \geq 2$. The node with its data selects the most advantageous CH, i.e. the nearest CH. In the absence of data, the node goes to sleep mode for an epoch period to save energy [6, 10]. The CH provides a notification to the members of the cluster with a TDMA or Code Division Multiple Access (CDMA) control type frame, thus guaranteeing a collision-free frame transmission to the cluster head during the current epoch.

LEACH is the archetype for distributed routing protocols and one of the most efficient WSN mechanisms for power management. At the same time, we must not ignore some of its shortcomings [8]. The entry of any node into the CH function is independent of the residual energy of the given node, resulting in an uneven distribution in the physical location of the clusters in the covered area [1, 2, 14, 17]. This is exacerbated by the fact that the CH node may be even further away from the SN than members of its cluster, delivering data with poor efficiency [4, 5, 15]. Thus, some CH nodes drain their energy source sooner than other nodes, forming energy-free holes in physical space [3, 13]. Furthermore, the robustness of the network is reduced if the low-power node is given the CH function, because the frames of the cluster members arriving at the CH may be cancelled during the transmission to the SN due to the lack of power [9].

3. Cost Balanced LEACH mechanism: CB-LEACH

We propose a new family of LEACH mechanism named CB-LEACH having better performance than classical LEACH. With the change we proposed, we endowed the basic LEACH mechanism with additional skills and intelligence. This is a Cost Balanced (CB) version of LEACH, which decides the route for transmitting frames based on complex metric. In the case of CB-LEACH, we allow the SN to move along a given path, as well as more efficient selection of the optimal CH for the end nodes. Because nodes in the current epoch time may be closer to the moving SN than the selected CH, they should send their frames directly to the SN instead of indirectly through the selected CH. To this end, we also include the SN in the set of selectable CH nodes. The energy of SN is considered not decreasing over time and is the largest in the WSN system. We consider not only the distance of the possible CHs from the end nodes but also their energy level, as well. To do this, a given sensor node decides own CH to connect to, based on the following metric:

$$\text{COST}(i, j, \alpha) = \alpha \cdot \frac{E_0}{E_{CH_j}} + (1 - \alpha) \cdot \left(\frac{D(i, j)}{d_0} \right)^{2b}, \quad (3.1)$$

where $\text{COST}(i, j, \alpha)$ is the metric of Node_i and CH_j calculated with a balance factor $\alpha \in [0, 1]$, $D(i, j)$ is the distance between Node_i and CH_j , d_0 is the propagation distance threshold and b is the path loss exponent, E_0 is the initial energy for any normal type node (NN) and E_{CH_j} is the actual energy level for a given CH_j . Parameter α named balance factor is our newly introduced element of the model to connect energy and distance properties of the nodes. Since the denominator of the first term of equation (3.1) can be zero after a long time, the properties of the system are analysed up to the last operating node, so that each term of the formula remains a positive and finite quantity.

In current epoch time Node_i chooses the CH_j for which the value of the $\text{COST}(i, j, \alpha)$ metric is the smallest. It can be observed that if $\alpha = 0$, we get back the mobility-enhanced CH version of the LEACH mechanism, which is further identified as ENH-LEACH (Enhanced). If we do not change the position of the SN in time, we get back the Basic LEACH mechanism, which is hereinafter referred to as BAS-LEACH.

However, if $\alpha = 1$, only the energy of the CH counts, which generates a competitive situation between the potential CHs and each Node_i will choose the same single CH, i.e. the SN with the maximum energy. In this case, each node sends its data directly to the SN, independent of the other potential CHs. This routing mechanism will be referred to as DS (Direct Sequence) hereinafter.

Our simulation measurements demonstrated that ENH-LEACH and DS are significantly different routing protocols. The two extreme values of the balancing factor α have advantages and disadvantages from different aspects. When α is in range $(0, 1)$ it has harmonization effect on the decisions of LEACH routing mechanism. Based on these, the proposed CB-LEACH is compatible with the family of protocols as follows:

$$\text{CB-LEACH} \sim \begin{cases} \text{ENH-LEACH}, & \text{if } \alpha = 0, \\ \text{Modified LEACH Family}, & \text{if } \alpha \in (0, 1), \\ \text{DS}, & \text{if } \alpha = 1. \end{cases} \quad (3.2)$$

The DS sends the frame from each node directly to the SN. This means long distances, which consumes a significant amount of energy. However, due to the random transmission mechanism, the planar distribution of the residual energy of the nodes is expected to be strongly dependent on the distance between end node and SN. ENH-LEACH divides the physical field of the WSN into two parts, since treating the SN as a cluster head provides the nodes in the vicinity of the SN with a radius d_0 a favorable power consumption for frame transmission time. The obvious question is what properties the modified LEACH family inherits from the two boundary cases as a function of the α parameter. Further questions can put about the dependence of the CB-LEACH system on other parameters.

The CB-LEACH mechanism depends on a significant number of parameters, including execution of 360 simulation cases. In each case, the communication activity of a given WSN was completed during epoch numbers in the scale of $5 \cdot 10^5$. Vectorisation of the analyzed system responses is based on normalization of elemental

answer vectors and concatenation of them in common vector named probe. There is a legitimate need to identify the parameters that most significantly influence the behavior of the present network. For this, we applied Singular Value Decomposition (SVD) analytic method [2], to efficiently evaluate behavior dependence on the tuple of independent parameters. Having relatively high number of simulation probes, SVD gives possibility to determine the number of most significant modes of the CB-LEACH system behavior.

4. CB-LEACH system analysis and characteristics

We analyzed and evaluated the synthetic state data obtained from $n' = 360$ probes of Direct Sequence (DS), Basic LEACH (BAS-LEACH), and Enhanced LEACH (ENH-LEACH) simulations according to different parameters, using the methods described in this chapter. These mechanisms can be considered as special cases of CB-LEACH family proposed by us.

Figure 2 shows Area of Interest after ending the simulation of one probe example. Area of Interest is the biggest dotted black circle. Red dotted arc represents d_0 (see formula (2.2)), radio channel distance threshold. There are N sensor nodes represented by black circles. Normal nodes (NN) and Advanced nodes (AN) are distinguished by the radius of black circles. Smallest circles belong to NNs and ANs have greater radius. For each simulation probe a number of N sensor nodes are spread in uniformly distributed coordinates inside of AoI. Gray scale of the node marker is proportional with the age of node. Darker nodes die first, brighter nodes die later.

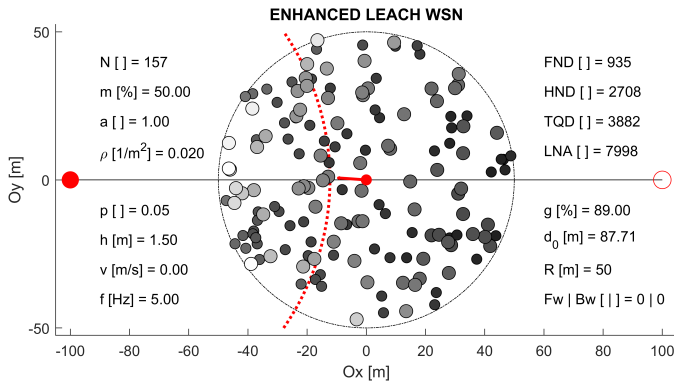


Figure 2. Example of one simulation probe after the execution of simulation. Parameter names conform to Table 1.

Depending on the constant velocity value, the sink node may move on trajectory in horizontal segment between filled red (left) and non filled (right) circles during the actual simulation. Number of forward and backward courses depend on the velocity and running time and are counted in variables Fw and Bw, respectively.

Being the sensor nodes uniformly distributed in space, the weighted energy point (WEP) in space of the WSN is situated in the center of the AoI. This WEP moves during the simulation if the remaining energy of the nodes is not uniformly decreasing in space. In this example red segment in the center zone of AoI represent movement of WEP during the simulation. FND, HND, TQD and LNA are epoch identifiers for first node die, half node die, third quarter node die and last node alive, respectively. Population density in the AoI is constant, $\rho = 0.2m^{-2}$ during all simulations, meaning one node per $5m^2$.

The unchanged characteristics of the WSN system and the values of the six parameters of the simulation environment are shown in Table 1. In the table specific parameters have more than one value, others just one. Parameters with variable values are marked with star (*) character at the beginning of the line. Hereby we have $n' = 5 \cdot 3 \cdot 3 \cdot 2 \cdot 2 \cdot 2 = 360$ combinations of the orthogonal parameters giving us n' simulation probes.

Table 1. Parameters of the simulated system. Parameters having variable values are marked with star (*) character at the beginning of the row. Other parameters have fixed value during the simulation probes.

<i>Parameters</i>	<i>Value(s)</i>
Physical area size ($xm \times ym$)	$100m \times 100m$
Radius of the field	$R = xm/2$
Initial and farthest position of the Sink Node	$(-xm, 0)(xm, 0)$
Number of nodes of the WSN	$N = 157$
* Balance Factor	$\alpha = 0, 0.25, 0.50, 0.75, 1.00$
* Ratio AN node number / total nodes, N	$m = 0.3, 0.5, 0.7$
* Velocity of the Sink Node	$v = 0, 5, 10$ m/s
* Radio frames length	$Fs = 1000, 4000$ bits
* Aggregation level of the radio frames	$g = 0.10, 0.89$
* Ratio of the CH nodes	$p = 5, 10$ %
Energy factor of the AN	$a = 1$
Initial energy unit	$E_0 = 2.5$ J
Energy consumption of the electronics	$E_{elec} = 50$ nJ/bit
Energy multipath factor vs. of antenna height	$E_{mp} = 1.3$ pJ
Energy consumption of the antenna amplifier	$E_{amp} = 0.1$ nJ/bit
Energy consumption of the frame aggregation	$EDA = 5$ nJ/bit
Radio antenna height	$h = 1.5$ m
Radio channel distance threshold	$d_0 = 87.7$ m
Energy Free Space Factor	$E_{fs} = 10$ pJ/bit/ m^2
Path loss exponent	$b = 4$

The CB-LEACH balancing factor α influences the cluster head selection strategy and the type of WSN routing based on equations (3.1) and (3.2). The cluster

head average probability, p plays a role in relation (2.1). Typically, the operation of the system is usually tested with relatively small values. To show the effect of heterogeneous initial energy levels, the set of N sensor nodes are classified into two energy groups: NN (Normal Node) and AN (Advanced Node). AN initially has $a + 1 > 1$ times greater energy than NN. The Ratio of the AN nodes number to the total nodes N is $m \in (0, 1)$. The initial energies of these two groups, and the entire WSN system, are as follows:

$$\begin{aligned} E_N &= (1 - m) \cdot N \cdot E_0, \\ E_A &= m \cdot (a + 1) \cdot N \cdot E_0, \\ E_T &= E_N + E_A = (a \cdot m + 1) \cdot N \cdot E_0. \end{aligned}$$

Examples of status responses time series of WSN system for Basic LEACH can be seen on Figure 3. On the left hand side figure NN and AN node types have the spatial average energy level versus time represented with blue and green curves, respectively. Average energy curve for the whole WSN is shown by red curve. On the right hand side figure same colors are used to represent alive nodes versus the time.

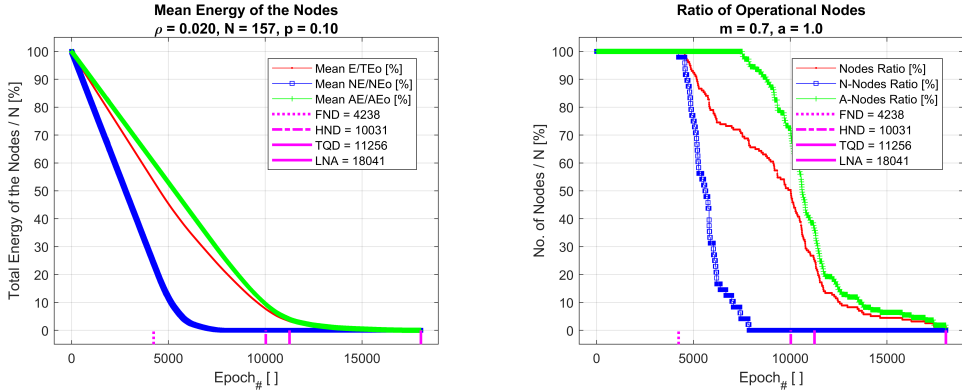


Figure 3. Examples of two elemental time series responses of Basic LEACH *probe*: Remaining average energy level in space, E_i^B (left) and Relative number of operational nodes, N_i^B (right). Initial parameters are: $\alpha = 0$; $m = 0.7$; $v = 0\text{m/s}$; $Fs = 4000\text{bit}$; $g = 0.1$; $p = 0.1$.

The response data series provided by the WSN system per routing mechanism are given by the following vector concatenations:

$$y_i^D = (E_i^D, S_i^D, F_i^D, N_i^D) \in M_{1,8e_0}, \quad (4.1)$$

$$y_i^B = (E_i^B, S_i^B, F_i^B, N_i^B, T_i^B, K_i^B, D_i^B, R_i^B) \in M_{1,8e_0}, \quad (4.2)$$

$$y_i^E = (E_i^E, S_i^E, F_i^E, N_i^E, T_i^E, K_i^E, D_i^E, R_i^E) \in M_{1,8e_0}, \quad (4.3)$$

where $M_{1,8e_0}$ is the class of matrices with one row and $8e_0$ columns. Upper indexes D, B and E stands for Direct Sequence, Basic LEACH and Enhanced LEACH mechanisms, respectively. Elemental responses of the WSN for simulation $i = 1, \dots, m'$ are vectors with e_0 elements each: spatial average remaining energy level in % (E_i), Shannon entropy in space of energy levels (S_i), relative number of frames sent to the sink node in % (F_i), relative number of operational nodes in % (N_i), relative number of transactions/epoch in % (T_i), relative number of clusters/epoch in % (K_i), average distance in space between CH and sink node in % of d_0 (D_i), average cluster radius in % of d_0 (R_i).

Since we executed simulation for each combination of independent parameter values, every probe gives us response as a set of data series of the WSN system for DS, BAS-LEACH and ENH-LEACH (see Figure 4). Because number of elements of the response time series belonging to different probe are not equal, it was used dilatation and compression of the time. The common length $e_0 = 27,953$ of one elemental response vector is the average size of the elemental time series length of the m' probes, determined from Basic and Enhanced LEACH cases. In this way response vectors in formulae (4.1), (4.2) and (4.3) are transformed and considered as status vectors versus progress in the range [0%, 100%).

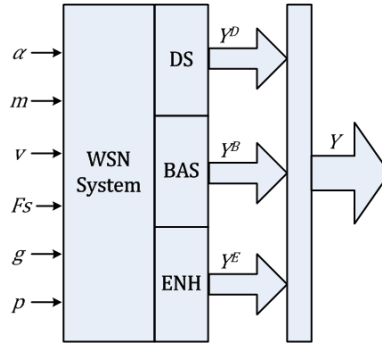


Figure 4. Structure of the WSN simulation system. For each parameter tuple the WSN system generates sensors in random positions of the AoI and executes simulation for DS, BAS-LEACH and ENH-LEACH routing mechanisms separately. Scaled and normalized response status data series are concatenated in probe vectors. Matrix Y is the response matrix of WSN system having n' probes.

Each of the above three row-vectors y_i^D, y_i^B, y_i^E contain $8e_0$ elements. Because in case of DS mechanism just E^D, S^D, F^D, N^D response data series have meaning, being half as many time series as BAS-LEACH or ENH-LEACH mechanisms have, each DS vector mentioned was scaled to double length, $2e_0$. Each elemental status data series of probe $y_i, i = \overline{1, n'}$ generated by simulation i of the WSN system is concatenated to create the probe vector identified by the following formula:

$$y_i = (y_i^D, y_i^B, y_i^E)^T \in M_{m', 1},$$

where T is the transpose operator. Collection of column vectors y_i , $i = \overline{1, n'}$ represents status matrix of the WSN system given by the following formula:

$$Y = (y_1, \dots, y_{n'}) \in M_{m', n'},$$

where $m' = (2 \cdot 4 + 8 + 8) \cdot e_0 = 24 \cdot e_0 = 670,872$ and $n' = 360$.

Should mention that since the structure of the y_i column vector representing the simulation probe i is fixed, it can be considered as a vectorized image map. Each of the n' different probes is a composed object of twenty vectorized images. Each object belonging to the corresponding probe has the same structure but different content from the others. Hereby we have better representation of the system responses and we search a smaller number of representative probes, k that are able to best characterize the WSN system.

We used Singular Value Decomposition to determine the number of most important modes able to form an orthogonal basis for all the n' probes. According to Figure 5 (left), the correlation coefficients between the responses y_i and y_j are in the interval $(0.5, 0.95)$, mostly closer to the larger values, where $i < j$, $i, j \in \{1, \dots, n'\}$. It is numerically confirmed by Figure 5 (right), that the $k = 6$ largest singular values represents 33.55% of the information given by $n' = 360$ singular values. The elbow part of the singular values scree plot proves that there are $k = 6$ virtual modes serving as synthesization basis for all $n' = 360$ probes. It means that there are 6 different virtual parameter tensor values (α, m, v, Fs, g, p) , which represent 6 virtual probes that primarily characterize our CB-LEACH WSN system. This is considered the main novelty of our findings about the proposed CB-LEACH family of routing mechanisms.

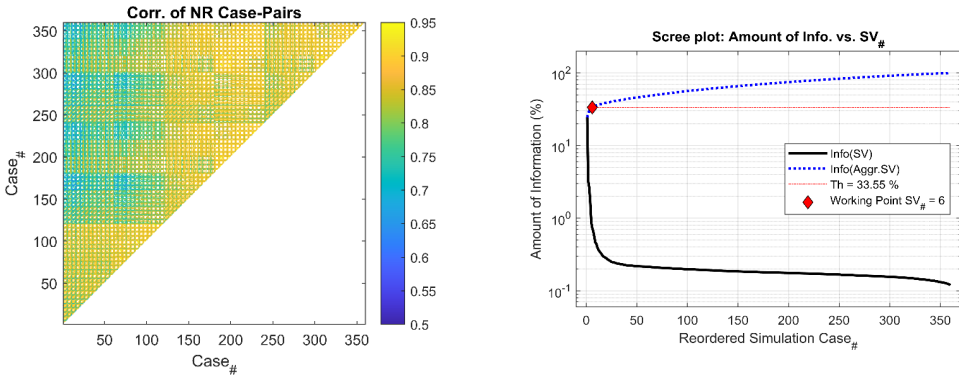


Figure 5. Correlation matrix of Y responses of WSN system (left) and singular values of the response matrix Y (right).

Decomposition of status matrix Y is given by the following formula:

$$Y = U * \Sigma * V^T,$$

where matrixes $U \in M_{m', m'}$, $\Sigma \in M_{m', n'}$ and $V \in M_{n', n'}$ are unitary basis in progress, singular value matrix and unitary basis matrix in AoI space, respectively.

The simplified response matrix \hat{Y} is the approximation of the matrix Y and contains in each column 20 elementary data series having the noise reduced significantly:

$$\hat{Y} = U_k * \Sigma_k * V_k^T,$$

where matrixes $U_k \in M_{m',k}$, $\Sigma_k \in M_{k,k}$ and $V_k \in M_{n',k}$ are most significant k columns of matrixes U , Σ and V , respectively.

Figure 6 demonstrates similarity between the original (Y) and filtered response system (\hat{Y}). Both parts of Figure 6 contain 7,200 data series versus progress. The response data series belonging to a given probe contains the values going from top to bottom along with the columns. The values of the normalized and scaled response data series are represented by color codes having values conform to the color bars. It can be observed that the two images are very similar in both aspect of layout and sharpness. The root mean square error (RMSE) of the two matrices is $RMSE = 9.15\%$.

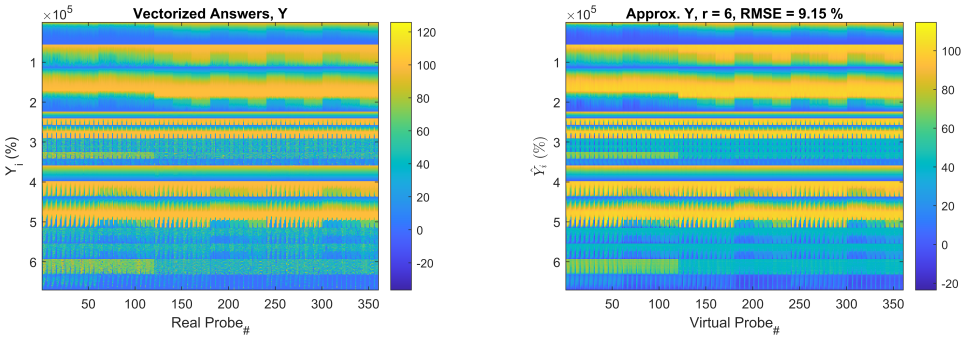


Figure 6. Data series responses of a WSN system, original Y (left) and approximated \hat{Y} (right). Progress direction is from top to bottom, normalized values are represented by color codes.

The facts found so far prove the feedback neural networks to be useful in analysis of complex systems as they reduce the learning process required for modelling. This aspect is not discussed in this paper because it is subject of our next research phase. Using the k -means clustering method, we classified the responses of the n' simulations cases (probes) into k classes, i.e. y_i , $i = \overline{(1, n)}$ column vectors into clusters. The result of this computation is illustrated in Figure 7. The number of probes belonging to groups $1, \dots, k$ are 96, 72, 96, 24, 38 and 34, respectively (see Figure 7 left). Since the number of members is of the similar order of magnitude per class, each class is important. Figure 7 (right) represents centroid vectors of $k = 6$ clusters conform to probes classification provided by the k -Mean algorithm. Centroid of a class is the mean vector in space for member vectors belonging to the same class. We should mention that significant difference exists between these centroid vectors.

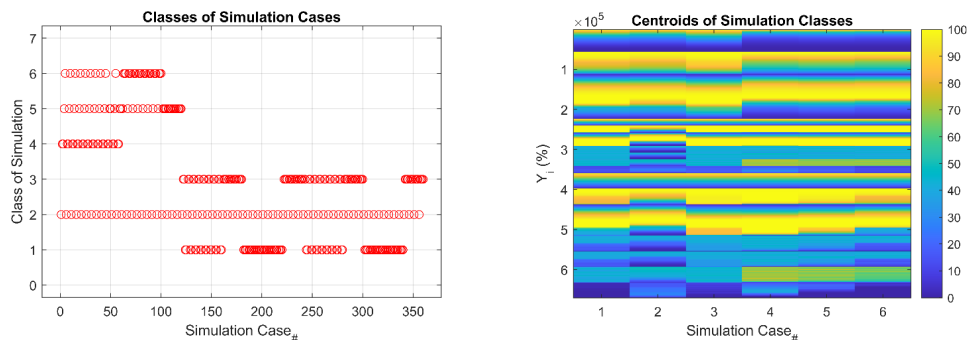


Figure 7. Classification of simulations by k -Mean algorithm into $k = 6$ clusters (left) and centroids of the probe classes (right).

This aspect proves detectable diversion between the probe classes resulted for simulations based on n' tuple of independent parameters of the proposed CB-LEACH family of WSN mechanisms.

5. Summary and conclusions

The method presented in the paper allows the behaviour analysis of the newly proposed CB-LEACH wireless sensor routing mechanism. CB-LEACH is an energy-efficient extension of the classic LEACH, the operation of which can be influenced by six parameters. To determine the optimal parameter tensor, the analysis of the synthetic state data set was generated by 360 different simulation cases which required the use of dimension reduction analysis based on Singular Value Decomposition to determine the number of most significant modes. This number is $k = 6$, which is not a coincidence due to the number of independent parameters (α, m, v, Fs, g and p), but generic property due to the special behaviour of the CB-LEACH family. To classify the $n' = 360$ simulation cases into k classes, we used k -Mean clusterization algorithm enrolling to the closest cluster with good accuracy each simulation probe based on independent tuples of parameters.

As a continuation of the research work, further analyzes are considered important on the noise reduction of response probes of elementary data series as a function of singular value number and reducing precessing capacity of the method. Sensitivity analysis will be executed to find approximated closed-form expression of the energy usage dependence on the multidimensional parameter set.

References

- [1] O. M. ALIA: *Dynamic relocation of mobile base station in wireless sensor networks using a cluster-based harmony search algorithm*, Information Sciences (2017), pp. 76–95, DOI: <https://doi.org/10.1016/j.ins.2016.12.046>.

- [2] S. BALJINDER, V. AMIT, K. MANIT: *A survey on various energy-efficient routing protocols in WSN*, International Journal of Advanced Research, Ideas and Innovations in Technology 4 (2019), pp. 862–865.
- [3] M. BELWAL: *Energy Efficient LEACH and Improved LEACH: A Review*, International Journal of Advanced Research in Computer Science 10 (2019), pp. 51–52, DOI: <https://doi.org/10.26483/ijarcs.v10i3.6412>.
- [4] Y. CHI, H. CHANG: *An energy-aware grid-based routing scheme for wireless sensor networks*, Telecommunication Systems 54 (2013), pp. 405–415, DOI: <https://doi.org/10.1007/s11235-013-9742-x>.
- [5] Z. GAL, M. KORTEBY: *Energy Sparing of the Leach Communication Mechanism in Heterogeneous WSN*, in: 8th International Conference on Advanced Science and Information Technology, 2019, pp. 53–64.
- [6] N. GHOSH, I. BANERJEE: *An energy-efficient path determination strategy for mobile data collectors in wireless sensor network*, Computers & Electrical Engineering 48 (2015), pp. 417–435, DOI: <https://doi.org/10.1016/j.compeleceng.2015.09.004>.
- [7] S. HARJIT, S. VARUN: *Energy Efficient Clustering for Network Stability and Longevity for Heterogeneous Wireless Sensor Network*, International Journal of Engineering Science and Computing 8 (2018), pp. 18867–18872.
- [8] W. HEINZELMAN, A. CHANDRAKASAN: *An application-specific protocol architecture for wireless micro sensor networks*, IEEE Transactions on Wireless Communications 1 (4 2002), pp. 660–670, DOI: <https://doi.org/10.1109/TWC.2002.804190>.
- [9] N. ISMAT, R. QURESHI, I. MUMTAZ: *Efficient Clustering for Mobile Wireless Sensor Networks*, in: 17th IEEE International Multi Topic Conference, 2014, pp. 110–114, DOI: <https://doi.org/10.1109/INMIC.2014.7097321>.
- [10] A. KHAN, A. ABDULLAH, M. RAZZAQUE, J. BANGASH: *Vgdra: A virtual gridbased dynamic routes adjustment scheme for mobile sink-based wireless sensor networks*, IEEE Sensors Journal 15 (2015), pp. 526–534.
- [11] J. KIM, J. IN, K. HUR, J. KIM, D. EOM: *An intelligent agent-based routing structure for mobile sinks in WSNs*, IEEE Transactions on Consumer Electronics 4 (2010), pp. 2310–2316.
- [12] W. LIANG, J. LUO, X. XU: *Prolonging network lifetime via a controlled mobile sink in wireless sensor networks*, in: IEEE global telecommunications conference GLOBECOM, 2010, pp. 1–6.
- [13] N. MITTAL, U. SINGH, B. SOHI: *A stable energy efficient clustering protocol for wireless sensor networks*, Wireless Networks 23 (2017), pp. 1809–1821.
- [14] H. SALARIAN, K. CHIN, F. NAGHDY: *An energy-efficient mobile-sink path selection strategy for wireless sensor networks*, IEEE Transactions on Vehicular Technology 63 (2014), pp. 2407–2419.
- [15] M. THAKUR: *Mobile Sink Based NLEACH Protocol by using Ant Colony Optimization*, International Journal of Science Research and Technology 2 (2016), pp. 1–10.
- [16] J. WANG, Y. CAO, B. LI, H. KIM, S. LEE: *Particle swarm optimization based clustering algorithm with mobile sink for WSNs*, Future Generation Computer Systems 76 (2016), pp. 452–457, DOI: <https://doi.org/10.1016/j.future.2016.08.004>.
- [17] M. A. ZAHHAD, S. AHMED, N. SABOR, S. SASAKI: *Mobile sink-based adaptive immune energy-efficient clustering protocol for improving the lifetime and stability period of wireless sensor networks*, IEEE Sensors Journal 15 (2015), pp. 4576–4586.
- [18] Z. ZHOU, C. DU, L. SHU, G. HANCKE, J. NIU, J. NING: *An Energy-Balanced Heuristic for Mobile Sink Scheduling in Hybrid WSNs*, IEEE Transactions on Industrial Informatics 12 (2016), pp. 28–40.

Portfolio solver for verifying Binarized Neural Networks

Gergely Kovásznai^a, Krisztián Gajdár^b,
Nina Narodytska^c

^aEszterházy Károly University, Eger, Hungary
kovasznoi.gergely@uni-eszterhazy.hu

^bEricsson Hungary, Budapest, Hungary

^cVMware Research, Palo Alto, USA

Submitted: November 21, 2020

Accepted: March 17, 2021

Published online: May 18, 2021

Abstract

Although deep learning is a very successful AI technology, many concerns have been raised about to what extent the decisions making process of deep neural networks can be trusted. Verifying of properties of neural networks such as adversarial robustness and network equivalence sheds light on the trustiness of such systems. We focus on an important family of deep neural networks, the Binarized Neural Networks (BNNs) that are useful in resource-constrained environments, like embedded devices. We introduce our portfolio solver that is able to encode BNN properties for SAT, SMT, and MIP solvers and run them in parallel, in a portfolio setting. In the paper we propose all the corresponding encodings of different types of BNN layers as well as BNN properties into SAT, SMT, cardinality constraints, and pseudo-Boolean constraints. Our experimental results demonstrate that our solver is capable of verifying adversarial robustness of medium-sized BNNs in reasonable time and seems to scale for larger BNNs. We also report on experiments on network equivalence with promising results.

Keywords: Artificial intelligence, neural network, adversarial robustness, formal method, verification, SAT, SMT, MIP

1. Introduction

Deep learning is a very successful AI technology that makes impact in a variety of practical applications ranging from vision to speech recognition and natural language [17]. However, many concerns have been raised about the decision-making process behind deep learning technology, in particular, deep neural networks. For instance, can we trust decisions that neural networks make [14, 18, 32]? One way to address this problem is to define properties that we expect the network to satisfy. Verifying whether the network satisfies these properties sheds light on the properties of the function that it represents [7, 23, 31, 34, 37].

One important family of deep neural networks is the class of *Binarized Neural Networks (BNNs)* [20]. These networks have a number of useful features that are useful in resource-constrained environments, like embedded devices or mobile phones [25, 28]. Firstly, these networks are memory efficient, as their parameters are primarily binary. Secondly, they are computationally efficient as all activations are binary, which enables the use of specialized algorithms for fast binary matrix multiplication. Moreover, BNNs allow a compact representation in Boolean logic [7, 31].

There exist approaches that formulate the verification of neural networks to Satisfiability Modulo Theories (SMT) [13, 19, 23], while others do the same to Mixed-Integer Programming (MIP) [11, 15, 36]. In some sense, this work can be considered to be the continuation of that in [7, 31], which translate all the MIP constraints to SAT.

The goal of this work is to attack the problem of verifying important properties of BNNs by applying several kinds of approaches and solvers, such as *SAT*, *SMT* and *MIP* solvers. We introduce our solver that is able to encode BNN properties for those solvers and run them in parallel, in a *portfolio setting*. We focus on the important properties of neural networks *adversarial robustness* and *network equivalence*.

In this paper we introduce how to use our solver and report on experiments on verifying both robustness and equivalence. Experimental results show that our solver is capable of verifying those properties of medium-sized BNNs in reasonable runtime, especially when the solvers MINICARD + Z3 are run in parallel.

2. Preliminaries

A *literal* is a Boolean variable x or its negation $\neg x$. A *clause* is a disjunction of literals. A Boolean formula is in *Conjunctive Normal Form (CNF)*, if it is a conjunction of clauses. We say that a Boolean formula, typically in CNF, is *satisfiable*, if there exists a truth assignment to the Boolean variables of the formula such that the formula evaluates to 1 (true). Otherwise, it is said to be *unsatisfiable (UNSAT)*. The *Boolean Satisfiability (SAT)* problem is the problem of determining if a Boolean formula is satisfiable.

Satisfiability Modulo Theories (SMT) is the decision problem of checking satis-

fiability of a Boolean formula with respect to some background theory. Common theories include the theory of integers, reals, fixed-size bit-vectors, etc. The logics that one could use might differ from each other in the linearity or non-linearity of arithmetic and the presence or absence of quantifiers. In this paper, we use the theory of integers combined with linear arithmetic and without quantifiers – denoted as QF_LIA in the SMT-LIB standard [5].

A *Boolean cardinality constraint* is defined as an expression $\sum_{i=1}^n l_i \circ_{\text{rel}} c$, where l_1, \dots, l_n are literals, $\circ_{\text{rel}} \in \{\geq, \leq, =\}$, and $c \in \mathbb{N}$ is a constant where $0 \leq c \leq n$.

A *pseudo-Boolean constraint* can be considered as a “weighted” Boolean cardinality constraint, and can be defined as an expression $\sum_{i=1}^n w_i l_i \circ_{\text{rel}} c$, where $w_i \in \mathbb{N}$, $w_i > 0$.

We assume the reader is familiar with the notion and elementary properties of feedforward neural networks. We consider a feedforward neural network to compute a function F where $F(\mathbf{x})$ represents the output of F on the input \mathbf{x} . Let $\ell(\mathbf{x})$ denote the ground truth label of \mathbf{x} . Our tool can analyze two properties of neural networks: adversarial robustness and network equivalence. We call a neural network robust on a given input if small perturbations to the input do not lead to misclassification, as defined as follows, where $\boldsymbol{\tau}$ represents the perturbation and $\epsilon \in \mathbb{N}$ the upper bound for the p -norm of $\boldsymbol{\tau}$.

Definition 2.1 (Adversarial robustness). A feedforward neural network F is (ϵ, p) -robust for an input \mathbf{x} if $\neg \exists \boldsymbol{\tau}, \|\boldsymbol{\tau}\|_p \leq \epsilon$ such that $F(\mathbf{x} + \boldsymbol{\tau}) \neq \ell(\mathbf{x})$.

The case of $p = \infty$, which bounds the maximum perturbation applied to each entry in \mathbf{x} , is especially interesting and has been considered frequently in literature.

Similar to robustness, the equivalence of neural networks is also a property that many would like to verify. We consider two neural networks equivalent if they generate the same output on all inputs, as defined as follows, where \mathcal{X} denotes the input domain.

Definition 2.2 (Network equivalence). Two feedforward neural networks F_1 and F_2 are equivalent if $\forall \mathbf{x} \in \mathcal{X} F_1(\mathbf{x}) = F_2(\mathbf{x})$.

3. Encoding of Binarized Neural Networks

A *Binarized Neural Network (BNN)* is a feedforward network where weights and activations are predominantly binary [20]. It is convenient to describe the structure of a BNN in terms of composition of blocks of layers rather than individual layers. Each block consists of a collection of linear and non-linear transformations. Blocks are assembled sequentially to form a BNN.

Internal block. Each internal block (denoted as BLOCK) in a BNN performs a collection of transformations over a binary input vector and outputs a binary vector. While the input and output of a BLOCK are binary vectors, the internal layers of BLOCK can produce real-valued intermediate outputs. A common construction

of an internal BLOCK (taken from [20]) is composed of three main operations:¹ a linear transformation (LIN), batch normalization (BN), and binarization (BIN). Table 1 presents the formal definition of these transformations. Figure 1 shows two BLOCKS connected sequentially.

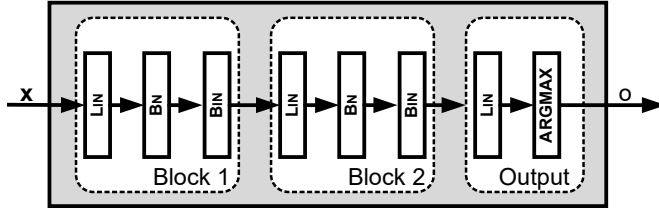


Figure 1. A schematic view of a binarized neural network. The internal blocks also have an additional HARDTANH layer during the training.

Table 1. Structure of internal and outputs blocks which stacked together form a binarized neural network. In the training phase, there might be an additional HARDTANH layer after batch normalization. A_k and b_k are parameters of the LIN layer, whereas $\alpha_{k_i}, \gamma_{k_i}, \mu_{k_i}, \sigma_{k_i}$ are parameters of the BN layer. The μ 's and σ 's correspond to mean and standard deviation computed in the training phase. The BIN layer is parameter free.

Structure of k^{th} internal block, $\text{BLOCK}_k : \{-1, 1\}^{n_k} \rightarrow \{-1, 1\}^{n_{k+1}}$ on $\mathbf{x}_k \in \{-1, 1\}^{n_k}$	
LIN	$\mathbf{y} = A_k \mathbf{x}_k + \mathbf{b}_k$, where $A_k \in \{-1, 1\}^{n_{k+1} \times n_k}$ and $\mathbf{b}_k, \mathbf{y} \in \mathbb{R}^{n_{k+1}}$
BN	$z_i = \alpha_{k_i} \left(\frac{y_i - \mu_{k_i}}{\sigma_{k_i}} \right) + \gamma_{k_i}$, where $\alpha_k, \gamma_k, \mu_k, \sigma_k, \mathbf{z} \in \mathbb{R}^{n_{k+1}}$. Assume $\sigma_{k_i} > 0$.
BIN	$\mathbf{x}_{k+1} = \text{sign}(\mathbf{z})$ where $\mathbf{x}_{k+1} \in \{-1, 1\}^{n_{k+1}}$
Structure of output block, $\text{O} : \{-1, 1\}^{n_m} \rightarrow [1, s]$ on input $\mathbf{x}_m \in \{-1, 1\}^{n_m}$	
LIN	$\mathbf{w} = A_m \mathbf{x}_m + \mathbf{b}_m$, where $A_m \in \{-1, 1\}^{s \times n_m}$ and $\mathbf{b}_m, \mathbf{w} \in \mathbb{R}^s$
ARGMAX	$o = \text{argmax}(\mathbf{w})$, where $o \in [1, s]$

Output block. The output block (denoted as O) produces the classification decision for a given binary input vector. It consists of two layers (see Table 1). The first layer applies a linear (affine) transformation that maps its input to a vector of integers, one for each output label class. This is followed by an ARGMAX layer, which outputs the index of the largest entry in this vector as the predicted label.

¹In the training phase, there is an additional HARDTANH layer after batch normalization layer that is omitted in the inference phase [20].

Network of blocks. BNN is a deep feedforward network formed by assembling a sequence of internal blocks and an output block. Suppose we have $m - 1$ internal blocks, $\text{BLOCK}_m, \dots, \text{BLOCK}_{m-1}$ that are placed consecutively, so the output of a block is the input to the next block in the list. Let n_k denote the number of input values to BLOCK_k . Let $\mathbf{x}_k \in \{-1, 1\}^{n_k}$ be the input to BLOCK_k and $\mathbf{x}_{k+1} \in \{-1, 1\}^{n_{k+1}}$ be its output. The input of the first block is the input of the network. We assume that the input of the network is a vector of integers, which holds for the image classification task if images are in the standard RGB format. Note that these integers can be encoded with binary values $\{-1, 1\}$ using a standard encoding. It is also an option to add an additional BNBIN block before BLOCK_1 to binarize the input images (see Sections 3.3 and 6.1). Therefore, we keep notations uniform for all layers by assuming that inputs are all binary. The output of the last layer, $\mathbf{x}_m \in \{-1, 1\}^{n_m}$, is passed to the output block O to obtain one of the s labels.

Definition 3.1 (Binarized Neural Network). A binarized neural network $\text{BNN} : \{-1, 1\}^{n_1} \rightarrow [1, \dots, s]$ is a feedforward network that is composed of m blocks, $\text{BLOCK}_1, \dots, \text{BLOCK}_{m-1}, \text{O}$. Formally, given an input \mathbf{x} ,

$$\text{BNN}(\mathbf{x}) = \text{O}(\text{BLOCK}_{m-1}(\dots \text{BLOCK}_1(\mathbf{x}) \dots)).$$

In the following sections, we show how to encode an entire BNN structure into Boolean constraints, including cardinality constraints.

3.1. Encoding of internal blocks

Each internal block is encoded separately as proposed in [7, 31]. Here we follow the encoding by Narodystka *et al.* Let $\mathbf{x} \in \{-1, 1\}^{n_k}$ denote the input to the k^{th} block, $\mathbf{o} \in \{-1, 1\}^{n_{k+1}}$ the output. Since the block consists of three layers, they are encoded separately as follows:

LIN. The first layer applies a linear transformation to the input vector \mathbf{x} . Let \mathbf{a}_i denote the i^{th} row of the matrix A_k and b_i the i^{th} element of the vector \mathbf{b}_k . We get the constraints

$$y_i = \langle \mathbf{a}_i, \mathbf{x} \rangle + b_i, \quad \text{for all } i \in [1, n_{k+1}].$$

BN. The second layer applies batch normalization to the output \mathbf{y} of the previous layer. Let $\alpha_i, \gamma_i, \mu_i, \sigma_i$ denote the i^{th} element of the vectors $\boldsymbol{\alpha}_k, \boldsymbol{\gamma}_k, \boldsymbol{\mu}_k, \boldsymbol{\sigma}_k$, respectively. Assume $\alpha_i \neq 0$. We get the constraints

$$z_i = \alpha_i \frac{y_i - \mu_i}{\sigma_i} + \gamma_i, \quad \text{for all } i \in [1, n_{k+1}].$$

BIN. The third layer applies binarization to the output \mathbf{z} of the previous layer, by implementing the sign function as follows:

$$o_i = \begin{cases} 1, & \text{if } z_i \geq 0, \\ -1, & \text{if } z_i < 0, \end{cases} \quad \text{for all } i \in [1, n_{k+1}].$$

The entire block can then be expressed as the constraints

$$o_i = \begin{cases} 1, & \text{if } \langle \mathbf{a}_i, \mathbf{x} \rangle \circ_{\text{rel}} C_i, \\ -1, & \text{otherwise,} \end{cases} \quad \text{for all } i \in [1, n_{k+1}], \quad (3.1)$$

where

$$C_i = -\frac{\sigma_i}{\alpha_i} \gamma_i + \mu_i - b_i$$

$$\circ_{\text{rel}} = \begin{cases} \geq, & \text{if } \alpha_i > 0, \\ \leq, & \text{if } \alpha_i < 0. \end{cases}$$

Let us recall that the input variables x_j and the output variables o_i take the values -1 and 1 . We need to replace them with the Boolean variables $x_j^{(b)}, o_i^{(b)} \in \{0, 1\}$ in order to further translate the constraints in (3.1) to the Boolean constraints

$$\sum_{j=1}^{n_k} l_{ij} \circ_{\text{rel}} D_i \Leftrightarrow o_i^{(b)}, \quad \text{for all } i \in [1, n_{k+1}],$$

where

$$l_{ij} = \begin{cases} x_j^{(b)}, & \text{if } j \in \mathbf{a}_i^+, \\ \neg x_j^{(b)}, & \text{if } j \in \mathbf{a}_i^-, \end{cases}$$

$$D_i = \begin{cases} [C_i'] + |\mathbf{a}_i^-|, & \text{if } \alpha_i > 0, \\ [C_i'] + |\mathbf{a}_i^-|, & \text{if } \alpha_i < 0, \end{cases}$$

$$C_i' = \left(C_i + \sum_j a_{ij} \right) / 2,$$

$$\mathbf{a}_i^+ = \{j \mid a_{ij} > 0\},$$

$$\mathbf{a}_i^- = \{j \mid a_{ij} < 0\}.$$

For further details on the derivation, see [31].

3.2. Encoding of the output block

The output block consists of a LIN layer followed by an ARGMAX layer. To encode ARGMAX, we need to encode an ordering relation over the outputs of the linear layer, and therefore we introduce the Boolean variables $d_{ii'}^{(b)}$ such that

$$\langle \mathbf{a}_i, \mathbf{x} \rangle + b_i \geq \langle \mathbf{a}_{i'}, \mathbf{x} \rangle + b_{i'} \Leftrightarrow d_{ii'}^{(b)}, \quad \text{for all } i, i' \in [1, s].$$

These constraints can be further translated into Boolean constraints, as proposed by Narodystka *et al.* in [31] and supplemented by us as follows:

$$\sum_{j=1}^{n_m} l_{ii'j} \geq E_{ii'} \Leftrightarrow d_{ii'}^{(b)}, \quad \text{for all } i, i' \in [1, s], \quad i \neq i',$$

where

$$l_{ii'j} = \begin{cases} x_j^{(b)}, & \text{if } j \in \mathbf{a}_{ii'}^+, \\ -x_j^{(b)}, & \text{if } j \in \mathbf{a}_{ii'}^-, \\ 0, & \text{otherwise,} \end{cases}$$

$$E_{ii'} = \left\lceil \left(b_{i'} - b_i + \sum_j a_{ij} - \sum_j a_{i'j} \right) / 4 \right\rceil + |\mathbf{a}_{ii'}^-|,$$

$$\mathbf{a}_{ii'}^+ = \{j \mid a_{ij} > 0 \wedge a_{i'j} < 0\},$$

$$\mathbf{a}_{ii'}^- = \{j \mid a_{ij} < 0 \wedge a_{i'j} > 0\}.$$

In the case of $i = i'$, $d_{ii'}^{(b)}$ must obviously be assigned to 1.

Finally, to encode ARGMAX, we have to pick the row in the matrix ($d_{ii'}$) which contains only 1s, as it can be encoded by the Boolean constraint

$$\sum_{i'} d_{ii'}^{(b)} = s \Leftrightarrow o_i^{(b)}, \quad \text{for all } i \in [1, s].$$

3.3. Encoding of the input binarization block

In our paper, and also in [31], experiments on checking adversarial robustness under the L_∞ norm are run on grayscale input images that are binarized by an additional BNBIN block before BLOCK₁. We now propose how this BNBIN block can be encoded to Boolean constraints.

Let $\alpha_0, \gamma_0, \mu_0, \sigma_0$ denote the parameters of the BN layer. Since adversarial robustness is about to be checked, the input $\mathbf{x} \in \mathbb{N}^{n_1}$ consists of constants, while the perturbation $\tau \in [-\epsilon, \epsilon]^{n_1}$ consists of integer variables and the output $\mathbf{o}^{(b)} \in \{0, 1\}^{n_1}$ consists of Boolean variables. The BNBIN block can be encoded by the constraints

$$\alpha_i \frac{x_i + \tau_i - \mu_i}{\sigma_i} + \gamma_i \geq 0 \Leftrightarrow o_i^{(b)}, \quad \text{for all } i \in [1, n_1], \quad (3.2)$$

where $\alpha_i, \gamma_i, \mu_i, \sigma_i$ denote the i^{th} element of the vectors $\alpha_0, \gamma_0, \mu_0, \sigma_0$, respectively.

The constraints in (3.2) further translate to

$$x_i + \tau_i - \mu_i + \frac{\sigma_i \gamma_i}{\alpha_i} \circ_{\text{rel}} 0 \Leftrightarrow o_i^{(b)}, \quad (3.3)$$

where

$$\circ_{\text{rel}} = \begin{cases} \geq, & \text{if } \alpha_i > 0, \\ \leq, & \text{if } \alpha_i < 0. \end{cases}$$

Then (3.3) translates to

$$\tau_i \circ_{\text{rel}} B_i \Leftrightarrow o_i^{(b)}, \quad (3.4)$$

where

$$B_i = \begin{cases} \lceil B'_i \rceil, & \text{if } \alpha_i > 0, \\ \lfloor B'_i \rfloor, & \text{if } \alpha_i < 0, \end{cases}$$

$$B'_i = \mu_i - x_i - \frac{\sigma_i \gamma_i}{\alpha_i}.$$

Since τ_i is in the given range $[-\epsilon, \epsilon]$, we can represent it as a *bit-vector* of a given bit-width. In order to apply *unsigned* bit-vector arithmetic, we translate the domain of τ_i into $[0, 2\epsilon]$. Thus, we can represent τ_i as a bit-vector variable of bit-width $w = \lceil \log_2(2\epsilon + 1) \rceil$ and apply unsigned bit-vector arithmetic to (3.4) as follows:

$$\tau_i^{[w]} \circ_{\text{rel}}^{\text{u}} (B_i + \epsilon)^{[w]} \Leftrightarrow o_i^{(b)}, \quad (3.5)$$

where $\circ_{\text{rel}}^{\text{u}}$ denotes the corresponding unsigned bit-vector relational operator `bvule` or `bvule`, respectively, and the bound $B_i + \epsilon$ is represented as a bit-vector constant of bit-width w . For the syntax and semantics of common bit-vector operators, see [24].

The constraints in (3.5) are not even needed to add in certain cases:

- if $B_i \leq -\epsilon$, then assign $o_i^{(b)}$ to 1 if $\alpha_i > 0$, and to 0 if $\alpha_i < 0$;
- if $B_i > \epsilon$, then assign $o_i^{(b)}$ to 0 if $\alpha_i > 0$, and to 1 if $\alpha_i < 0$.

Some further constraints are worth to add to restrict the domain of τ_i :

$$\begin{aligned} \tau_i^{[w]} &\geq^{\text{u}} 0^{[w]} \\ \tau_i^{[w]} &\leq^{\text{u}} (2\epsilon)^{[w]} \\ \tau_i^{[w]} &\geq^{\text{u}} (\epsilon - x_i)^{[w]}, \text{ if } x_i < \epsilon \\ \tau_i^{[w]} &\leq^{\text{u}} (\epsilon + \max_x - x_i)^{[w]}, \text{ if } x_i > \max_x - \epsilon \end{aligned} \quad (3.6)$$

where \max_x is the highest possible value for the input values in \mathbf{x} .²

In our tool, all the bit-vector constraints in (3.5) and (3.6) are bit-blasted into CNF.

3.4. Encoding of BNN properties

In this paper, we focus on the properties defined in Section 2, namely adversarial robustness and network equivalence.

²In our experiments, the input represents pixels of grayscale images, therefore $\max_x = 255$.

3.4.1. Adversarial robustness

We assume that the BNN consists of an input binarization block, internal blocks and an output block. Let $\text{BNN}(\mathbf{x} + \boldsymbol{\tau}, \mathbf{o}^{(b)})$ denote the encoding of the whole BNN over the perturbed input $\mathbf{x} + \boldsymbol{\tau}$ and the output $\mathbf{o}^{(b)}$. Note that $\mathbf{x} \in \mathbb{N}^{n_1}$ is an input from the the training or test set, therefore its ground truth label $\ell(\mathbf{x})$ is given. On the other hand, the perturbation $\boldsymbol{\tau} \in [-\epsilon, \epsilon]^{n_1}$ consists of integer variables. The output $\mathbf{o}^{(b)} \in \{0, 1\}^s$ consists of Boolean variables. Basically, we are looking for a satisfying assignment for the perturbation variables $\boldsymbol{\tau}$ such that the BNN outputs a label different from $\ell(\mathbf{x})$. Thus, checking adversarial robustness translates into checking the satisfiability of the following constraint:

$$\text{BNN}(\mathbf{x} + \boldsymbol{\tau}, \mathbf{o}^{(b)}) \wedge \neg o_{\ell(\mathbf{x})}^{(b)}.$$

3.4.2. Network equivalence

We want to check if two BNNs classify binarized inputs completely the same. Therefore we assume that those BNNs do not have BNBIN blocks, or if they do, then they apply the same BNBIN block. Therefore, let $\text{BNN}_1(\mathbf{x}^{(b)}, \mathbf{o}_1^{(b)})$ and $\text{BNN}_2(\mathbf{x}^{(b)}, \mathbf{o}_2^{(b)})$ denote the encoding of the internal blocks and the output block of the two BNNs, respectively, over the same binary input $\mathbf{x}^{(b)}$. Checking the equivalence of those BNNs translates into checking the satisfiability of the following constraint:

$$\text{BNN}_1(\mathbf{x}, \mathbf{o}_1^{(b)}) \wedge \text{BNN}_2(\mathbf{x}, \mathbf{o}_2^{(b)}) \wedge \mathbf{o}_1^{(b)} \neq \mathbf{o}_2^{(b)}.$$

We translate the inequality $\mathbf{o}_1^{(b)} \neq \mathbf{o}_2^{(b)}$ over vectors of Boolean variables into

$$\neg(o_{1,1}^{(b)} \Leftrightarrow o_{2,1}^{(b)}) \vee \dots \vee \neg(o_{1,s}^{(b)} \Leftrightarrow o_{2,s}^{(b)})$$

which can then be further translated to a set of clauses by using Tseitin transformation.

4. Encoding of clauses and Boolean cardinality constraints

In Section 3, we proposed an encoding of BNNs into *clauses* $l_1 \vee \dots \vee l_n$ as well as *equivalences* over Boolean cardinality constraints in the form

$$l \Leftrightarrow \sum_{i=1}^n l_i \geq c, \tag{4.1}$$

where l, l_1, \dots, l_n are literals and $c \in \mathbb{N}$ is a constant where $0 \leq c \leq n$. Note that our encoding applies “*AtMost*” Boolean cardinality constraints as well. Such a constraint $\sum_{i=1}^n l_i \leq c$ can always be translated to an “*AtLeast*” constraint $\sum_{i=1}^n \neg l_i \geq n - c$.

Depending on the approaches one wants to apply to the satisfiability checking of those constraints, they have to be encoded in different ways.

4.1. Encoding into SAT

There are various existing, well-known approaches expressing Boolean cardinality constraints into Boolean logic, for example by using sequential counters [35], cardinality networks [1] or modulo totalizers [30, 33].

Sequential counters [35] encode an “AtLeast” Boolean cardinality constraint into the following Boolean formula:

$$\begin{aligned}
 & (l_1 \Leftrightarrow v_{1,1}) \\
 \wedge & \quad \neg v_{1,j} && \text{for } j \in [2, c], \\
 \wedge & \quad (v_{i,1} \Leftrightarrow l_i \vee v_{i-1,1}) && \text{for } i \in [2, n], \\
 \wedge & \quad (v_{i,j} \Leftrightarrow (l_i \wedge v_{i-1,j-1}) \vee v_{i-1,j}) && \text{for } i \in [2, n], j \in [2, c].
 \end{aligned}$$

All the Boolean variables $v_{i,j}$ are introduced as fresh variables and the formula above can be converted into its CNF [35]. On the top of that, to encode the constraint (4.1), we only need to additionally encode the formula $l \Leftrightarrow v_{n,c}$.

Cardinality networks [1] yield another, refined approach for encoding Boolean cardinality constraints. For improving reasoning about cardinality constraints encoded, for example, using sequential counters, a cardinality network encoding of a cardinality constraint divides the cardinality constraint into multiple instances of the base operations *half sorting* and *simplified half merging*, which basically work as building blocks.

The *modulo totalizer* cardinality encoding [33] and its variant for k -cardinality [30] improve the above described approach based on cardinality network, especially in connection with MaxSAT solving. The modulo totalizer approach of [33] addresses limitations of the half sorting cardinality network approach from [1], by using totalizer encodings from [3] in order to reduce the number of variables during CNF encodings. The modulo totalizer cardinality encoding of [33] decreases the number of clauses used in [3], and hence improves cardinality network encodings during constraint propagation.

4.2. Encoding into SMT

It is straightforward to encode clauses and constraints (4.1) into SMT over the logic QF_LIA. We would like to note that bit-vector constraints (3.5), (3.6) are bit-blasted into CNF in our tool and then added as clauses, even when being encoded into SMT. As future work, one could try to solve all the constraints over the logic QF_BV.

4.3. Encoding into Boolean cardinality constraints

The encoding that we proposed for BNNs consists of *clauses* on the one hand, and *equivalences* over Boolean cardinality constraints in the form (4.1) on the other hand. We show how to encode both type of constraints into a set of Boolean cardinality constraints.

A clause $l_1 \vee \dots \vee l_n$ can be encoded as the Boolean cardinality constraint $\sum_{i=1} l_i \geq 1$.

A constraint (4.1) can be unfolded into two implications (assume $c > 0$):

$$\begin{aligned} l &\Rightarrow \sum_{i=1} l_i \geq c, \\ \neg l &\Rightarrow \sum_{i=1} l_i \leq c - 1. \end{aligned} \tag{4.2}$$

By following the idea on the GitHub page³ of the SAT solver MINICARD [27], an implied Boolean cardinality constraints (4.2) can be translated to a (non-implied) Boolean cardinality constraint

$$\sum_{i=1} l_i + \underbrace{\neg l + \dots + \neg l}_c \geq c, \tag{4.3}$$

which can then be solved by cardinality solvers with duplicated-literal handling, such as MINICARD.

4.4. Encoding into pseudo-Boolean constraints

The Boolean cardinality encoding from Section 4.3 can be fed into pseudo-Boolean solvers as well. The Boolean cardinality constraint (4.3) can naturally be translated to a pseudo-Boolean constraint $\sum_{i=1} l_i + \neg l \cdot c \geq c$.

5. Implementation

All the encodings described in the previous sections are implemented in Python, as part of our solver. Since our solver is a portfolio solver, it executes different kind of solvers (SAT, SMT, MIP) in parallel, by instantiating `ProcessPool` from the Python module `pathos multiprocessing` [29], which can run jobs with a non-blocking and unordered map.

The Python package PYSAT [21] provides a unified API to several SAT solvers such as MINISAT [12], GLUCOSE [2] and LINGELING [6]. PYSAT also supports a lot of encodings for Boolean cardinality constraints, including sequential counters [35], cardinality networks [1] and modulo totalizer [30, 33]. Furthermore, PYSAT offers API to the SAT solver MINICARD [27], which handles Boolean cardinality constraints *natively* on the level of watched literals and conflict analysis, instead of translating them into CNF.

In a similar manner, the Python package PYSMT [16] provides a unified API to several SMT solvers, such as MATHSAT [8], Z3 [9], CVC4 [4] and YICES [10].

The Python package MIP provides tools to solve mixed-integer linear programming instances and provides a unified API to MIP solvers such as CLP, CBC and GUROBI.

³<https://github.com/liffiton/minicard>

When running our portfolio solver, one can easily choose the solvers to execute in parallel, by using the following command-line arguments:

- sat-solver**. Choose any SAT solver supported by the PYSAT package such as MINISAT, GLUCOSE, etc., including MINICARD, or disable this option by using the value `none`.
- smt-solver**. Choose any SMT solver supported by PYSMT such as Z3, MATHSAT, etc., or disable this option by using the value `none`. Note that you might need to install the corresponding SMT solver for PYSMT by using the `pysmt-install` command.
- mip-solver**. Choose any MIP solver supported by the MIP package, most importantly GUROBI, or disable this option by using the value `none`. Note that you might need to purchase a license for GUROBI.
- card-enc**. Choose any cardinality encoding supported by the PYSAT package such as sequential counters, cardinality networks, modulo totalizer, k -cardinality modulo totalizer, etc., or disable this option by using the value `none`.
- timeout**. Set the timeout in seconds.

Our solver consists of two Python programs `bnn_adv_robust_check.py` and `bnn_eq_check.py` to check adversarial robustness and network equivalence, respectively. If `bnn_adv_robust_check.py` returns UNSAT, then the given input image is considered to be robust under the given maximal perturbation value passed as a command-line argument. In case of SAT answer, the tool displays the perturbed input values and the label resulted by misclassification.

If `bnn_eq_check.py` returns UNSAT, then the two given BNNs are considered to be equivalent. In case of SAT answer, the tool displays the common input values for which the BNNs return different outputs, which are also displayed. Note that an output is displayed as a list of Boolean literals among which the single positive literal represents the output label.

6. Experiments and results

Our experiments were run on Intel i5-7200U 2.50 GHz CPU (2 cores, 4 threads) with 8 GB memory. The time limit was set to 300 seconds.

In our experiments, the BNN architecture is the same as in the experiments in [31]: it consists of 4 internal blocks and 1 output block. Each internal block contains a LIN layer with 200, 100, 100 and 100 neurons, respectively. We use an additional HARDTANH layer only during the training of the network. We trained the network on the MNIST dataset [26]. The accuracy of the resulting network is 93%.

6.1. Verifying adversarial robustness

In the first set of experiments, we focused on the important problem of checking adversarial robustness under the L_∞ norm. From the MNIST dataset, we randomly picked 20 images (from the test set) that were correctly classified by the network for each of the 10 classes. This resulted in a set of 200 images that we consider in our experiments on adversarial robustness. We experimented with three different maximum perturbation values by varying $\epsilon \in \{1, 3, 5\}$.

To process the inputs, we add a BNIN block to the BNN before BLOCK₁. The BNIN block applies binarization to the grayscale MNIST images. We would like emphasize that our experiments did not apply any additional preprocessing, as opposed to the experiments in [31] that first try to perturb only the top 50% of highly salient pixels in the input image. Furthermore, our solver does not apply any additional search procedure on the top of the solvers being run in parallel, as opposed to the experiments in [31] that apply a counterexample-guided (CEG) search procedure based on Craig interpolation. In this sense, our solver explores the search space without applying any additional procedures.

Figure 2 shows some of the results of our experiments. Each column shows the number of solved instances out of the 200 selected instances and the average runtime in seconds. The bar chart under certain cells shows the distribution of different solvers providing the results. The bottom charts present the results in a more detailed way, where the distribution of runtimes suggests that our solver can solve ca. 85–95% of the instances in less than 30 seconds.

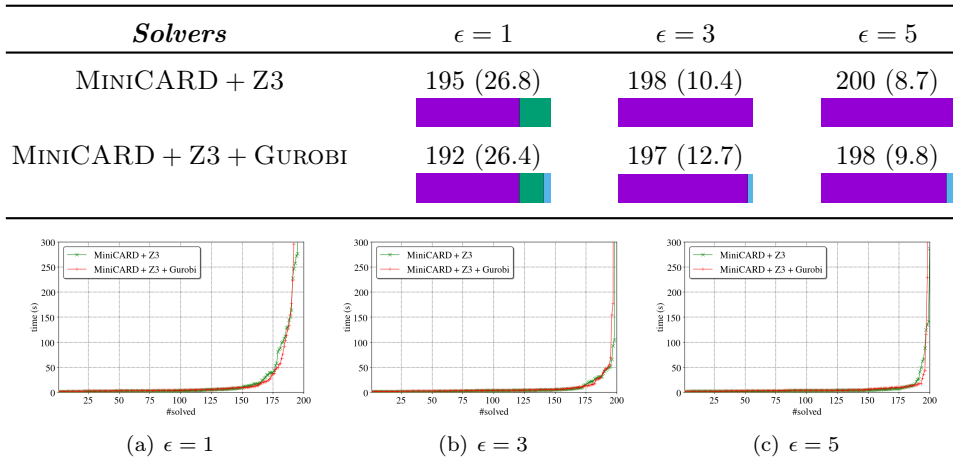


Figure 2. Results on checking adversarial robustness of 4-BLOCK BNN on MNIST dataset, for different maximum perturbation values ϵ . Colors represent the ratio of solved instances by different solvers: purple for MINICARD, green for Z3, blue for GUROBI.

As the figure shows, our solver produced the best results when running

MINICARD as a SAT solver and Z3 as an SMT solver in parallel. Since, in our preliminary experiments, GUROBI had showed promising performance, we also ran experiments with GUROBI parallel to MINICARD and Z3. Of course, we also tried different combinations of solvers in our experiments, but we found the ones in the table the most promising.

In order to investigate how our solver scales for larger BNNs, we constructed another BNN with 5 internal blocks containing LIN layers of size 300, 200, 150, 100 and 100, respectively, and trained it on the MNIST dataset. The accuracy of the resulting network is 94%. Figure 3 shows the results of our corresponding experiments.

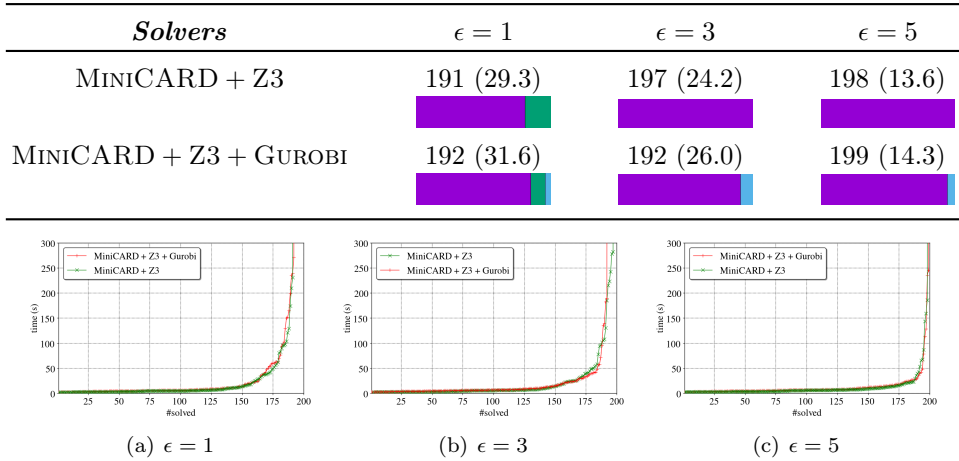


Figure 3. Results on checking adversarial robustness of 5-BLOCK BNN on MNIST dataset.

6.2. Verifying network equivalence

In the second set of experiments, we focused on the problem of checking network equivalence. From our 4-BLOCK BNN trained to classify MNIST images, we generated 20 slightly different variants by altering a few weights in the network. For this sake, we randomly flip $\delta > 0$ weights in A_m . Then, we run our solver to check if the original BNN is equivalent with an altered variant. Since the aim was to generate difficult benchmark instances, i.e., which are “almost UNSAT”, we chose small values for δ . Figure 4 shows the results of our corresponding experiments.

6.3. Side notes

In our solver’s source code, there exist implemented features that are not yet accessible due to the lack of API features of certain Python packages. Although PySAT’s CNF encodings of Boolean cardinality constraints are accessible via our

<i>Solvers</i>	$\delta = 2$	$\delta = 5$
MINICARD + Z3	19 (92.3)	20 (64.5)
MINICARD + Z3 + GUROBI	17 (124.5)	19 (77.1)

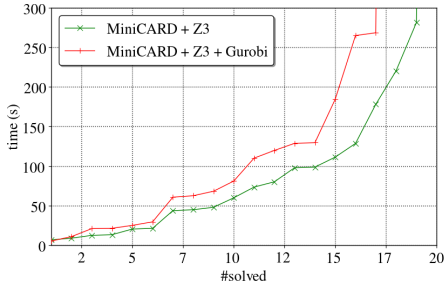
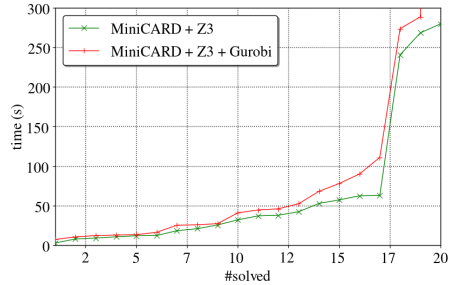
(a) $\delta = 2$ (b) $\delta = 5$

Figure 4. Results on checking network equivalence, for different δ values.

solver’s command-line argument `--card-enc`, equivalences (4.1) cannot directly be dealt with PYSAT since the output variable of a CNF encoding cannot be accessed through PYSAT’s API. For instance, we would need to access the Boolean variable $v_{n,c}$ when using sequential counter encoding as described in Section 4.1. Therefore, in our solver’s current version, each equivalence (4.1) is first encoded into a pair of Boolean cardinality constraints as described in Section 4.3, and the resulting cardinality constraints are then encoded into CNF. Note that encoding equivalences (4.1) directly into Boolean logic would result in more easy-to-solve instances, once PYSAT allows. In the latter case, on the other hand, the encoding into CNF might dominate the runtime, since millions of variables and millions of clauses are generated even for our 4-BLOCK BNN.

7. Conclusions

We introduced a new portfolio-style solver to verify important properties of binarized neural networks such as adversarial robustness and network equivalence. Our solver encodes those BNN properties, as we propose SAT, SMT, cardinality and pseudo-Boolean encodings in the paper. Our experiments demonstrated that our solver was capable of verifying adversarial robustness of medium-sized BNNs on the MNIST dataset in reasonable time and seemed to scale for larger BNNs. We also ran experiments on network equivalence with impressive results on the SAT instances.

After we submitted this paper, K. Jia and M. Rinard have recently published a paper about a framework for verifying robustness for BNNs [22]. They devel-

oped a SAT solver with native support for reified cardinality constraints and, also, proposed strategies to train BNNs such that weight matrices were sparse and cardinality bounds low. Based on their experimental results, their solver might outperform our solver on their benchmarks. As part of future work, we would like to run experiments with both solvers on those benchmarks.

We will try to overcome the problems that originate in using the PySAT Python packages, in order to make already implemented “hidden” features accessible for users. Furthermore, we are planning to extend the palette of solvers with Google’s OR-Tools, which look promising based on our preliminary experiments.

References

- [1] R. ASÍN, R. NIEUWENHUIS, A. OLIVERAS, E. RODRÍGUEZ-CARBONELL: *Cardinality Networks: a theoretical and empirical study*, Constraints 16.2 (2011), pp. 195–221.
- [2] G. AUDEMARD, L. SIMON: *Lazy Clause Exchange Policy for Parallel SAT Solvers*, in: Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT), vol. 8561, Lecture Notes in Computer Science, Springer, 2014, pp. 197–205.
- [3] O. BAILLEUX, Y. BOUFKHAD: *Efficient CNF Encoding of Boolean Cardinality Constraints*, in: Proc. 9th International Conference on Principles and Practice of Constraint Programming (CP), 2003, pp. 108–122.
- [4] C. BARRETT, C. L. CONWAY, M. DETERS, L. HADAREAN, D. JOVANOVIĆ, T. KING, A. REYNOLDS, C. TINELLI: *CVC4*, in: Proc. Int. Conf. on Computer Aided Verification (CAV), vol. 6806, Lecture Notes in Computer Science, Springer, 2011, pp. 171–177.
- [5] C. BARRETT, P. FONTAINE, C. TINELLI: *The Satisfiability Modulo Theories Library (SMT-LIB)*, www.SMT-LIB.org, 2016.
- [6] A. BIÈRE: *CaDiCaL, Lingeling, Plingeling, Treengeling, YalSAT Entering the SAT Competition 2017*, in: Proc. of SAT Competition 2017 – Solver and Benchmark Descriptions, vol. B-2017-1, Department of Computer Science Series of Publications B, University of Helsinki, 2017, pp. 14–15.
- [7] C. CHENG, G. NÜHRENBURG, H. RUESS: *Verification of Binarized Neural Networks (2017)*, arXiv: 1710.03107.
- [8] A. CIMATTI, A. GRIGGIO, B. SCHAAFSMA, R. SEBASTIANI: *The MathSAT5 SMT Solver*, in: Proc. Int. Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), ed. by N. PITERMAN, S. SMOLKA, vol. 7795, Lecture Notes in Computer Science, Springer, 2013, pp. 93–107.
- [9] L. DE MOURA, N. BJØRNER: *Z3: An Efficient SMT Solver*, in: Proc. Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), TACAS’08/ETAPS’08, Springer-Verlag, 2008, pp. 337–340.
- [10] B. DUTERTRE: *Yices 2.2*, in: Proc. Int. Conf. on Computer-Aided Verification (CAV), ed. by A. BIÈRE, R. BLOEM, vol. 8559, Lecture Notes in Computer Science, Springer, 2014, pp. 737–744.
- [11] S. DUTTA, S. JHA, S. SANKARANARAYANAN, A. TIWARI: *Output Range Analysis for Deep Feedforward Neural Networks*, in: NASA Formal Methods, Springer, 2018, pp. 121–138.
- [12] N. EÉN, N. SÖRENSON: *An Extensible SAT-solver*, in: Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT), vol. 2919, Lecture Notes in Computer Science, Springer, 2004, pp. 502–518.
- [13] R. EHLERS: *Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks*, in: Automated Technology for Verification and Analysis, Springer, 2017, pp. 269–286.

- [14] EU DATA PROTECTION REGULATION: *Regulation (EU) 2016/679 of the European Parliament and of the Council*, 2016.
- [15] M. FISCHETTI, J. JO: *Deep Neural Networks and Mixed Integer Linear Optimization*, *Constraints* 23 (3 2018), pp. 296–309, DOI: <https://doi.org/10.1007/s10601-018-9285-6>.
- [16] M. GARIO, A. MICHELI: *PySMT: a solver-agnostic library for fast prototyping of SMT-based algorithms*, in: *International Workshop on Satisfiability Modulo Theories (SMT)*, 2015.
- [17] I. GOODFELLOW, Y. BENGIO, A. COURVILLE: *Deep Learning*, The MIT Press, 2016, ISBN: 0262035618.
- [18] B. GOODMAN, S. R. FLAXMAN: *European Union Regulations on Algorithmic Decision-Making and a “Right to Explanation”*, *AI Magazine* 38.3 (2017), pp. 50–57.
- [19] X. HUANG, M. KWIATKOWSKA, S. WANG, M. WU: *Safety Verification of Deep Neural Networks*, in: *Computer Aided Verification*, Springer, 2017, pp. 3–29.
- [20] I. HUBARA, M. COURBARIAUX, D. SOUDRY, R. EL-YANIV, Y. BENGIO: *Binarized Neural Networks*, in: *Advances in Neural Information Processing Systems* 29, Curran Associates, Inc., 2016, pp. 4107–4115.
- [21] A. IGNATIEV, A. MORGADO, J. MARQUES-SILVA: *PySAT: A Python Toolkit for Prototyping with SAT Oracles*, in: *Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT)*, vol. 10929, Lecture Notes in Computer Science, Springer, 2018, pp. 428–437.
- [22] K. JIA, M. RINARD: *Efficient Exact Verification of Binarized Neural Networks* (2020), arXiv: 2005.03597 [cs.AI].
- [23] G. KATZ, C. W. BARRETT, D. L. DILL, K. JULIAN, M. J. KOCHENDERFER: *Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks*, in: *CAV*, 2017, pp. 97–117, DOI: https://doi.org/10.1007/978-3-319-63387-9_5.
- [24] G. KOVÁSZNAI, A. FRÖHLICH, A. BIÈRE: *Complexity of Fixed-Size Bit-Vector Logics*, *Theory of Computing Systems* 59 (2016), pp. 323–376, ISSN: 1433-0490, DOI: <https://doi.org/10.1007/s00224-015-9653-1>.
- [25] J. KUNG, D. ZHANG, G. VAN DER WAL, S. CHAI, S. MUKHOPADHYAY: *Efficient Object Detection Using Embedded Binarized Neural Networks*, *Journal of Signal Processing Systems* (2017), pp. 1–14.
- [26] Y. LECUN, L. BOTTOU, Y. BENGIO, P. HAFFNER: *Gradient-Based Learning Applied to Document Recognition*, *Proceedings of the IEEE* 86.11 (Nov. 1998), pp. 2278–2324.
- [27] M. H. LIFFITON, J. C. MAGLALANG: *More Expressive Constraints for Free*, in: *Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT)*, vol. 7317, Lecture Notes in Computer Science, Springer, 2012, pp. 485–486.
- [28] B. MCDANEL, S. TEERAPITTAYANON, H. T. KUNG: *Embedded Binarized Neural Networks*, in: *EWSN*, Junction Publishing, Canada / ACM, 2017, pp. 168–173.
- [29] M. M. MCKERNS, L. STRAND, T. SULLIVAN, A. FANG, M. A. AIVAZIS: *Building a framework for predictive science* (2012), arXiv: 1202.1056.
- [30] A. MORGADO, A. IGNATIEV, J. MARQUES-SILVA: *MSCG: Robust Core-Guided MaxSAT Solving*, *JSAT* 9 (2014), pp. 129–134.
- [31] N. NARODYTSKA, S. KASIVISWANATHAN, L. RYZHYK, M. SAGIV, T. WALSH: *Verifying Properties of Binarized Deep Neural Networks*, in: *32nd AAAI Conference on Artificial Intelligence*, 2018, pp. 6615–6624.
- [32] NIPS IML SYMPOSIUM: *NIPS Interpretable ML Symposium*, Dec. 2017.
- [33] T. OGAWA, Y. LIU, R. HASEGAWA, M. KOSHIMURA, H. FUJITA: *Modulo Based CNF Encoding of Cardinality Constraints and Its Application to MaxSAT Solvers*, in: *25th International Conference on Tools with Artificial Intelligence (ICTAI)*, IEEE, 2013, pp. 9–17.

-
- [34] G. SINGH, T. GEHR, M. PÜSCHEL, M. T. VECHEV: *Boosting Robustness Certification of Neural Networks*, in: 7th International Conference on Learning Representations, OpenReview.net, 2019.
 - [35] C. SINZ: *Towards an Optimal CNF Encoding of Boolean Cardinality Constraints*, in: Proc. Principles and Practice of Constraint Programming (CP), Springer, 2005, pp. 827–831.
 - [36] V. TJENG, K. Y. XIAO, R. TEDRAKE: *Evaluating Robustness of Neural Networks with Mixed Integer Programming*, in: 7th International Conference on Learning Representations, OpenReview.net, 2019.
 - [37] T. WENG, H. ZHANG, H. CHEN, Z. SONG, C. HSIEH, L. DANIEL, D. S. BONING, I. S. DHILLON: *Towards Fast Computation of Certified Robustness for ReLU Networks*, in: ICML, 2018, pp. 5273–5282.

A contribution to scheduling jobs submitted by finite-sources in computational clusters*

Attila Kuki, Tamás Bérczes, Ádám Tóth, János Sztrik

University of Debrecen, Hungary

{kuki.attila,berczes.tamas,toth.adam,sztrik.janos}@inf.unideb.hu

Submitted: December 21, 2020

Accepted: March 17, 2021

Published online: May 18, 2021

Abstract

Data science and data processing are very popular topics nowadays. Unlike a few years ago, everything is connected to data now and we have to handle these kinds of large data well. Therefore the distributed heterogeneous resources of networks e.g. the computational grid, have attracted great interest. It has become a challenge to schedule jobs in order to utilize the available resources effectively. The allocation of arriving jobs has a great impact on the efficiency and the energy consumption of the system.

A generalized finite source model is presented in this paper. Our main goal is to build up models for the performance evaluation of scheduling compute-intensive jobs with unknown service times in a computational cluster that consists of servers of different types. For this purpose we determine various performance measures for all combinations of three scheduling policies (two of them are the novelty of this paper: the MRT and the MRTHP policies) which can be used for assigning jobs to servers with three schemes for buffering arriving jobs. Furthermore, we investigate the effect of switching off idle servers on the energy consumption of the system under these combinations of scheduling policies and buffering schemes.

Computational results obtained by simulation show that the choice of the scheduling policy and the buffering scheme plays an important role in ensuring the quality of service parameters such as the waiting time and the

*The research work was supported by the construction EFOP - 3.6.3 - VEKOP - 16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund.

The research work was supported by the Austro-Hungarian Cooperation Grant No 106öu4, 2020.

response time experienced in the case of arriving jobs. However, the energy consumption is only affected by the scheduling policy and the energy saving mode, while the buffering scheme does not have a significant impact.

Keywords: Computational cluster, performance evaluation, buffering scheme, finite-source queueing systems

AMS Subject Classification: 68M10, 68M20

1. Introduction

This paper deals with scheduling jobs in heterogeneous resources of networks, e.g. the computational clusters. In the literature various job allocation algorithms have been proposed to schedule arriving jobs in computational clusters [1, 6, 7]. In addition, some algorithms have been designed to consider knowledge about the characteristics of jobs; these algorithms may be classified as either clairvoyant or as non-clairvoyant [15–17].

Besides the effective scheduling, the energy consumption of such grid systems turns into a crucial requirement due to the rapid increase of the size of the grid and the goal of a green computational cluster. The most common techniques of reducing energy consumption are related to the dynamic power management used at runtime. It is therefore of interest to examine algorithms that offer the greatest performance while using an amount of energy that is as low as possible.

Do introduced a generalized infinite model for the performance evaluation of scheduling compute-intensive jobs with unknown service times in computational clusters [2, 14]. In this paper we use a finite model instead of the infinite one [12] to make the queueing model more realistic and we introduce two new scheduling policies: in addition to the previously introduced High Performance priority policy, we also consider a Mean Response Time priority and a Mean Response time with High Performance priority policy. We investigate these policies with respect to three schemes of buffering the arriving jobs: Separate Queue, Class Queue, and Common Queue. The novelty of this paper is introducing these two new policies. To our knowledge, this is the first time when such a detailed investigation of scheduling policies of this kind has been performed. Among the three mentioned buffering schemes the Common Queue proved to be the most efficient. The novelty of this paper is to develop two new scheduling policies: the MRT and the MRTHP policies. With these two new policies the performance measures reach and overcome the characteristics of the Separate Queue.

The state space of the describing Markov chain is very large, thus it is rather difficult to calculate the system measures in the traditional way of writing down and solving the underlying steady-state equations. To obtain the performance measures we used SimPack, a collection of C/C++ libraries and executable programs for computer simulation [3]. In this collection several simulation algorithms are supported including discrete event simulation, continuous simulation, and combined (multi-model) simulation. The purpose of the SimPack toolkit is to provide the user a set of utilities that illustrate the basics of building a working simulation

from a model description. Simulation results show that between the newly applied algorithms the MRTHP is capable of lessening the difference of the performance measures of the buffering schemes. In the case of the Separate Queue, MRTHP significantly decreases important factors such as the mean waiting time and the mean response time. Furthermore, we study the effect of scheduling policies and buffering policies on the energy consumption of a system that switches off idle servers with and without an energy saving mode. According to the obtained results, the energy consumption of the different scheduling algorithms is relatively identical.

Some related investigations are described in [13]. Using the techniques described in this paper, it would be worth applying finite-source models for those problems, as well.

The rest of the paper is organized as follows: Section 2 describes the corresponding queueing model with components to study the behavior of the computational clusters and the derivation of the main steady-state performance measures. In Section 3 we show some numerical results that were derived by simulation with SimPack and subsequently visualized as diagrams. Section 4 presents our conclusions.

2. System model

A cluster is considered that serves compute-intensive jobs according to the following characteristics:

- Every job can be executed on any server.
- Jobs are served according to FIFO (first in, first out) policy.
- The service times of jobs are unknown to the local scheduler.
- Jobs under service cannot be interrupted (non-preemption);
- Jobs are atomic, i.e., they can not be divided into smaller pieces;

We assume, furthermore, that jobs arrive to the system from a finite number N of sources and that each source generates jobs according to an exponential distribution with parameter λ ; thus the maximum rate of the incoming jobs is $N \cdot \lambda$. Servers are organized in I classes with J servers per class. Service times, which denote the times required for the servers to execute jobs, are exponentially distributed with rate μ_i in class i . The exponentiality is not a strict constraint here. In real-life applications the arrival and service behaviours are often very close to the exponential behaviour. The service rate μ_{system} of the whole system can be thus defined as

$$\mu_{\text{system}} = \sum_{i=1}^I J \cdot \mu_i.$$

The system load ρ_{system} , the total amount of traffic carried by the system, can be written as the ratio of between the arriving and the service rate:

$$\rho_{\text{system}} = \frac{\lambda \cdot \bar{N}}{\mu_{\text{system}}},$$

where \bar{N} is the average number of jobs in the system. Because the numbers of sources of the considered model is finite, the stationary distributions always exist, which implies the stability of the system.

2.1. Scheduling policies

Furthermore, we assume that every server is attached to a queue that buffers arriving jobs and from which the server removes jobs for execution (multiple servers may share a queue, see Section 2.2 for the various buffering schemes considered). We investigate in our model the following policies for scheduling arriving jobs to server queues:

- *HP (High Performance priority)*: This policy chooses the shortest queue in the system. If there is more than one queue with this property, a queue whose server has the highest performance is chosen.
- *MRT (Mean Response Time priority)*: This policy first calculates the expected mean response time for every queue and then selects a queue where this value is minimal.
- *MRTHP (Mean Response Time with High Performance priority)*: This policy is a combination of MRT and HP. If there is an idle server, it behaves like the HP policy; if all servers are busy, it behaves like MRT.

The comparison of these policies and the effect of MRT and MRTHP policies to the performance measures and the energy efficiency are discussed in sections 3.1 and 3.2.

In order to obtain the performance, mean response times, and energy consumption of a server, we consider every server of the cluster to be of a specific type (class). Let S denote the set of server classes and $I = |S|$ the size of S . Let $s \in S$ be a server class which can be characterized by the following parameters:

- C_s : This is the throughput of the server i.e., the number of completed operations per time; it is measured in `ssj_ops` according to the `SPECpower_ssj2008` benchmark [11].
- $P_{\text{ac},s}$: This is the average active power of the server under full load; it is measured in Watt according to the `SPECpower_ssj2008` benchmark.
- $P_{\text{id},s}$: This is the power consumption of the server in the idle state; it is measured in Watt according to the `SPECpower_ssj2008` benchmark.

It is presumed that when a server of class s is busy, then it works with throughput C_s and power consumption $P_{ac,s}$. According to SPECpower_ssj2008, the ratio $C_s/P_{ac,s}$ describes the energy efficiency of the server; higher ratio means higher efficiency. When the server becomes idle, the internal clock of the CPU is stopped via software such that the server consumes power $P_{id,s} < P_{ac,s}$; alternatively, idle servers may be completely switched off such that they do not consume power at all.

To choose a server with the highest performance in the HP respectively MRTHP policy, a server of class s with the highest value of C_s is selected; to choose a server with the smallest mean response time in the MRT respectively MRTHP policy, a server of class s with the smallest ratio q/C_s of queue length q and throughput s is selected.

2.2. Buffering schemes

In the following subsections, we present the various schemes for buffering arriving jobs and how they implement the previously introduced scheduling policies.

2.2.1. Separate Queue

In the Separate Queue scheme, every server has its own queue as depicted in Figure 1. Jobs are scheduled to the queue of a specific server according to the chosen policy, and they remain in that queue as long as the server is busy. If the server becomes idle, then it receives the first waiting job from its queue.

Henceforth let c_{ij} denote the status of server j in class i (0 denotes idle, 1 denotes busy or not in class i), and let q_{ij} denote the number of jobs in the queue of that server (which can range from 0 to $N - I \cdot J$). The state of the cluster at time t can be considered as a Continuous Time Markov Chain with dimension $I \cdot J + I \cdot J$: $X(t) = (c_{11}(t); \dots; c_{IJ}(t); q_{11}(t); \dots; q_{IJ}(t))$.

The system's steady-state probabilities can be defined the following way:

$$P(c_{11}; \dots; c_{IJ}; q_{11}; \dots; q_{IJ}) = \lim_{t \rightarrow \infty} P((c_{11}(t) = c_{11}; \dots; c_{IJ}(t) = c_{IJ}; \\ q_{11}(t) = q_{11}; \dots; q_{IJ}(t) = q_{IJ})$$

Since the state space of the describing Markov chain is very large, it is rather difficult to calculate the system measures in the traditional way of writing down and solving the underlying steady-state equations.

To obtain the performance measures we therefore used SimPack, a collection of C/C++ libraries and executable programs for computer simulation [3].

Using the simulation program the following important performance measures of the system can be calculated:

- R_{ij} – The probability that the server j in class i is busy,
- L_{ij} – The probability that the server j in class i is idle,

- Q_{ij} – The mean length of queue ij ,
- \bar{Q} – The mean number of jobs in the queues: $\bar{Q} = \sum_{i=1}^I \sum_{j=1}^J Q_{ij}$.

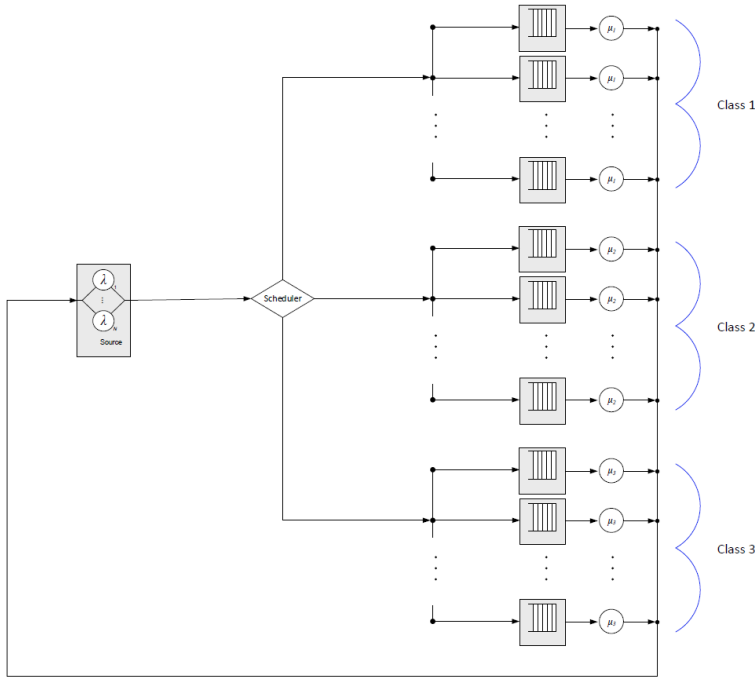


Figure 1. The Separate Queue scheme.

2.2.2. Class Queue

In the Class Queue scheme a buffer is assigned to each class (see Figure 2). Jobs are scheduled to the queue of a specific class according to the chosen policy, and they remain in that queue as long as all servers of the class are busy. If a server becomes idle, then it receives the first waiting job from the queue of its class.

Henceforth, let c_{ij} denote the status of server j in class i (0 means idle and 1 means busy) and let q_i denote the number of jobs in its queue (which can range from 0 to $N - I \cdot J$). The state of the cluster at time t can be considered as a Continuous Time Markov Chain with dimension $I \cdot J + I$: $X(t) = (c_{11}(t); \dots; c_{IJ}(t); q_1(t); \dots; q_I(t))$.

The system's steady-state probabilities can be defined the following way:

$$P(c_{11}; \dots; c_{IJ}; q_1; \dots; q_I) = \lim_{t \rightarrow \infty} P((c_{11}(t) = c_{11}; \dots; c_{IJ}(t) = c_{IJ};$$

$$q_1(t) = q_1; \dots; q_I(t) = q_I)$$

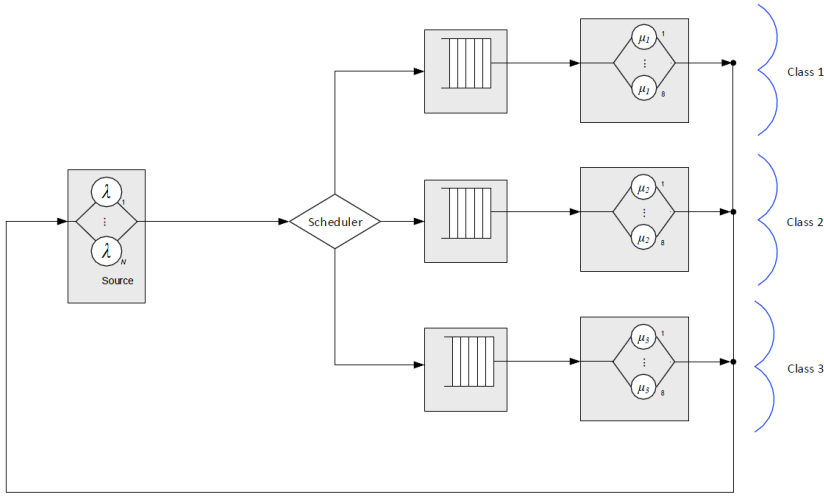


Figure 2. The Class Queue scheme.

Using the simulation program the following important performance measures of the system can be calculated:

- R_{ij} – The probability that server j in class i is busy,
- L_{ij} – The probability that server j in class i is idle,
- Q_i – The mean length of queue i ,
- \bar{Q} – The mean number of jobs in the queues: $\bar{Q} = \sum_{i=1}^I Q_i$.

2.2.3. Common Queue

In the Common Queue scheme, only a single common buffer is available for all servers (see Figure 3). If a job arrives, then its service begins immediately if at least one server is idle. If more than one server is idle, then the local scheduler chooses the server with the highest performance. If all the servers are busy, then the local scheduler places the job into the queue and the job remains there until one of the servers become idle.

Henceforth, let c_{ij} denote server j in class i (0 means idle and 1 means busy), and let q_1 denote the number of jobs in the queue (which can range from 0 to $N - I \cdot J$).

The state of the cluster at time t can be considered as a Continuous Time Markov Chain with dimension $I \cdot J + 1$: $X(t) = (c_{11}(t); \dots; c_{IJ}(t); q_1(t))$.

The system’s steady-state probabilities can be defined the following way:

$$P(c_{11}; \dots; c_{IJ}; q_1) = \lim_{t \rightarrow \infty} P(c_{11}(t) = c_{11}; \dots; c_{IJ}(t) = c_{IJ}; q_1(t) = q_1).$$

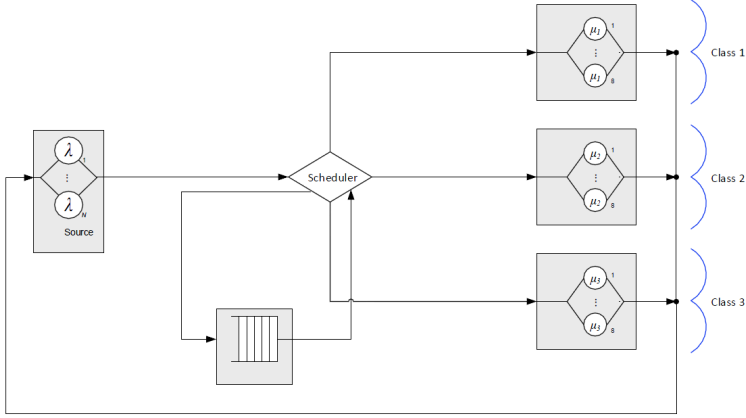


Figure 3. The Common Queue scheme.

Using the simulation program the following important performance measures of the system can be calculated:

- R_{ij} – The probability that server j in class i is busy,
- L_{ij} – The probability that server j in class i is idle,
- \bar{Q} – The mean number of jobs in the queue:

$$\bar{Q} = \sum_{c_{11}=0}^1 \dots \sum_{c_{IJ}=0}^1 \sum_{q_1=0}^{N-I,J} q_1 \cdot P(c_{11}, \dots, c_{IJ}; q_1).$$

2.3. Generic performance measures

For all three buffer schemes, the following further performance measures can be obtained by the help of the previously calculated measures:

- \bar{R} – The mean number of jobs at the servers: $\bar{R} = \sum_{i=1}^I \sum_{j=1}^J R_{ij}$
- \bar{O} – The mean number of jobs in the system: $\bar{O} = \bar{Q} + \bar{R}$
- \bar{N} – The mean number of jobs in the queue: $\bar{N} = N - \bar{Q} - \bar{R}$
- $\bar{\lambda}$ – The mean generating intensity: $\bar{\lambda} = \lambda \bar{N}$
- \bar{T} – The mean response time: $\bar{T} = \frac{\bar{O}}{\lambda}$
- \bar{W} – The mean waiting time: $\bar{W} = \frac{\bar{Q}}{\lambda}$

It is worth mentioning that practical implementation of the Separate queue scheme is the easiest because waiting jobs can be placed inside each physical server. For example jobs and parameters can be allocated in the local disk of each physical server.

The common queue scheme can be used in MaaS (Message Queuing as a Service) in the cloud computing paradigm. A possible example of the compute-intensive services of unknown service times is AWS cloud lambda service [4, 5].

2.4. Energy metrics

Let $P_{id,i}$ and $P_{ac,i}$ denote the active power consumption of server i when idle respectively busy. Furthermore, $R_{i,j}$ denotes the probability that server i in class j is busy and $L_{i,j}$ denotes the probability that it is idle. Then the average energy consumption of the whole system can be defined in the following way depending on whether idle servers are switched off or not:

- $AE_{no-switch}$ – The mean energy consumption of the system when idle servers are not switched off:

$$AE_{no-switch} = \sum_{i=1}^I \left(P_{ac,i} \sum_{j=1}^J R_{i,j} + P_{id,i} \sum_{j=1}^J L_{i,j} \right).$$

- $AE_{switch-off}$ – The mean energy consumption of the system when idle servers are switched off:

$$AE_{switch-off} = \sum_{i=1}^I \left(P_{ac,i} \sum_{j=1}^J R_{i,j} \right).$$

3. Numerical results

We have implemented the models introduced in Section 2 with the help of the SimPack package and now we present results on the comparison of scheduling algorithm. For this purpose, we have modeled three classes of Commercial Off-The-Shelf (COTS) servers with different types of processors (Intel Xeon E5-2670, Intel Xeon E5-2660, and Intel Xeon E5-4650L) whose characteristics are depicted in Table 1.

The simulations were performed with the parameters depicted in Table 2. Jobs are generated according to an exponential distribution with parameter λ from a source of N components and are routed to I classes of servers with J servers per class; the servers in class s process jobs according to an exponential distribution with parameter μ_s ; for this purpose, the performance C_s with maximum value 6419263 `ssj_ops` is adjusted to a service rate μ_s with maximum value 1 (i.e., every job is assumed to require 1 second on an Intel Xeon E5-2670 processor).

Table 1. Server classes.

Type of server	C_s (ssj_ops)	$P_{ac,s}$ (W)	$C_s/P_{ac,s}$	$P_{id,s}$ (W)
Acer AW2000h-Aw170h F2 (Intel Xeon E5-2670)[9]	6419263	1700	3776	364
Acer AW2000h-Aw170h F2 (Intel Xeon E5-2660)[8]	5286503	1275	4146	331
PowerEdge R820 (Intel Xeon E5-4650L)[10]	2790966	457	6102	108

Table 2. Simulation parameters.

Notation	Parameter	Value
N	Number of jobs in the source	150
I	Number of server classes	3
J	Number of servers per class	8
λ	Job generation rate	0.07–0.18
μ_s	Service rates of servers in class s	1; 0,82; 0,43

3.1. Performance measures

To evaluate the performance of the system, we analyze the mean service time, the mean waiting time, and the mean response time. Several figures are devoted to service, waiting, and response times. Though, response times can be more important, than service times, figures with service times are also presented. At most of the investigated cases, the waiting times and response times provide almost the same characteristics, thus due to the range constraint of the paper, only one of them is presented here. The service times have different characteristics, so beside the figures of response and/or waiting times, service times figures are also included.

Figure 4 shows the mean service time as a function of the generation rate λ using the HP policy for all buffering schemes. We see that as the generation rate increases, the mean service time also increases. This phenomenon can be explained by that jobs are first scheduled to the servers with highest performance. It also can be observed that for every buffering scheme the mean service times are almost the same, independently of the loads of the servers.

Furthermore, we can observe that, as the arrival rate starts to increase, slower servers start to play a more and more important role in the mean service time. Hereby the execution of the jobs become slower, thus jobs spend more time at the server. Of course, this is true only for a specific generation rate, because the more jobs arrive in the system, the higher the system load is. As we can see,

if the generation rate is greater then 0.16, we reach the maximum system load. This means that eventually every server becomes busy and the mean service time becomes constant.

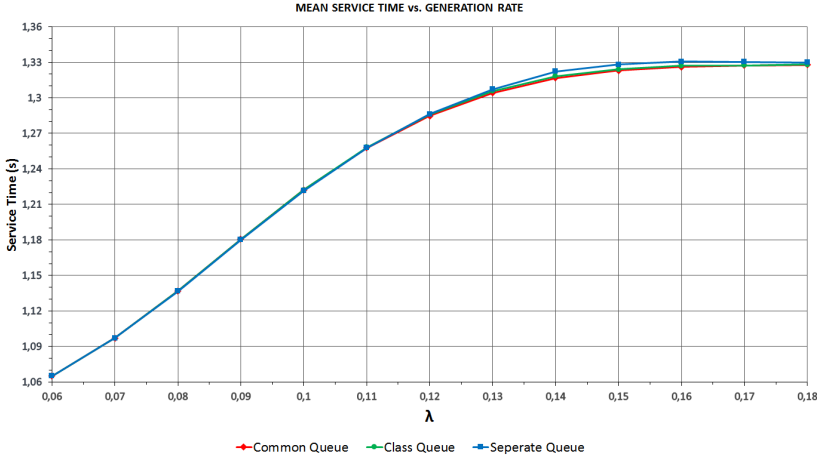


Figure 4. The mean service time applying the HP policy.

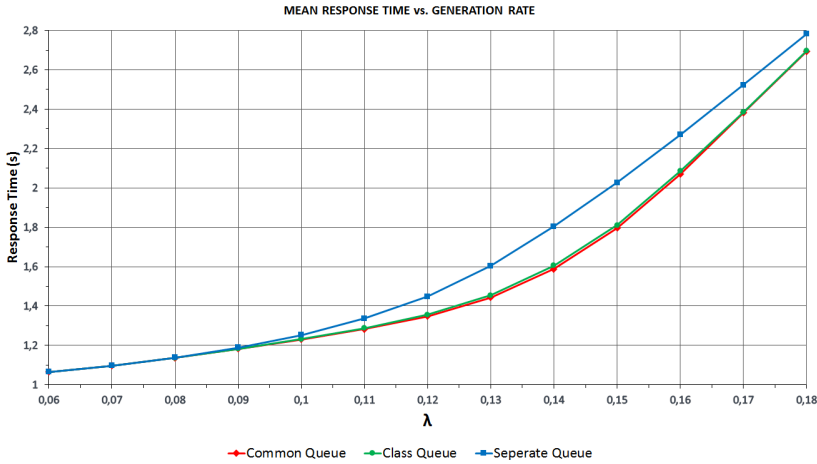


Figure 5. The mean response time applying the HP policy.

Figure 5 shows the mean response time and as a function of the generation rate using the HP policy (the same result can be obtained for the mean waiting time). On close inspection, the Common Queue scheme performs best but the difference to the Class Queue scheme is very small. As long as the generation rate does not reach 0.1, we cannot observe major differences between both schemes. But in the

range from 0.1 to 0.18 the difference appears vigorously, especially between the Separate Queue scheme and the other ones. In case of the Separate Queue scheme, the values of the mean waiting time and response time are the highest among the schemes. The reason why the Common Queue scheme ensures the lowest values for both the mean waiting and response times is that this model is able to utilize the available resources in the most efficient way. However, it has to be considered that the realization of Common Queue scheme is the most complicated one among the applied buffering schemes.

Figure 6 demonstrates the effect of the new policies MRT and MRTHP on the mean service time for the Separate Queue scheme. It is clearly visible that the MRT policy gives much smaller values as well as a relatively smaller system load. Comparing the HP and the MRTHP policy, we see that there is a slight difference discernible between them which starts to manifest when the system load gets high.

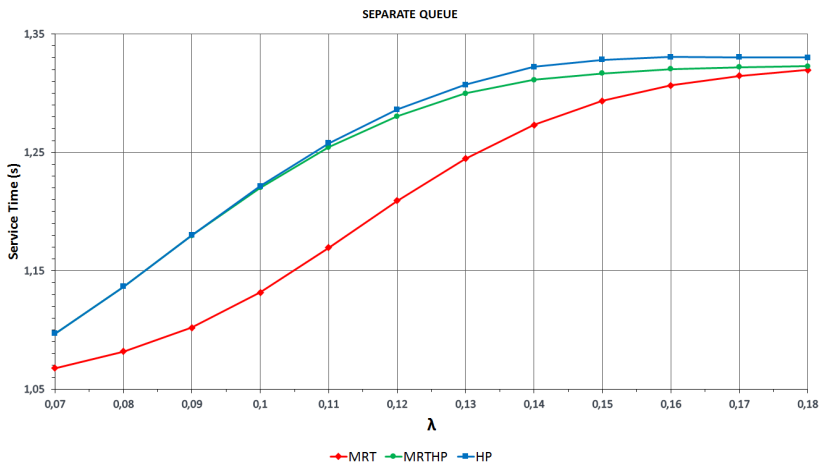


Figure 6. The effect of HP, MRT and MRTHP on the mean service time in case of Separate Queue.

Figure 7 shows the effect of the HP, MRT, and MRTHP policies on the mean waiting time using the Separate Queue scheme. A similar figure can be generated for mean response time. We can observe that the MRT policy is still the worst among the three policies. But the significant difference is that the MRTHP policy provides the most preferential values and not the HP policy. This is especially true when the system load is in the medium range.

In Figure 8 we can see the effect of the scheduling policies on the mean service time for the Class Queue scheme. It is clearly visible that the MRT policy gives much smaller values than the HP and the MRTHP policies, but the difference is here smaller than for the Separate Queue scheme (compare to Figure 6). We can observe that there is no difference between the HP and the MRTHP policies.

Figure 9 shows that using a higher generation rate the mean waiting time (and

similarly, the mean response time) become higher. We can see that the MRT policy gives the highest values and that there is only a very small difference between the HP and the MRTHP policies. It is worth noting that using MRT/MRTHP policy for the Common Queue scheme we get back the results of the HP policy.

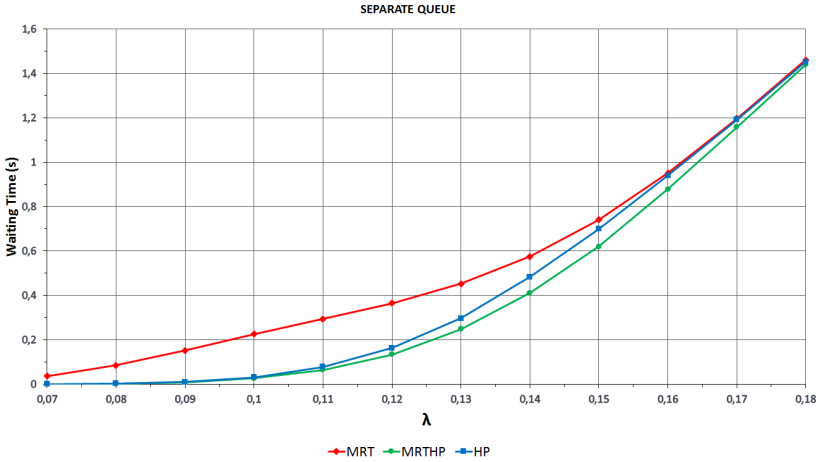


Figure 7. The effect of HP, MRT and MRTHP on the mean waiting time in case of Separate Queue.

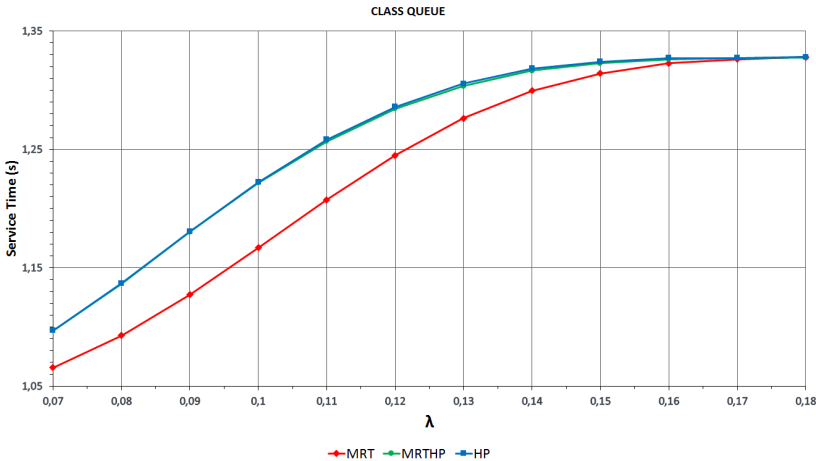


Figure 8. The effect of HP, MRT and MRTHP on the mean service time in case of Class Queue.

So all in all, we can observe that the significance of MRTHP is higher for the Separate Queue scheme than for the Class Queue scheme and is negligible

for the Common Queue scheme. Furthermore, while the MRTHP policy brings the Separate Queue scheme and the Class Queue scheme closer to the Common Queue scheme, the Common Queue scheme still seems to be the best to choose. However, the practical implementation of the Separate Queue scheme is the easiest and cheapest among all schemes; since the application of the MRT policy also makes the Separate Queue scheme competitive with the Common Queue Scheme, the combination of MRT policy and Separate Queue scheme may be preferred.

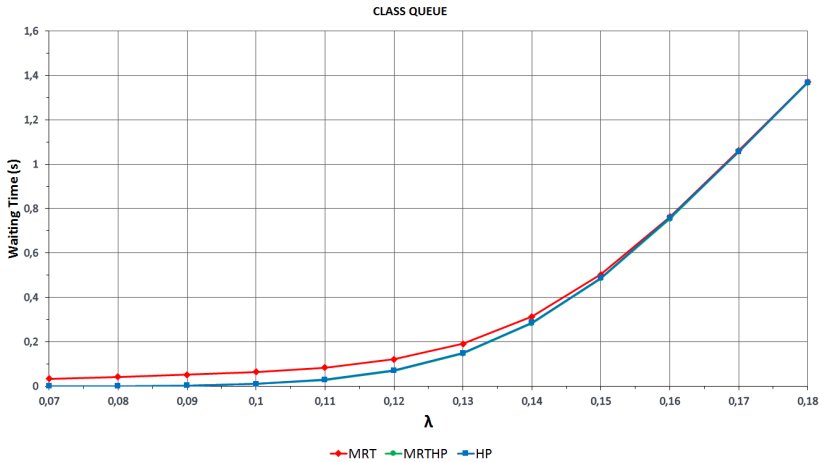


Figure 9. The effect of HP, MRT and MRTHP on the mean waiting time in case of Class Queue.

3.2. Energy consumption

Figures 10 and 11 demonstrate for the Separate Queue scheme the mean energy consumption in cases when idle servers are not switched off ($AE_{\text{no-switch}}$) respectively are switched off ($AE_{\text{switch-off}}$). As we can see, we get the highest energy consumption with the MRT policy and the lowest one with the HP policy; between the HP policy and the MRT policy there is only a small difference in case of $AE_{\text{switch-off}}$ and higher generation rates. As it can be expected, the difference between all policies disappears for high generation rates, because all servers become permanently busy, such that the energy consumption converges to around 1520 W.s/job.

Figures 12 and 13 demonstrate the mean energy consumption for the Class Queue scheme. In both cases we get the highest energy consumption using the MRT policy, and there is not any noticeable difference between the HP policy and the MRTHP policy. Again for high generation rates, the energy consumption converges for all policies to around 1520 W.s/job.

Finally, Figure 14 demonstrates how much energy can be saved by switching off the idle servers (for the Separate Queue scheme and the HP policy). Since

the saving decreases for higher generation costs and switching servers off and on involves extra costs, the choice to switch off servers must be clearly taken with care.

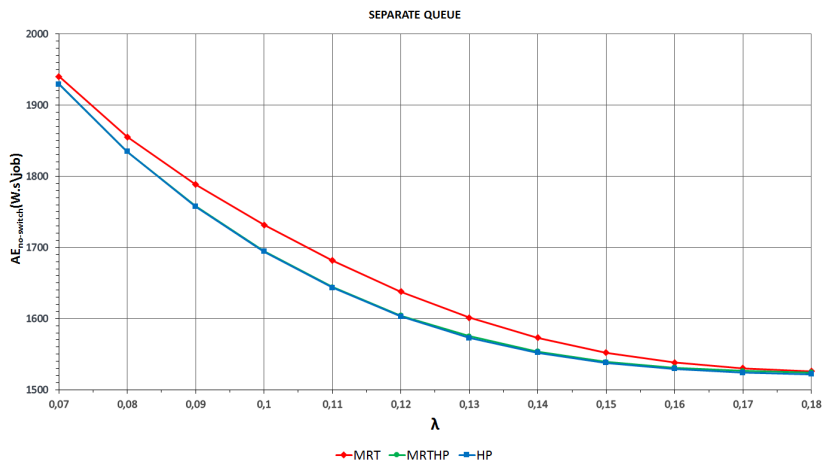


Figure 10. $AE_{no-switch}$ vs. generation rate using Separate Queue.

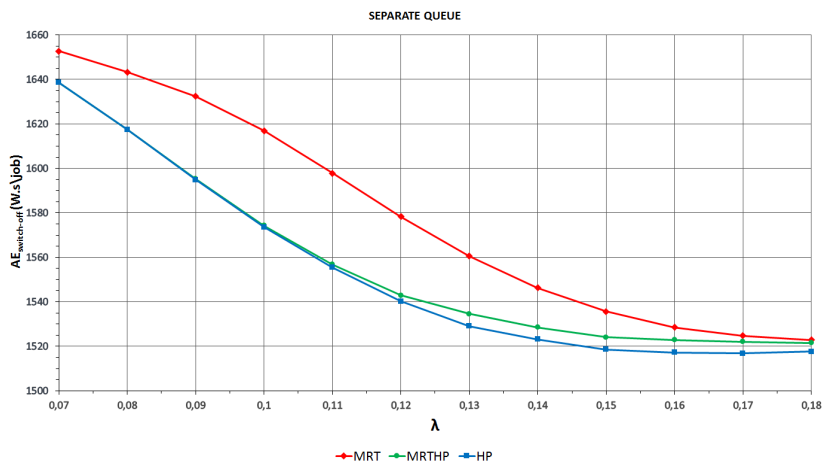


Figure 11. $AE_{switch-off}$ vs. generation rate using Separate Queue.

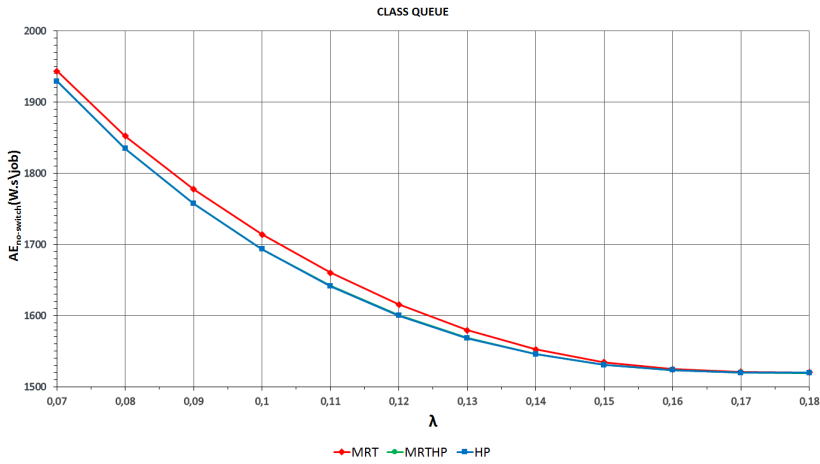


Figure 12. $AE_{no-switch}$ vs. generation rate using Class Queue.

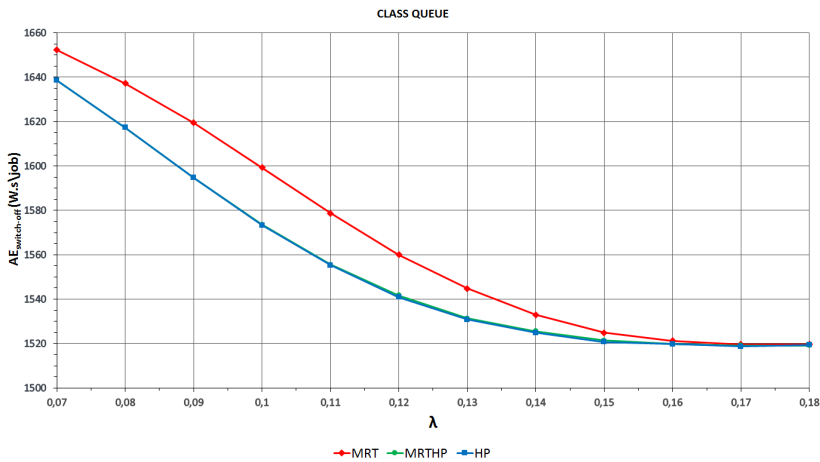


Figure 13. $AE_{switch-off}$ vs. generation rate using Class Queue.

4. Conclusions

So far the High Performance priority policy was considered in similar investigations. In this paper we have introduced and considered two new scheduling policies, namely the Mean Response Time priority and the Mean Response Time with High Performance priority policies. We investigate these policies with respect to three schemes of buffering the arriving jobs: Separate Queue, Class Queue, and Common

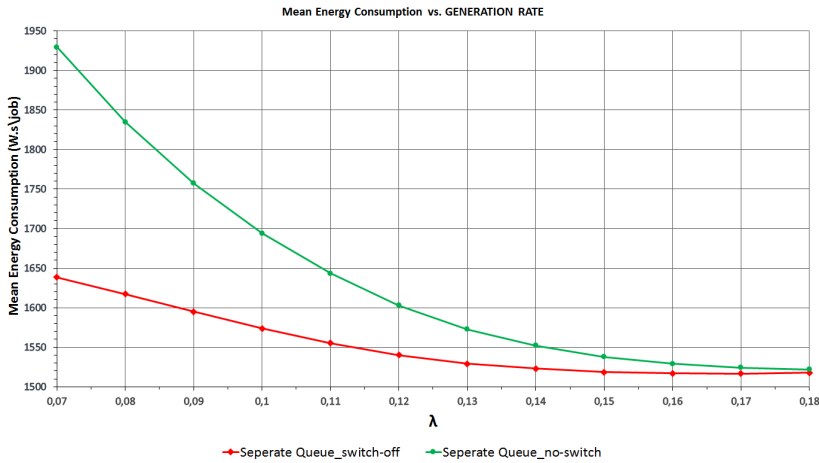


Figure 14. Mean energy consumption in case of switch off and switch on using HP policy.

Queue. Furthermore, we study the effect of scheduling policies and buffering policies on the energy consumption of a system that switches off idle servers with and without an energy saving mode. Since the state space of the describing Markov chain is very large, for that reason we used SimPack, a collection of C/C++ libraries and executable programs for computer simulation in order to obtain the performance measures.

The results described in this paper show that in the case of Separate and Class Queue the Mean Response time with High Performance priority policy improves the performance measures preeminently, in particular the mean sojourn time and the mean waiting time, compared to High Performance priority policy. As the Common Queue scheme operates with only one queue, we gain the same results with the application of the proposed new algorithms in the case of High Performance priority policy. Utilizing the MRHP priority policy the difference between the performance metrics of Separate and Common Queue decreases significantly; the numerical results show that the buffering schemes do not affect significantly the energy consumption of the investigated clusters. Accordingly selecting a good scheduling policy can augment the overall cluster performance without increased power consumption whenever the buffering scheme possesses more than one queue altogether.

References

- [1] M. CANKAR, M. ARTAČ, M. ŠTERK, U. LOTRIČ, B. SLIVNIK: *Co-Allocation with Collective Requests in Grid Systems*, Journal of Universal Computer Science 19.3 (2013), pp. 282–300, DOI: <https://doi.org/10.3217/jucs-019-03-0282>.

- [2] T. V. DO, B. T. VU, X. T. TRAN, A. P. NGUYEN: *A generalized model for investigating scheduling schemes in computational clusters*, Simulation Modelling Practice and Theory 37.0 (2013), pp. 30–42.
- [3] P. A. FISHWICK: *Simpack: Getting Started With Simulation Programming In C And C++*, in: WSC '92 Proceedings of the 24th Conference on Winter Simulation, ed. by J. S. ET AL., ACM, New York, 1992, pp. 154–162.
- [4] K. SALAH: *A queuing model to achieve proper elasticity for cloud cluster jobs*, International Journal of Cloud Computing 1.1 (2013), pp. 755–761.
- [5] K. SALAH, K. ELBADAWI, R. BOUTABA: *Performance modeling and analysis of network firewalls*, IEEE Transactions on network and service management 9.1 (2012), pp. 12–21.
- [6] A. TCHERNYKH, J. RAMÍREZ, A. AVETISYAN, N. KUZJURIN, D. GRUSHIN, S. ZHUK: *Two level job-scheduling strategies for a computational grid*, in: Proceedings of the 6th International Conference on Parallel Processing and Applied Mathematics (PPAMA05), 2006, pp. 774–781.
- [7] G. TERZOPOULOS, H. D. KARATZA: *Performance evaluation of a real-time grid system using power-saving capable processors*, The Journal of Supercomputing 61.3 (2012), pp. 1135–1153.
- [8] THE STANDARD PERFORMANCE EVALUATION CORPORATION: *Acer Incorporated Acer AW2000 h-AW170h F2 (Intel Xeon E5-2660)*,
URL: https://www.spec.org/power_ssj2008/results/res2012q4/power_ssj2008-20120918-00546.html.
- [9] THE STANDARD PERFORMANCE EVALUATION CORPORATION: *Acer Incorporated Acer AW2000 h-Aw170h F2 (Intel Xeon E5-2670)*,
URL: https://www.spec.org/power_ssj2008/results/res2013q1/power_ssj2008-20121212-00590.html.
- [10] THE STANDARD PERFORMANCE EVALUATION CORPORATION: *Dell Inc. PowerEdge R820 (Intel Xeon E5-4650L)*,
URL: https://www.spec.org/power_ssj2008/results/res2012q4/power_ssj2008-20121113-00586.html.
- [11] THE STANDARD PERFORMANCE EVALUATION CORPORATION: *SPECpower_ssj2008 Result File Fields*,
URL: https://www.spec.org/power/docs/SPECpower_ssj2008-Result_File_Fields.html.
- [12] Á. TÓTH, T. BÉRCZES, A.KUKI, B. ALMÁSI, W. SCHREINER, J. WANG, F. WANG: *Analysis of finite-source cluster networks*, CREATIVE MATHEMATICS AND INFORMATICS 2 (2016), pp. 223–235.
- [13] X. T. TRAN, T. V. DO, C. ROTTER, D. HWANG: *A New Data Layout Scheme for Energy-Efficient MapReduce Processing Tasks*, Journal of Grid Computing 16.2 (2018), pp. 285–298.
- [14] X. T. TRAN, T. V. DO, B. T. VU: *New Algorithms for Balancing Energy Consumption and Performance in Computational Clusters*, Computing and Informatics 36.2 (2017), pp. 307–330.
- [15] S. ZIKOS, H. D. KARATZA: *A clairvoyant site allocation policy based on service demands of jobs in a computational grid*, Simulation Modelling Practice and Theory 19.6 (2011), pp. 1465–1478.
- [16] S. ZIKOS, H. D. KARATZA: *Communication cost effective scheduling policies of nonclairvoyant jobs with load balancing in a grid*, Journal of Systems and Software 82.12 (2009), pp. 2103–2116.
- [17] S. ZIKOS, H. D. KARATZA: *The impact of service demand variability on resource allocation strategies in a grid system*, ACM Transactions on Modeling and Computer Simulation 20 (2010), 19:1–19:29.

Investigation of the efficiency of an interconnected convolutional neural network by classifying medical images*

Oktavian Lantang, Gyorgy Terdik,
Andras Hajdu, Attila Tiba

Faculty of Informatics, University of Debrecen, Hungary
oktavian_lantang@unsrat.ac.id, terdik.gyorgy@inf.unideb.hu
hajdu.andras@inf.unideb.hu, tiba.attila@inf.unideb.hu

Submitted: December 22, 2020

Accepted: April 7, 2021

Published online: May 18, 2021

Abstract

Convolutional Neural Network (CNN) for medical image classification has produced satisfying work [11, 12, 15]. Several pretrained models such as VGG19 [17], InceptionV3 [18], and MobileNet [8] are architectures that can be relied on to design high accuracy classification models. This work investigates the performance of three pretrained models with two methods of training. The first method trains the model independently, meaning that each model is given an input and trained separately, then the best results are determined by majority voting. In the second method the three pretrained models are trained simultaneously as interconnected models.

The interconnected model adopts an ensemble architecture as is shown in [7]. By training multiple CNNs, this work gives optimum results compared to a single CNN. The difference is that the three subnetworks are trained simultaneously in an interconnected network and showing one expected result.

*This work was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund. Research was also supported by the ÚNKP-20-4-I New National Excellence Program of the Ministry for Innovation and Technology from the Source of the National Research, Development and Innovation Fund, and by LPDP Indonesia in the form of a doctoral scholarship (<https://www.lpdp.kemenkeu.go.id>).

In the training process the interconnected model determines each subnetwork's weight by itself. Furthermore, this model will apply the most suitable weight to the final decision. The interconnected model showed comparable performance after training on several datasets. The measurement includes comparing the Accuracy, Precision and Recall scores as is shown in confusion matrix [3, 14].

Keywords: Convolutional Neural Network, medical image classification, interconnected model

1. Introduction

For the last decade, the Convolutional Neural Network (CNN) has done an impressive image classification task. Some of the successfully developed models, that achieved good results in classification tasks, include VGG19 [17], InceptionV3 [18], and MobileNet [8]. Referring to its architecture, CNN stacks several convolutions down or sideways according to each architecture's characteristics and then combined with a multilayer perceptron at the end of the network.

Medical Imaging is a technique of visualizing body parts to conduct clinical analysis or get a medical response. Furthermore, it also builds a database of body anatomy and physiology, allowing experts to identify abnormalities [4, 6]. Briefly, the medical imaging is started from the sensor's stage which penetrates the human body, subsequently it is transformed into signals and read by the detector, continuously mathematically manipulated and eventually visualized into an image [9].

The medical image classification has been well implemented in the following tasks. In [11] by adopting the VGG19 architecture, they developed four Convolution blocks. The first block consists of two convolutions with 64 channels using the ReLU activation function followed by the Max pooling layer to reduce its dimensions. Two convolutions fill the second block with 128 channels using the ReLU activation function and the Max pooling layer. The third block is similar to the previous one, but the convolutions' channels are changed for 256 with the ReLU activation function and Max pooling layer at the end of the block. The final block also consists of two convolutions with 512 channels using the ReLU activation function and a Max pooling layer to decrease its dimensions. The architecture is extended to the Multilayer Perceptron (MLP), consisting of two fully connected layers with a ReLU activation function and one final layer with a Sigmoid activation function after passing through the Flatten layer. Architecture also uses Dropouts in order to resist Overfitting. The model was then tested on the PatchCamelyon dataset, which was published in the Kaggle competition. This work has successfully exhibited good performance by achieving 0.92 and 0.98 for the validation accuracy score and the Area Under Curve, respectively.

Similar results can be seen in [15]. This architecture utilizes 121 layers of CNN, known as DenseNet to train the input images which are the frontal views of chest X-ray photos. The result is the probability score for the presence of pneumonia

on the input images. Further, the F1 score of the model was compared to four pathologists' F1 score. The results reported that the F1 score of the model was better than the mean F1 score of the four pathologies. CheXNet's F1 score was 0.435, while the mean score for the four pathologies was 0.387. All the F1 scores were measured by 95% confidence interval. This study was also compared with the results of previous studies in predicting 14 levels of pneumonia. These results showed that CheXNet model exceeds previous studies' results by dominating the best accuracy scores of the fourteen levels of pneumonia. In [12] AlexNet was used to demonstrate that CNN is capable of classifying Blood Smear Digital Images for malaria detection. The architecture was composed of four blocks. The first block was filled with two convolutions and it ended with the Max pooling layer. On the other hand, the second convolution was supplied with two blocks and it ended with the Average pooling layer. The third block was filled with two convolution layers without having a pooling layer. The ReLU activation function was used for these three blocks of convolution. The last block was the MLP with three fully connected layers which had 256 neurons. This architecture ended with Soft-Max two classes according to the given classification problem. The reported results were as follows: 97.37% Accuracy, 96.99% Sensitivity, 97.75% Specificity, 97.73% Precision, and 97.36% F1 Score.

The interconnected model was depicted in [7], which trained three CNNs together. The three CNNs used AlexNet, VGGNet, and GoogLeNet. The three CNNs were trained simultaneously on the skin cancer dataset, and then the best results were determined by voting. At the end of the article, they compared the AUCs of the three CNNs trained separately, in pairs, and simultaneously. The final results showed that the three models' best average AUC score was achieved when they trained simultaneously.

2. Datasets, hardware and software

In this work, we trained the developed model on three datasets published by the Kaggle dataset. The three datasets were the result of digitizing medical images of the human body. The first dataset was the chest X-ray dataset¹ representing data on a small amount of 5216 photos. The second was the Malaria dataset², representing an intermediate amount of data, namely 27,560 pictures. The last was the PatchCam dataset³, which was a large dataset with a total of 220,025 images. The chest X-ray dataset [10] was a radiological image of human lungs categorized into two classes and not distributed proportionally, consisting of 3875 and 1341 for viral pneumonia/bacterial and normal ones. The entire picture had run through doctors' labeling process and followed by an expert's level of accuracy verification. Here are examples of a chest X-ray dataset in Figure 1.

¹<https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia>

²<https://www.kaggle.com/miracle9to9/files1>

³<https://www.kaggle.com/c/histopathologic-cancer-detection>

The malaria dataset was owned by the Open Knowledge Foundation⁴ and published by the Kaggle dataset. Data was the result of digitization from the Thin Blood Smear process. The image was taken using an Android smartphone application integrated with a microscope using standard lighting. The data was distributed proportionally, with a total of 27,560 images. Experts carry out the labeling process by producing two categories of images, namely parasitized and normal. Here are examples image from the dataset in Figure 2.

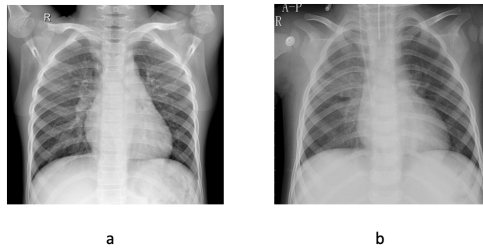


Figure 1. X-ray dataset, (a) Normal and (b) bacterial/viral pneumonia.

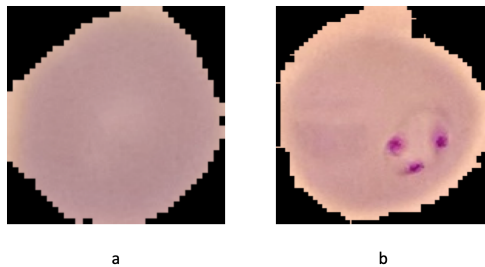


Figure 2. Malaria dataset, (a) Normal and (b) parasitized.

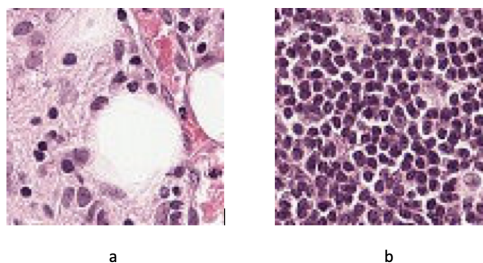


Figure 3. PatchCam Dataset, (a) Cancerous and (b) non-cancerous.

⁴<https://opendatacommons.org/licenses/by/1-0/index.html>

The next dataset was the PatchCam Dataset [1, 19], published at the Kaggle competition. The data were small pathology images converted into digital format, consisting of 220,025 images, and not evenly distributed in the two classes, cancerous and non-cancerous. Here are examples from the PatchCam dataset in Figure 3.

For the daily experiment, we used Google Collaboratory, and then the data were trained on a Dell Desktop with GEFORCE GTX 1060 6GB. Each code in this work was written in Python version 3.6 by exploiting jupyter notebook. Apart from that, the Tensorflow and Keras frameworks were also used in this work.

3. Methodology

3.1. Network architecture

In this work, we proposed an interconnected CNN model. This model was a combination of three subnetworks consisted of three pretrained models. The purpose of combining the three subnetworks is to let the three subnetworks work independently in the training process to determine the influence of each subnetwork on decision making. Thus, the interconnected model will get the proper weight, increasing its ability in the classification task. The three subnetworks, namely, VGG19 consisted of sixteen convolution layers, InceptionV3 consisted of forty-eight layers of convolution, and MobileNet consisted of eighteen layers of convolution. The Multilayer Perceptron (MLP) of each subnetwork was replaced with three Fully Connected layers using the ReLU activation function to fit the interconnected models needed. Before entering into MLP, the architectural design required a Flatten layer to convert the features' dimensions. The next layer was the Concatenation layer, where the three output layers will be combined so that the interconnected model will only have one output. Afterward, the three Fully Connected layers were installed, consisting of two Fully Connected layers using the ReLU activation function and one Fully Connected layer using Soft-Max two classes to represent our dataset's classification problem. Here it is shown the architectural design in Figure 4.

3.2. Training process

Due to variations between small and large datasets, the augmentation method [13] was employed to provide sufficient data for the model. The augmentation process that we have implemented includes rotation, shifting, shearing, zooming, and flipping. For simplicity purposes, a few aspects were standardized. We took 10% of the images from each dataset, then used them as a test set. Afterward, 70% of the remaining images were allocated as a training set and 30% as a validation set. The input sizes of all datasets were set to 100×100 pixels. The batch sizes were set to 16 and an epoch of 50 for each training process.

In our work, emphasis is put on having a training process which is carried out simultaneously in a series of interconnections. Although each subnetwork has

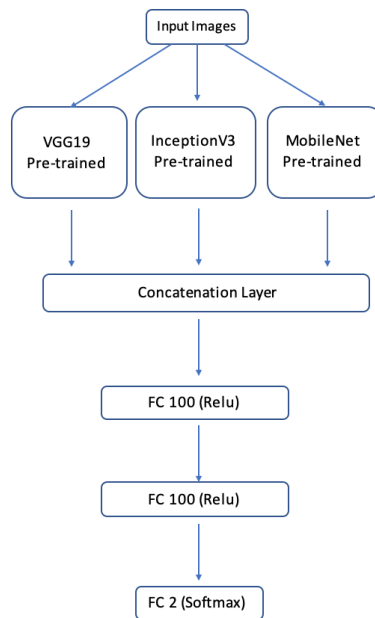


Figure 4. Architecture of the interconnected Model.

authority in the training process, the training process is an integral part that cannot be separated from one and another. This process causes each subnetwork's weights to be determined by the training process itself and not by the user. When a subnetwork has better performance than others, the subnetwork will automatically have more significant impact on the overall interconnected model. Conversely, if a subnetwork produces unsatisfactory performance, it will have less weight in the final decision process. In [7], the initial weights was determined to be equivalent for the three subnetworks. In our work, the interconnected model determined its weights according to the training process that each model gone through. The weighting process of each subnetwork was intervened neither at the initial nor during the final decision stage.

Utilizing the transfer learning method in the training process caused many layers that may not be necessary and will consume extra resources of the computational of our work. The Freezing layers technique as explained in [2] was implemented to save computation resources without destroying the model's performance. The process includes freezing several layers causing the input images to go through these layers to avoid updating weights. This Freezing layers method aimed to diversify the three subnetworks, allowing three different perspectives to more effectively notice the image's characteristics.

During the training process, the model tried to get the smallest possible Loss score to get the best possible Accuracy score. Thus, we aimed to minimize the Loss score of the model using the Soft-Max Loss function. The Soft-Max Loss

function itself was the product of implementing the Soft-Max function into the Loss function. To see the connection between these two functions more clearly, let's look at the formula (3.1). As explained in [5, 16], the formula Soft-Max function $f(s): \mathbb{R}^K \rightarrow \mathbb{R}^K$ is a vector function in the range 0 to 1, where K is number of classes.

$$f(s)_i = \frac{e^{s_i}}{\sum_{c=1}^K e^{s_c}}. \quad (3.1)$$

This formula is obtained by calculating the e number to the power of s_i , s_i itself refers to the score s from class i . Hence, the numerator divided by the sum of the constant e to the power of all score in number of classes. So that when implemented into the Soft-Max loss function [5, 16] it will become:

$$CE = - \sum_i^K t_i \log(f(s)_i). \quad (3.2)$$

Equation (3.2) explains that cross-entropy CE is the sum of ground truth t_i logarithm the CNN score of each class that represents by $f(s)_i$.

We were also optimizing our model by setting Adam optimizer at 1e-4 learning rate and decay 1e-6 for each subsequent epoch. Furthermore, we calculated the accuracy score based on the Confusion Matrix [3, 14], which results in True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). TP is representing ill patients as precisely predicted to be ill patients. Meanwhile, TN is healthy patients correctly predicted as healthy patients. On the other hand, FP is healthy patients incorrectly predicted as ill patients. and vice versa FN is ill patients, mistakenly classified as healthy patients. We also measured the Precision and Recall score to see the performance of the ill predicted label. For more details, the Accuracy, Precision, and Recall score calculation are in the equations. (3.3), (3.4), and (3.5).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \quad (3.3)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad (3.4)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (3.5)$$

4. Results

4.1. Training and validation accuracy

In the first experiment, the three subnetworks were trained separately. Thus each subnetwork provided its prediction result. The results from each subnetwork were then used in the voting process. Experiments using the chest X-ray dataset showed that the three pretrained models can be appropriately implemented. It can be observed from the ability of the three pretrained models to validate the training

results. The two intersecting lines in Figure 5 indicated that the training accuracy and the validation accuracy scores were comparable. It revealed that the three pre-trained models were suitable and did not overfit. Each model achieved a validation score of 0.91 for VGG19, 0.89 for InceptionV3, and 0.93 for MobileNet.

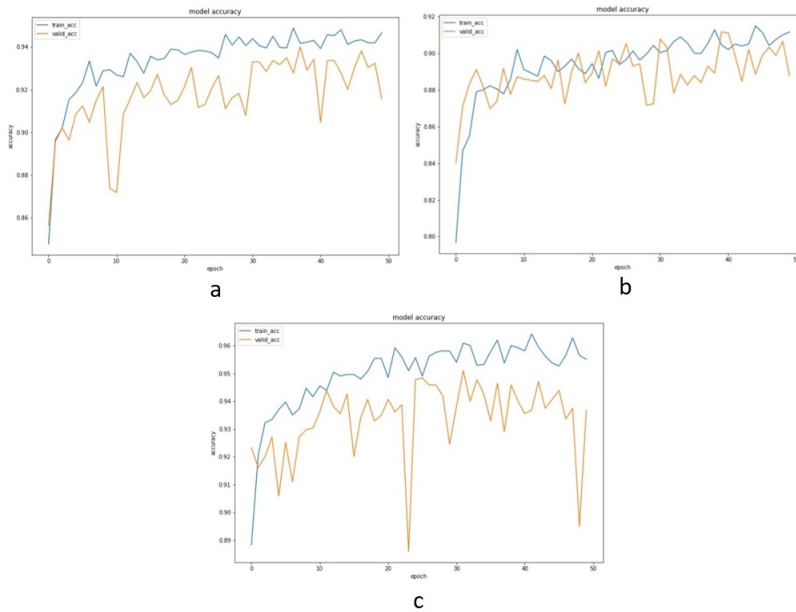


Figure 5. Training and validation accuracy for chest X-ray dataset: a) VGG19, b) InceptionV3 and, c) MobileNet.

Likewise, the training process using the malaria dataset showed a good performance of VGG19 by achieving a validation score of 0.90. InceptionV3 and MobileNet had satisfactory validation scores of 0.80 as shown in Figure 6.

For training the PatchCam dataset, VGG19 achieved optimum results in the classification task reported a score of 0.86 for validation accuracy. InceptionV3 showed satisfactory performance with a validation score of 0.70. The MobileNet pretrained model also achieved the same score with a small overfit condition. In Figure 7, the training and validation processes on the PathCam dataset were depicted.

The training process for the interconnected model operated the same as the separate training process. The only difference was the interconnected model trains three submodels simultaneously. This model had trained all parameters owned by the three pretrained models. These experiments reported that the interconnected model showed comparable performance with any single model. The interconnected model's validation accuracy scores compared to all single models are presented in Table 1.

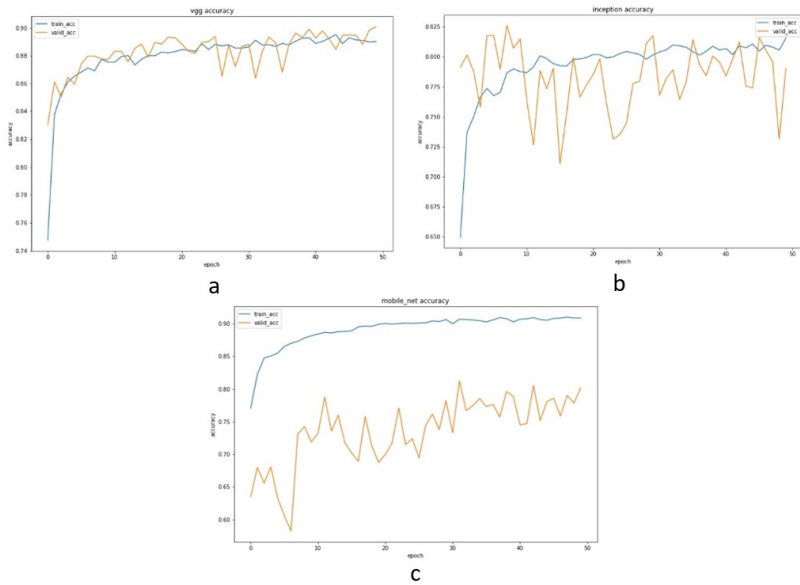


Figure 6. Training and validation accuracy for malaria dataset: a) VGG19, b) InceptionV3 and, c) MobileNet.

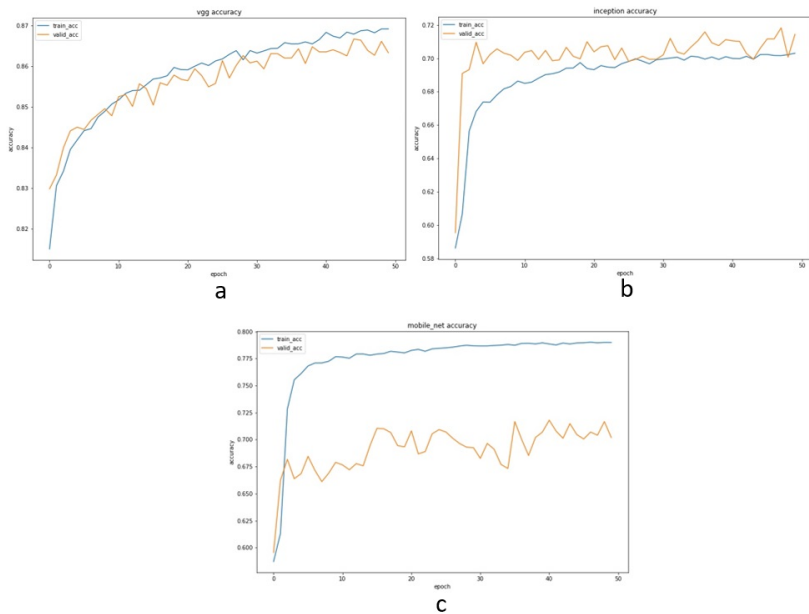


Figure 7. Training and validation accuracy for PatchCam dataset: a) VGG-19, b) InceptionV3 and, c) MobileNet.

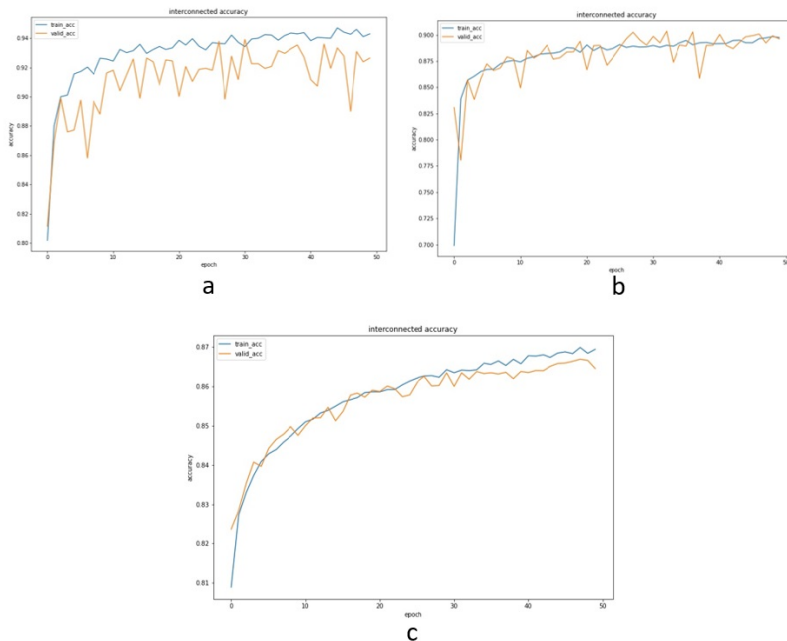


Figure 8. Training and validation accuracy of the interconnected model on all datasets: a) chest X-ray, b) Malaria, c) PatchCam.

Table 1. Validation accuracy of all models for three datasets.

Dataset	VGG19	InceptionV3	MobileNet	Interconnected
chest X-ray	0.91	0.89	0.93	0.93
Malaria	0.90	0.80	0.80	0.90
PatchCam	0.86	0.70	0.70	0.86

4.2. Visualization of training process

Figure 9 explains the steps that occur during the training process by visualizing [2] the images. Figure *a* represents the image that was input to the model. The original image size, as previously mentioned, was 100x100 pixels. Given that this dataset's training process studied the completeness of photographs of human lungs, an example image from the normal category dataset is presented. Having the extracted features as shown in parts *b* to *d*, then the part *e* shows that the model can detect lungs in a healthy condition with an image representation showing the lungs appearing intact.

Figure 10 depicts the malaria dataset image from the input process, feature extraction, and object detection in human blood. Part *a* is the image input to the model. Parts *b* to *d* are the extracted features by the models. Part *e* is the model's

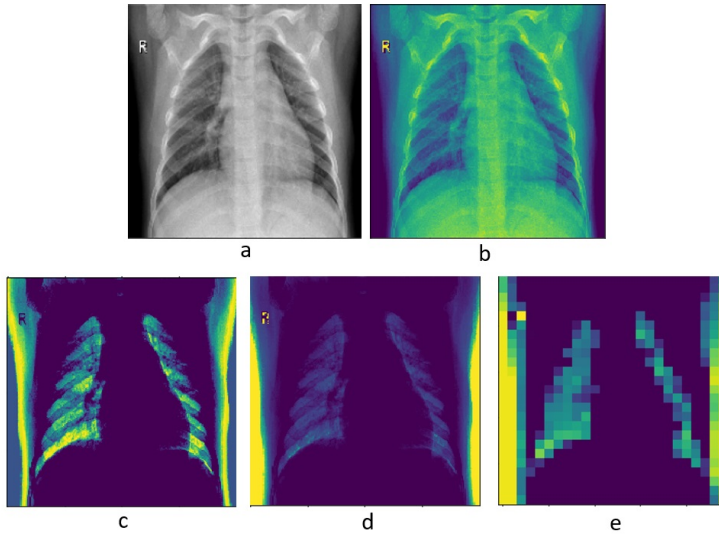


Figure 9. a) Input, b)-d) Extracted Features, e) Heatmap.

heatmap as the classification result for the image labeled as malaria.

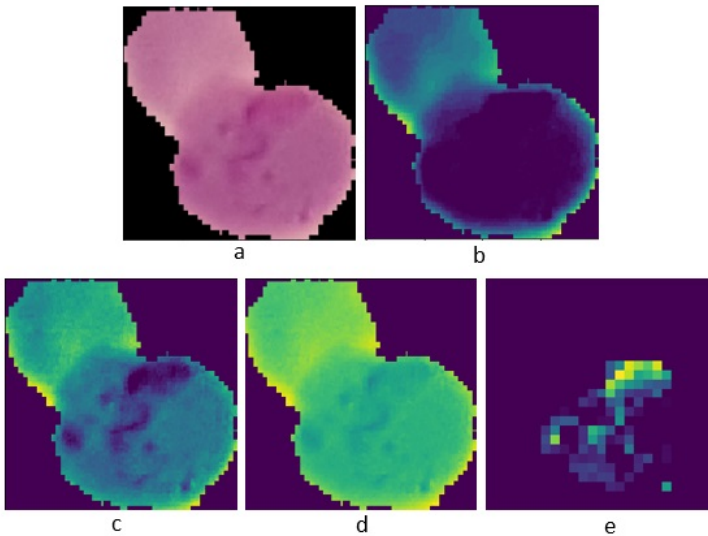


Figure 10. a) Input, b)-d) Extracted features, e) Heatmap.

The training process on the PatchCam dataset aimed to detect the presence of cancer cells in the image. Therefore, when the image in Figure 11 a was input to the model, the model performed the feature extraction process shown in Figure 11

b to *d*. After that, the model can detect the cancer cells' presence in the image as displayed in the heatmap in Figure 11 *e*.

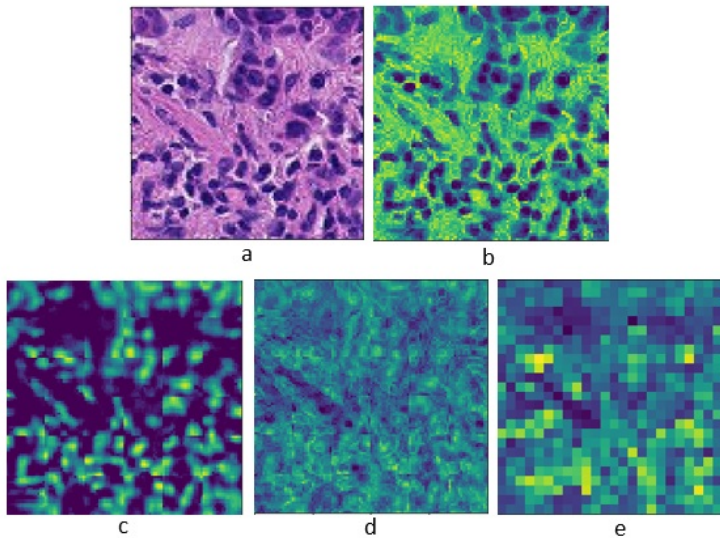


Figure 11. a) Input, b)-d) Extracted Features, e) Heatmap.

4.3. Predicted results

After training and validating all models on the dataset, The model was tested using the test set. This was performed to observe the model's ability to predict new data. From the three pretrained models that we trained on the chest X-ray dataset, it can be reported that all models can predict the data accurately. Table 2 shows that all pretrained models achieved good accuracy scores, which were 0.91, 0.84, and 0.91 for VGG19, InceptionV3, and MobileNet, respectively. However, in this case, the interconnected model's achievement has not exceeded majority voting performance, which can be seen from the majority voting accuracy score of 0.91. Nevertheless, the interconnected model results were comparable with both the single model and the majority voting. Table 2 also shows that the correctly and incorrectly predicted images is well distributed. The precision and recall score in Table 2 also indicates that the interconnected model's ability was slightly better than the majority voting model. In retrieving images containing pneumonia, the interconnected model found 383 images with 7 images error or equivalent to a recall score of 0.98, compared to majority voting, with 370 images containing pneumonia and 20 images error or equivalent to 0.85 recall score. Even so, the two models' precision score was comparable, namely, 0.91 for majority voting and 0.87 for the interconnected model.

A similar approach was applied to the malaria dataset to see the three models'

Table 2. Confusion matrix and classification report of chest X-ray dataset.

Model	TP	FN	TN	FP	Accuracy	Precision	Recall
VGG19	361	29	206	28	0.91	0.88	0.88
InceptionV3	345	45	179	55	0.84	0.80	0.76
MobileNet	370	20	195	39	0.91	0.90	0.95
Majority Voting	370	20	198	36	0.91	0.91	0.85
Interconnected	383	7	177	57	0.90	0.87	0.98

ability to predict new data. Different results were obtained from this experiment, as shown in Table 3. This experiment gained an accuracy score for each pretrained model, consisting of 0.87 for VGG19, 0.87 for InceptionV3, and 0.72 for MobileNet. The abilities MobileNet was not optimum because there was a significant error in predicting data in the positive class. In this experiment, the majority voting method cannot provide maximum results, namely 0.86 accuracy. On the other hand, the interconnected model can provide a proper weight, increasing the accuracy score to 0.88. Furthermore, the interconnected model showed satisfying performance on the positive class with a recall score of 0.82 compared to majority voting with a recall score of 0.74. This value represented the number of images identified with malaria that can be retrieved as many as 823 images with 177 errors for the interconnected model. In comparison, majority voting found 737 images with an error of 263 images. The Precision scores in Table 3 represent the level of precision of each model. It is shown that all models have a good level of precision, which is above 0.93.

Table 3. Confusion matrix and classification report of Malaria dataset.

Model	TP	FN	TN	FP	Accuracy	Precision	Recall
VGG19	798	202	938	62	0.87	0.93	0.80
InceptionV3	776	224	963	37	0.87	0.95	0.78
MobileNet	443	557	997	3	0.72	0.99	0.44
Majority Voting	737	263	988	12	0.86	0.98	0.74
Interconnected	823	177	942	58	0.88	0.93	0.82

The last experiment utilized the PatchCam dataset. Comparing the three sub-network, the VGG19 model showed the best performance with an accuracy score of 0.87, followed by the InceptionV3 model with an accuracy score of 0.70, and the MobileNet with an accuracy score of 0.66. This experiment also produced one dominant model. Thus, the majority voting method did not work optimally and gained an accuracy score of 0.83. On the other hand, the interconnected model can provide a suitable weight for each model to get an accuracy score of 0.87. As seen in Table 4, the interconnected model has better Precision and Recall scores

than the majority voting. Thus the interconnected model has better performance in both classification classes, cancer and non-cancer images.

Table 4. Confusion matrix and classification report of PatchCam dataset.

Model	TP	FN	TN	FP	Accuracy	Precision	Recall
VGG19	6692	2199	12400	709	0.87	0.89	0.75
InceptionV3	6638	2253	8838	4271	0.70	0.61	0.75
MobileNet	2696	6195	11882	1227	0.66	0.69	0.30
Majority Voting	5902	2989	12342	767	0.83	0.88	0.66
Interconnected	6721	2170	12316	793	0.87	0.89	0.77

5. Conclusions

After training all pretrained models separately and simultaneously on the three datasets, we concluded the interconnected model could be used when majority voting did not work optimally. The results can be seen in the second and third experiments using the Malaria and PatchCam dataset. Although the interconnected model's accuracy score slightly corrected the score of majority voting, in predicting positive classes, the interconnected model worked better than other models on the three datasets. The interconnected model worked by giving the submodels the best weights without training them separately. This method was more efficient than first training of the three submodels to determine their abilities and then consider their appropriate weights.

Our work focused on the investigating of the interconnected model's ability versus majority voting, but not analyzing the individual optimization of each sub-network's architecture. It means that we used the transfer learning method without adjustment. This can be confirmed by comparing our work's accuracy with several references which use the same dataset [11, 12, 15]. In this case, InceptionV3 and MobileNet's pretrained models required adjustments in the depth and number of layers. Therefore, in our future work we will examine each network's further optimization possibilities to help the developed system to be more accurate. In addition, the usage of the imbalanced dataset also influences the model's performance. We consider using a method that can handle the imbalanced dataset's problem to increase the model's accuracy.

References

- [1] B. E. BEJNORDI, M. VETA, P. J. V. DIEST, B. V. GINNEKEN, N. KARSSEMEIJER, G. LITJENS, J. A. V. D. LAAK, C. CONSORTIUM: *Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer*, JAMA 318.22 (2017), pp. 2199–2210.

- [2] F. CHOLLET: *Deep Learning With Python*, in: New York, USA: Manning Publication Co, 2018, p. 160.
- [3] T. FAWCETT: *An Introduction to ROC Analysis*, Pattern Recognition Letters 27.8 (2006), pp. 861–874,
DOI: <https://doi.org/10.1016/j.patrec.2005.10.010>.
- [4] D. GANGULY, S. CHAKRABORTY, M. BALITANAS, T.-H. KIM: *Medical Imaging: A Review*, in: International Conference on Security-Enriched Urban Computing and Smart Grid, Heidelberg, Berlin: Springer, 2010, pp. 504–516,
DOI: http://dx.doi.org/10.1007/978-3-642-16444-6_63.
- [5] I. GOODFELLOW, Y. BENGIO, A. COURVILLE, in: *Deep Learning*, Cambridge MIT press, 2016, p. 181.
- [6] S. GOSWAMI, U. DEY, P. ROY, A. ASHOUR, N. DEY: *Medical Video Processing: Concept and Applications*, in: Feature Detectors and Motion Detection in Video Processing, IGI Global, 2017, pp. 1–17,
DOI: <http://dx.doi.org/10.4018/978-1-5225-1025-3.ch001>.
- [7] B. HARANGI, A. BARAN, A. HAJDU: *Classification of skin lesions using an ensemble of deep neural networks*, in: 2018 40th annual international conference of the IEEE engineering in medicine and biology society(EMBC), Honolulu, HI, USA: IEEE, 2018, pp. 2575–2578,
DOI: <https://doi.org/10.1109/EMBC.2018.8512800>.
- [8] A. G. HOWARD, M. ZHU, B. CHEN, D. KALENICHENKO, W. WANG, T. WEYAND, M. ANDREETTO, H. ADAM: *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*, arXiv:1704.04861 (2017).
- [9] H. KASBAN, M. A. M. EL-BENDARY, D. H. SALAMA: *A comparative study of medical imaging techniques*, International Journal of Information Science and Intelligent System 4.2 (2015), pp. 37–58.
- [10] D. KERMAN, K. ZHANG, M. GOLDBAUM: *Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification*, Mendeley Data v2.2, <https://data.mendeley.com/datasets/rscbjbr9sj/2>, 2018,
DOI: <https://doi.org/10.17632/rscbjbr9sj.2>.
- [11] O. LANTANG, A. TIBA, A. HAJDU, G. TERDIK: *Convolutional Neural Network for Predicting The Spread of Cancer*, in: Proceedings of the 2019 10th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), Naples, Italy: IEEE, 2019, pp. 175–180,
DOI: <https://doi.org/10.1109/CogInfoCom47531.2019.9089939>.
- [12] Z. LIANG, A. POWELL, I. ERSOY, M. POOSTCHI, K. SILAMUT, K. PALANIAPPAN, P. GUO, M. A. HOSSAIN, A. SAMMER, R. J. MAUDE, J. X. HUANG, S. JAEGER, G. THOMA: *CNN-Based Image Analysis for Malaria Diagnosis*, in: 2016 IEEE Conference on Bioinformatics and Biomedicine (BBIM), Shenzhen, China: IEEE, 2016, pp. 493–496,
DOI: <https://doi.org/10.1109/BIBM.2016.7822567>.
- [13] A. MIKOŁAJCZYK, M. GROCHOWSKI: *Data augmentation for improving deep learning in image classification problem*, in: 2018 International Interdisciplinary PhD Workshop(IIPhDW), Swinoujście, Poland: IEEE, 2018,
DOI: <https://doi.org/10.1109/IIPHDW.2018.8388338>.
- [14] D. M. W. POWERS: *Evaluation: From Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation*, Journal of Machine Learning Technologies 2.1 (2010), pp. 37–63,
DOI: <https://doi.org/10.9735/2229-3981>.
- [15] P. RAJPURKAR, J. IRVIN, K. ZHU, B. YANG, H. MEHTA, T. DUAN, D. DING, A. BAGUL, R. L. BALL, C. LANGLOTZ, K. SHPANSKAYA, M. P. LUNGREN, A. Y. NG: *CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning*, arXiv preprint arXiv:1711.05225 (2017).

-
- [16] P. SADOWSKI: *Notes on back propagation*, <https://www.ics.uci.edu/pjsadows/notes.pdf> (online) (2016).
 - [17] K. SIMONYAN, A. ZISSERMAN: *Very Deep Convolutional Networks for Large-Scale Image Recognition*, arXiv preprint arXiv:1409.1556 (2014).
 - [18] C. SZEGEDY, W. LIU, Y. JIA, P. SERMANET, S. REED, D. ANGUELOV, D. ERHAN, V. VANHOUCHE, A. RABINOVICH: *Going Deeper with Convolutions*, in: Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA: IEEE, 2015, pp. 1–9, DOI: <https://doi.org/10.1109/cvpr.2015.7298594>.
 - [19] B. S. VEELING, J. LINMANS, J. WINKENS, T. COHEN, M. WELLING: *Rotation Equivariant CNNs for Digital Pathology*, in: International Conference on Medical image computing and computer-assisted intervention, Springer, Cham, 2018, pp. 210–218.

Motor imagery EEG classification using feedforward neural network*

Tamás Majoros^a, Stefan Oniga^{ab}, Yu Xie^a

^aIntelligent Embedded Systems Research Laboratory, Faculty of Informatics,
University of Debrecen
majoros.tamas@inf.unideb.hu
yu.xie@inf.unideb.hu
oniga.istvan@inf.unideb.hu

^bTechnical University of Cluj-Napoca, North University Centre of Baia Mare

Submitted: December 21, 2020

Accepted: April 19, 2021

Published online: May 18, 2021

Abstract

Electroencephalography (EEG) is a complex voltage signal of the brain and its correct interpretation requires years of training. Modern machine-learning methods help us to extract information from EEG recordings and therefore several brain-computer interface (BCI) systems use them in clinical applications.

By processing the publicly available PhysioNet EEG dataset, we extracted information that could be used for training feedforward neural network to classify three types of activities performed by 109 volunteers. While volunteers were performing different activities, a BCI2000 system was recording their EEG signals from 64 electrodes. We used motor imagery runs where a target appeared on either the top or the bottom of a screen. The subject was instructed to imagine opening and closing either both his/her fists (if the target is on top) or both his/her feet (if the target is on the bottom) until the target disappears from the screen.

We used the EEGLAB Matlab toolbox for EEG signal processing and applied several feature extraction techniques. Then we evaluated the classification performance of feedforward, multilayer perceptron (MLP) networks

*This work was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund.

with different structures (number of layers, number of neurons). Achieved accuracy score for test data was 71.5%.

Keywords: Neural network, multilayer perceptron, classification, EEG, BCI

1. Introduction

During brain activity ionic current flow generates voltage fluctuations that can be measured on the scalp with electrodes. Measured signals (electroencephalogram, EEG) are quite complex and their correct interpretation demands expertise. However, due to the great advances in machine learning science, machine learning techniques are increasingly used for interpreting EEG signals [3, 10].

A brain–computer interface (BCI) is a communication pathway between a brain and a computer. A possible application area is to perform certain physical effects with just brain waves, not using muscles. This can be helpful for people who are immobile, elderly or paralyzed, therefore these systems are widely used for clinical applications in both rehabilitation [2] and communication [8].

Motor imagery is the imagination of the movement of various body parts. It causes cortex activation and oscillations in the EEG, therefore it can be detected by the BCI device to infer a user’s intent. Accurate and sufficiently fast recognition of the activity to be performed is essential in developing such BCI applications.

Artificial neural networks (ANN) are integral parts of BCI systems in many cases. The goal of our work is to create a neural network that can be applied in BCI applications to recognize motor imagery activities. However, in order for these networks to perform well, they need many training examples.

Creating a database which is suitable for training neural networks requires a lot of volunteers, an advanced data collection system, and several months of work. Fortunately, there are quite a few EEG databases that are publicly available. These were made for different purposes, for example for recognition of motor imagery, epileptic seizure, various brain lesions. To avoid the difficulties of creating our own database, we used such a publicly available database.

In recent years, a number of consumer-grade BCI devices have been developed that are allowed to use outside of clinical settings [9]. In the future we would like to use a neural network trained on a large, public database to recognize EEG signals recorded by our own BCI device.

2. PhysioNet EEG dataset

Researchers created numerous EEG datasets for various purposes and made them publicly available. Some of them are made to evaluate and study various health-related problems, such as epilepsy, autism or sleep disorders. Another common area of use is related to motor and motor imagery activities. One such dataset is the PhysioNet EEG dataset [4], which contains one- and two-minute EEG recordings from 109 volunteers.

Researchers used the BCI2000 software suite to record brain waves while subjects performed various tasks. 14 measurements were performed on all 109 volunteers, resulting in 1526 one- and two-minute data files. Two of the 14 measurements are baseline runs with eyes open and closed, these are one minute long. The remaining 12 are two-minute runs: two motor and two motor imagery measurements, repeated three times. These activities are:

- A target appears on the left or right side of the screen. The volunteer closes the fist on that side.
- A target appears on the left or right side of the screen. The volunteer imagines closing the fist on that side.
- A target appears on the top or bottom side of the screen. The volunteer closes both fists or feet, respectively.
- A target appears on the top or bottom side of the screen. The volunteer imagines closing both fists or feet, respectively.

We used those motor imagery runs where a target appeared on the top or the bottom of a screen. The subjects imagined opening and closing either both fists (if the target is on top) or both feet (if the target is on the bottom). Subjects were given new tasks every four seconds.

The data are available in EDF+, which is a standard EEG data recording format [6]. It includes the use of standard electrode names and supports timestamped annotations. EEG data were recorded from 64 electrodes with a sampling frequency of 160 Hz. The 64 electrodes were placed on the scalp as per the international 10-10 system.

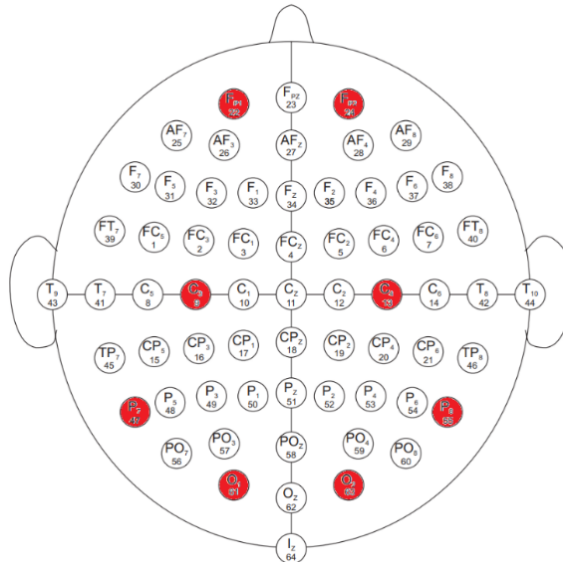


Figure 1. Placement of electrodes on the head.

Since we want to use the neural network trained on this dataset to process our own data in the future, it is important to use only those data that can be acquired by our own device too. Our device is an OpenBCI Ultracortex Mark IV headset with Cyton board. It has eight channels: C3, C4, Fp1, Fp2, P7, P8, O1 and O2, so only these channels were taken into account. The available electrodes in PhysioNet database are shown in Figure 1. The eight channels available on our device are marked in red.

3. Method

3.1. Data preprocessing

Electrophysiological signals, including EEG can be effectively processed in Matlab using the EEGLAB toolbox [1]. It provides a graphical user interface (GUI) allowing users to interactively process data. It supports various file formats, including EDF, and several data processing methods like independent component analysis (ICA) and time/frequency analysis (TFA).

The original scalp data is a matrix. Time course of the measured voltages on the channels are represented by the rows of the matrix. Since measurements are approximately two minutes long with 160 Hz sampling frequency, and eight channels are used, the size of the data matrix is 8×19920 .

An additional three rows were added to the matrix to show which activity was being performed by the volunteer at the time of sampling. In each column, one of the three values is one and the other two are zeros according to the current activity (fists, feet, relax). These additional three rows extend the size of the matrix to 11×19920 .

A frequently used data preprocessing step is the feature extraction from the raw data to make the applied machine learning model more effective. These features can be extracted from the wavelet, frequency, or time domain.

In the case of EEG signals, neural oscillations can be observed in the frequency domain. Time-series data can be transformed to the frequency domain using spectral methods. Figure 2 shows the distribution of the signal power over frequency for the first second of the first measurement of the first volunteer. To achieve this, power spectrum density (PSD) function was used.

The frequency components are often grouped into bands, these are the alpha, beta, gamma, delta and theta bands. The frequency limits of the bands are not precisely defined, the limits used are slightly different for different articles [5, 7, 11]. Ranges used by us are summarized in Table 1. Absolute power of the signal in the five frequency bands was also calculated. We used eight channels and five frequency bands, therefore we got 40 extra values for each time of sampling. These were also added to the data matrix to make them usable as inputs for the neural network.

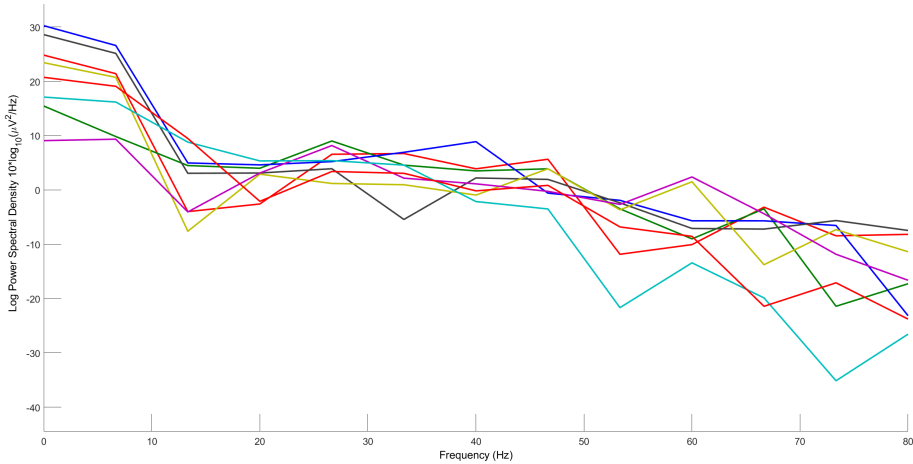


Figure 2. Power spectral density of a one-second window.

Table 1. Bands and their frequency ranges.

Band	Frequency range
Alpha	8-14 Hz
Beta	14-30 Hz
Gamma	30-80 Hz
Delta	1-4 Hz
Theta	4-8 Hz

To separate linearly mixed independent sources in EEG sensors, we performed independent component analysis (ICA). The result of the analysis is an invertible data decomposition. Several algorithms are provided by the EEGLAB. We used the Infomax because of its efficiency. In EEGLAB the Infomax algorithm returns two matrices, a data sphering matrix (*icasphere*) and the ICA weight matrix (*icaweights*) [1]. Their product is the unmixing matrix:

$$U = icaweights \times icasphere.$$

The product of the unmixing matrix and the raw data matrix is the activation matrix. Each row of this represents the time course of the activity of one component process spatially filtered from the raw data.

$$A = U \times rawdata.$$

After the ICA decomposition, power values were calculated again and both the activation matrix and the power matrix were added to the data matrix. Figure 3 shows the component map of the first measurement of the first subject. Eye artifacts can be clearly seen in EEG data.

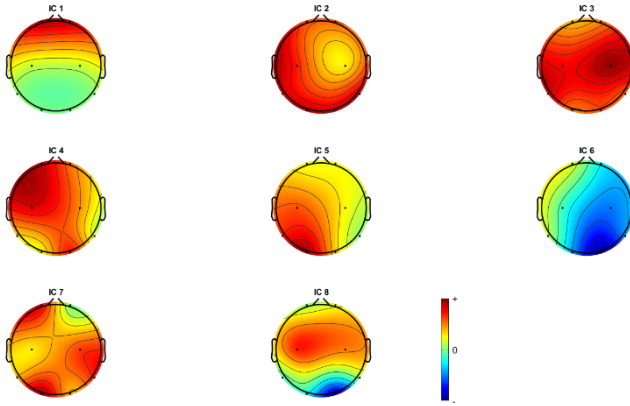


Figure 3. Component map of the first measurement of the first volunteer.

3.2. Machine learning chain

The machine learning process involves several steps and can therefore be considered as a chain. In our case, these steps are the import of the raw data, preprocessing (including normalization, segmentation and feature extraction), neural network instantiation, training and test.

The data from the sensors form a continuous stream of discrete values. Training the neural network with raw sensor data typically does not give adequate classification performance. As a result, some kind of data preprocessing is inevitable to improve it. As a first step, we used segmentation, which is the division of the available data into smaller units (windows).

Finding the appropriate window size is the main challenge in this task. If it is too small, it may not provide enough information about the performed activity, causing lower classification accuracy. Otherwise it may cover more than one activity in one window and causes increased latency. We tried numerous FFT window sizes for power spectral density calculation. Windows were overlapping, at each sample the window consisted the current sample and the previous $N-1$ samples. Then the order of the columns was shuffled randomly in the data matrix.

We used feedforward, multi-layer perceptron (MLP) networks and compared different architectures to improve accuracy. In all cases, the Levenberg-Marquardt backpropagation training method was used with the same training data: a two-minute motor imagery measurement from the first subject. Initial weights came from a normal Gaussian distribution, epoch limit was 1000 cycles. Number of hidden layers was one or two with 20 or 60 neurons. Activation functions were log-sigmoid and hyperbolic tangent sigmoid. Since there were three different classes, the number of neurons in the output layer was also three and its activation function was linear.

Best results were achieved with the two-layer network using log-sigmoid function

in both hidden layers as shown in Figure 4, thus in further work this was the applied architecture.

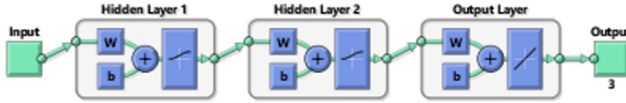


Figure 4. Structure of the applied neural network.

4. Results

Initially, we worked with a smaller amount of data (one two-minute measurement) and the indicator used to evaluate the performance of the network was accuracy. It gives the ratio of the correct predictions to total predictions:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}.$$

Using only the raw data (8 inputs), without any features, the classification accuracy on test data was 56.2%. By adding band powers too (calculated with 0.5 second window size), using 48 inputs, accuracy was significantly better, 91.5%. Using only the 40 power values did not cause a change in the classification performance, so in order to reduce training time, raw data were no longer used.

To find an appropriate FFT window size for band power calculation, neural network was trained using various windows sizes. Training and test with the data of a two-minute measurement from one subject were repeated five times to decrease statistical uncertainty. Applied neural network was the 2-layer MLP with 20-20 neurons in the hidden layers. Accuracy and its standard deviation on training and test data for different window sizes are summarized in Table 2.

Table 2. Accuracy for different window sizes.

Window size (sample)	Accuracy on test data
40	78.64 ± 1.30%
60	86.90 ± 0.61%
80	91.52 ± 0.69%
100	93.76 ± 1.33%
120	95.89 ± 0.15%
140	97.57 ± 0.47%
160	98.62 ± 0.21%
180	98.34 ± 0.41%
200	97.99 ± 0.33%

Based on the above results, a window size of 160 samples is a reasonable choice, as it contains enough information about the activity performed, but does not yet cause too much latency in the case of real-time data processing, since whenever the target appears on the screen, in approximately one second we are able to determine from the EEG signal what the subject sees.

The same 2-layer MLP with 20-20 neurons in the hidden layers was tested on another same-type data file from the same volunteer. Classification accuracy was only 42.1%. To improve accuracy, we merged the three same-type data matrix from the same subject to one bigger matrix. 70% of this data was used for training and the rest for testing. A mean squared error (MSE) of 0.076 was achieved for the training and 0.081 for the test data.

As a new approach, we processed data from 50 subjects using the leave-one-out method: data from 49 subjects were used for training, the last one for testing. MSE was 0.195 for the training and 0.207 for the test data.

In order to improve the classification accuracy of the neural network, we normalized the input data. Two different methods were used: in the first case, the raw data of the given one-second window were normalized before the power calculation, while in the second case, the data of the entire two-minute measurement were. MSE values were 0.196 (training) and 0.207 (test) in the first case, 0.197 (training) and 0.216 (test) for the second case. Normalization did not affect neural network performance in terms of MSE, so this was no longer used.

Table 3. Achieved MSE for the different experiments.

Neural network input	MSE training data	MSE test data
All data files from 1 subject 70% training, 30% test	0.076	0.081
All data files from 50 subjects training on 49, testing on 1 subject without normalization	0.195	0.207
All data files from 50 subjects training on 49, testing on 1 subject 1-second window is normalized	0.196	0.207
All data files from 50 subjects training on 49, testing on 1 subject 2-minute measurement is normalized	0.197	0.216
All data files from 50 subjects training on 49, testing on 1 subject with ICA, input is the activation matrix	0.206	0.204
All data files from 50 subjects training on 49, testing on 1 subject with ICA, input is the activation matrix power	0.197	0.199

Furthermore in order to improve the performance, we applied independent com-

ponent analysis (ICA) on raw data and the activation matrix served as input for the neural network. We got 0.206 MSE value for training and 0.204 for test data. Then we calculated the power values of the activation matrix and used the 40 power values as inputs. MSE values were 0.197 and 0.199 for training and test data, respectively. Results are summarized in Table 3.

In terms of classification accuracy, we found that the network recognized relaxation task with much greater accuracy than motor imagery tasks. This is because the data set was unbalanced, with much more data available for the relaxation task. To avoid this, we performed a balancing of the data set so that the same number of samples was available for all three activities. In addition, we increased the number of neurons to 60-60 in the hidden layers. The data of 10 volunteers were used, divided into two parts: 70% training, 30% test data. Training time on our computer (CPU: Intel Core i7-4790, RAM: 8 GB) was around 100 hours, accuracy for test data was 71.5%.

5. Conclusions

In this paper we proposed a multilayer perceptron (MLP) neural network approach for motor imagery task recognition from EEG data. In the longer term, the aim of our research is to use a neural network trained on a large dataset to analyze data recorded by our own device. Accordingly we used a publicly available dataset and tried to recognize the motor imagery activities of the subjects. We applied several data preprocessing methods and examined their effects on the performance of the neural network.

Our results demonstrated the importance of data preprocessing, and its effects on the classification performance of a neural network. However, it also showed that MLP might not be the best choice for EEG classification, as the classification accuracy score achieved is not high enough. J. Wang et al. [13] showed that compared with MLP, SVM and linear discriminant analysis (LDA), convolutional neural network (CNN) can provide better EEG signal classification performance. Tang et al. [12] applied conventional classification methods, such as power + support vector machine (SVM), CSP + SVM, autoregression (AR) + SVM and compared them with deep CNN in motor imagery EEG classification. The performance of CNN was better than the other three methods. Our preliminary results using CNN also show a much better accuracy score which is over 90%.

References

- [1] A. DELORME, S. MAKEIG: *EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis*, Journal of Neuroscience Methods 134.1 (2004), pp. 9–21, DOI: <https://doi.org/10.1016/j.jneumeth.2003.10.009>.

- [2] L. VAN DOKKUM, T. WARD, I. LAFFONT: *Brain computer interfaces for neurorehabilitation – its current status as a rehabilitation strategy post-stroke*, *Annals of Physical and Rehabilitation Medicine* 58.1 (2015), pp. 3–8,
DOI: <https://doi.org/10.1016/j.rehab.2014.09.016>.
- [3] L. A. GEMEIN, R. T. SCHIRRMESTER, P. CHRABAŚCZ, D. WILSON, J. BOEDECKER, A. SCHULZE-BONHAGE, F. HUTTER, T. BALL: *Machine-learning-based diagnostics of EEG pathology*, *NeuroImage* 220 (2020), p. 117021,
DOI: <https://doi.org/10.1016/j.neuroimage.2020.117021>.
- [4] A. L. GOLDBERGER, L. A. AMARAL, L. GLASS, J. M. HAUSDORFF, P. C. IVANOV, R. G. MARK, J. E. MIETUS, G. B. MOODY, C. K. PENG, H. E. STANLEY: *PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals*, *Circulation* 101.23 (2000), pp. 215–220,
DOI: <https://doi.org/10.1161/01.CIR.101.23.e215>.
- [5] I. S. ISA, B. S. ZAINUDDIN, Z. HUSSAIN, S. N. SULAIMAN: *Preliminary Study on Analyzing EEG Alpha Brainwave Signal Activities Based on Visual Stimulation*, *Procedia Computer Science* 42 (2014), pp. 85–920,
DOI: <https://doi.org/10.1016/j.procs.2014.11.037>.
- [6] B. KEMP, J. OLIVAN: *European data format ‘plus’ (EDF+), an EDF alike standard format for the exchange of physiological data*, *Clinical Neurophysiology* 114.9 (2003), pp. 1755–1761,
DOI: [https://doi.org/10.1016/S1388-2457\(03\)00123-8](https://doi.org/10.1016/S1388-2457(03)00123-8).
- [7] D. KUČIKIENĖ, R. PRANINSKIENĖ: *The impact of music on the bioelectrical oscillations of the brain*, *Acta medica Lituanica* 25.2 (2018), pp. 101–106,
DOI: <https://doi.org/10.6001/actamedica.v25i2.3763>.
- [8] I. LAZAROU, S. NIKOLOPOULOS, P. C. PETRANTONAKIS, I. KOMPATSIARIS, M. TSOLAKI: *EEG-Based Brain-Computer Interfaces for Communication and Rehabilitation of People with Motor Impairment: A Novel Approach of the 21 st Century*, *Frontiers in human neuroscience* 12 (2018),
DOI: <https://doi.org/10.3389/fnhum.2018.00014>.
- [9] R. MASKELIUNAS, R. DAMASEVICIUS, I. MARTISIUS, M. VASILJEVAS: *Consumer grade EEG devices: are they usable for control tasks?*, *PeerJ* 4 (2016),
DOI: <https://doi.org/10.7717/peerj.1746>.
- [10] Y. ROY, H. BANVILLE, I. ALBUQUERQUE, A. GRAMFORT, T. H. FALK, J. FAUBERT: *Deep learning-based electroencephalography analysis: a systematic review*, *Journal of Neural Engineering* 16.5 (2019),
DOI: <https://doi.org/10.1088/1741-2552/ab260c>.
- [11] J. SUTO, S. ONIGA, P. P. SITAR: *Music stimuli recognition from electroencephalogram signal with machine learning*, in: 2018 7th International Conference on Computers Communications and Control (ICCCC), 2018, pp. 260–264,
DOI: <https://doi.org/10.1109/ICCCC.2018.8390468>.
- [12] Z. TANG, C. LI, S. SUN: *Single-trial EEG classification of motor imagery using deep convolutional neural networks*, *Optik - International Journal for Light and Electron Optics* 130 (2017), pp. 11–18,
DOI: <https://doi.org/10.1016/j.ijleo.2016.10.117>.
- [13] J. WANG, G. YU, L. ZHONG, W. CHEN, Y. SUN: *Classification of EEG signal using convolutional neural networks*, in: 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2019, pp. 1694–1698,
DOI: <https://doi.org/10.1109/ICIEA.2019.8834381>.

TEE-based protection of cryptographic keys on embedded IoT devices*

Dorottya Papp, Máté Zombor, Levente Buttyán

Laboratory of Cryptography and System Security,
Department of Networked Systems and Services,
Budapest University of Technology and Economics
www.crysys.hu

Submitted: November 13, 2020

Accepted: February 11, 2021

Published online: May 18, 2021

Abstract

The Internet of Things (IoT) consists of billions of embedded devices connected to the Internet. Secure remote management of many of these devices requires them to store and use long-term cryptographic keys. In this work we propose to protect cryptographic keys in embedded IoT devices using a Trusted Execution Environment (TEE) which is supported on many embedded platforms. Our approach provides similar protection as secure co-processors, but does not actually require an additional secure hardware element.

Keywords: Trusted Execution Environment, cryptographic keys, key management

AMS Subject Classification: 68M25 (Computer Security)

1. Introduction

The Internet of Things (or IoT for short) consists of billions of embedded devices connected to the Internet. This new phenomenon is the basis for today's smart applications in the domains of manufacturing (Industry 4.0), transportation (Cooper-

*The presented work was carried out within the SETIT Project (2018-1.2.1-NKP-2018-00004), which has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the 2018-1.2.1-NKP funding scheme.

ative Intelligent Transportation Systems), and healthcare (personalized e-Health), as well as in everyday life (smart cities, smart homes). However, in almost all application areas of IoT, we face security and privacy issues which require solutions developed for or adapted to the special characteristics of IoT systems. Security and privacy mechanisms should take into account the resource limitations of embedded devices and they should not rely on special hardware that would significantly increase the development cost of IoT applications. This leads to interesting challenges for managing cryptographic keys on IoT devices.

In many applications, IoT devices are managed remotely by system operators. Such remote management requires secure remote access to the devices, which in turn, requires the devices to store and use long-term cryptographic keys. For instance, the operator usually needs to authenticate the device before uploading configuration data or software updates on it, which may require the device to use a long-term, device-specific private key. However, as IoT devices are connected to the Internet, they may be compromised by malicious actors (aka attackers). If an attacker can obtain the long-term key of a compromised device, (s)he can impersonate and clone that device, which is undesirable. Hence, there is a need to protect long-term cryptographic keys on IoT devices such that a key remains inaccessible to the attacker even if the device itself is compromised.

A possible solution to the problem above would be to store cryptographic keys on IoT devices in secure co-processors, such as a TPM chip¹ that would never output a key, but only use it internally in cryptographic operations. However, requiring an additional co-processor on every IoT device would be too expensive in most cases.

In this work we propose a more cost efficient approach: we ensure protection of cryptographic keys by using a Trusted Execution Environment (TEE), which is mostly based on software with some minimal hardware support, and it is supported on many embedded platforms used in IoT applications. For instance, many embedded devices use ARM processors that feature the ARM TrustZone technology², which enables the establishment of a software-based TEE and provides some hardware-based protection mechanisms to them. TEEs usually implement a persistent secure storage service (see, e.g., the TEE specifications³ of GlobalPlatform, a non-profit industry association aiming at enabling digital services and devices to be trusted and securely managed throughout their lifecycle), which can be used to store long-term cryptographic keys. Moreover, operations with those keys can be performed by trusted applications running within the TEE, hence, the keys would never leave the protected environment of the TEE.

¹<https://trustedcomputinggroup.org/resource/tpm-library-specification> (last accessed: Oct 3, 2020)

²<https://developer.arm.com/ip-products/security-ip/trustzone> (last accessed: Oct 3, 2020)

³<https://globalplatform.org/specs-library/?filter-committee=tee> (last accessed: Oct 3, 2020)

2. Background

Long-term cryptographic keys have been traditionally protected using additional hardware elements, such as Hardware Security Modules (HSMs) or Trusted Platform Modules (TPMs) and secure co-processors. These hardware components provide cryptographic operations to implement secure boot, trustworthy reporting, attestation, and other components of secure computing [2]. HSMs are external hardware modules which can be attached to existing computer systems and used via PCI, USB, or network connection. They provide cryptographic functionality, as well as tamper-resistance, and are often used to securely generate, store and use cryptographic keys. Typically, HSMs implement PKCS #11⁴, a platform-independent API to handle cryptographic tokens. The API itself is called Cryptoki and has header files for C and C++ applications; vendors usually have their own compliant implementations. There exists also software-based HSM implementations, for example, the SoftHSM⁵, which is a well maintained open source project. It is part of the OpenDNSSEC project⁶ with goal of being a complete implementation of PKCS #11.

TPMs, on the other hand, are chips embedded on the computer's motherboard and offer several security-relevant features in a standardized manner: protected memory and registers to securely execute commands, tamper-evident hardware module to store keys, cryptographic processing capability and a true random number generator. They are usually used as hardware roots of trust for measurement, storage and reporting, as well as to implement critical functionalities. TPM chips are commercially available on the market [3] and there is research effort [1, 14] to implement the same concepts in software.

The main disadvantage of the previously mentioned hardware-based solutions is that they are additional and often costly components of the system. By comparison, IoT devices are constrained not only in resources but in cost as well [3]. As a result, hardware-based protection for cryptographic keys is not viable economically in the IoT setting. There exists software-based implementations of the hardware concepts, but those are typically implemented as kernel modules which could be compromised by an attacker with elevated privileges.

However, there exists an emerging technology which can provide a secure and integrity-protected processing environment: the TEE. TEE runs on the same hardware as the device's main operating system (OS) but it is also isolated at the hardware-level. Many chips used in embedded devices offer the hardware support necessary to realize Trusted Execution Environments [15]. Examples include the ARM TrustZone⁷, the Intel Software Guard eXtension⁸ (SGX) [9], and the AMD

⁴<http://docs.oasis-open.org/pkcs11/pkcs11-base/v2.40/os/pkcs11-base-v2.40-os.html> (last accessed: Nov 04, 2020)

⁵<https://www.opendnssec.org/softhsm/> (last accessed: Nov 04, 2020)

⁶<https://www.opendnssec.org/> (last accessed: Nov 04, 2020)

⁷<https://developer.arm.com/ip-products/security-ip/trustzone> (last accessed: Oct 12, 2020)

⁸<https://software.intel.com/content/www/us/en/develop/documentation/sgx-developer-guide/top.html> (last accessed: Oct 12, 2020)

Secure Encrypted Virtualization [5]. OP-TEE⁹ and Open-TEE [7] are two TEE implementations which can be deployed on these chips.

Figure 1 shows the main components of a device with TEE capabilities. Logically, execution can be separated into the Rich Execution Environment (REE) and the TEE. Code running in the REE has access only to unprotected resources (e.g. memory). “Code” in the REE can be partitioned into the Rich operating system, usually a traditional OS such as Linux, and one or more applications, which run on top of the Rich OS. Such an application is called a Client Application (CA) in the TEE architecture. CAs implement the basic features of the device, e.g. web servers for configuration, applications for sensing physical parameters of the environment, or the actuator controlling a physical process. When necessary, CAs can request services from the TEE via the TEE Client API. This API forwards the request to a special component in the Rich OS, the REE Communication Agent, which triggers a context switch and gives control to the TEE.

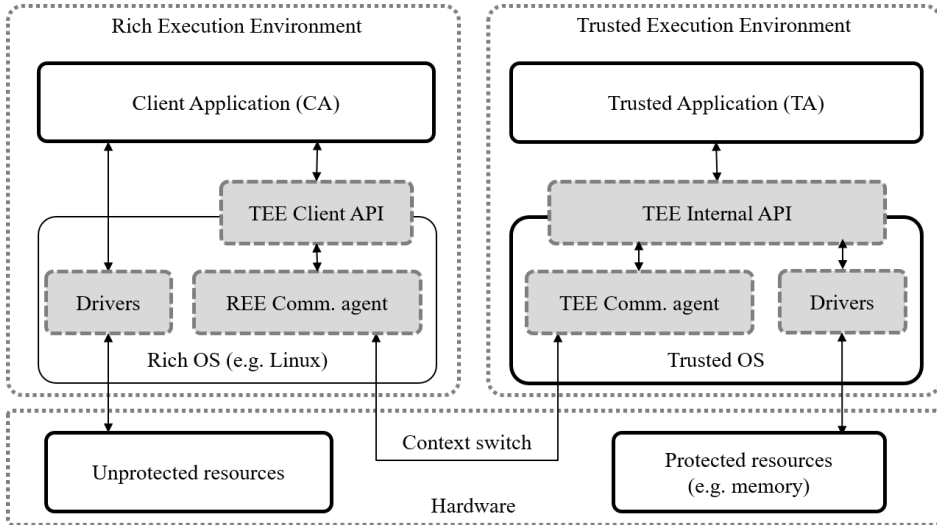


Figure 1. Logical overview of a device with Trusted Execution Environment capabilities.

Code in the TEE has access to protected resources, which are unavailable to the REE. For example, certain memory locations are only available to code running in the TEE. This protection is provided by the hardware components of the device. In the case of the ARM TrustZone, for example, the architecture includes a special register storing the Non-secure (NS) bit to determine whether the executed code belongs to the REE or the TEE. If the NS bit is set, signalling that the executed code belongs to the REE, access to certain protected memory locations is automatically denied. The TEE is similar to the REE in the sense that it has

⁹<https://optee.readthedocs.io/en/latest/index.html> (last accessed: Oct 12, 2020)

an operating system (the trusted OS) and several applications, which are called Trusted Applications (TAs). TAs provide those services for the REE whose computation requires strong security guarantees, for example, remote attestation [12, 13], tamper-resistant logging and storage [10, 11], or secure real-time computation for the Industrial IoT [8].

3. Architectural overview

The basic idea of our approach is to use the TEE to provide similar protection to keys as a secure co-processor but without actually requiring another processor on the device: the same processor runs a normal execution environment (the REE) and a TEE, and also implements the required hardware mechanisms that isolate these two execution environments. This isolation ensures that even if the REE is compromised, the attacker would not be able to obtain the keys stored and used within the TEE. This protection mechanism prevents attackers to clone compromised devices.

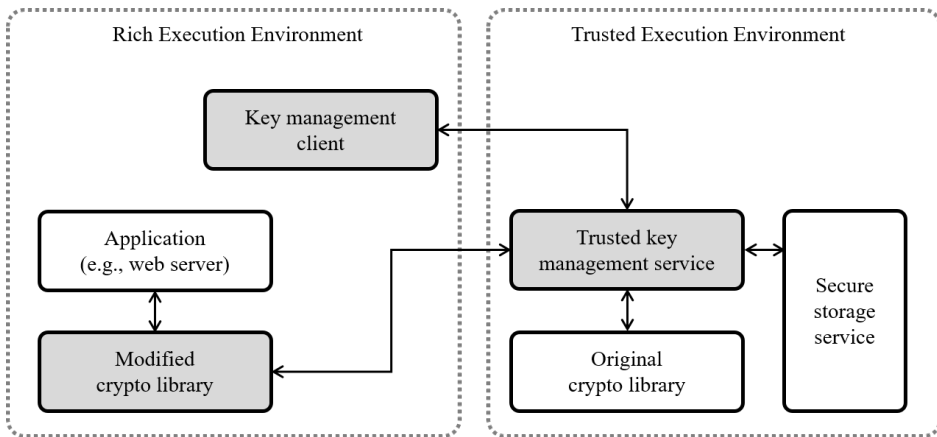


Figure 2. Architecture of our TEE based key management solution. Grey boxes represent components that we developed or modified.

The architecture of our solution is illustrated in Figure 2. Private keys and private-public keypairs are stored in the secure storage of the TEE. We also store the intended use of keys, e.g. signing or decryption, in an additional attribute in the TEE. The keys could be generated by the operator off-line and loaded in the secure storage in a controlled way with the help of a key management client, or the key can actually be generated and stored in secure storage by the trusted key management service itself. In the latter case, the trusted key management service would output the corresponding public key to the key management client such that it can be made available to applications running outside of the TEE. In both cases, handles to the private keys would be output from the trusted key management

service that can be used by applications in the REE to refer to the private keys when requesting operations with them.

Generating and loading keys into the TEE should only be performed by the device’s operator, therefore, such requests must be authenticated. Request authentication requires the operator to set up a master password before the device is deployed. The trusted key management service allows the key management client to install the master password only once, it cannot be changed later. Requests related to key management must provide not only the invocation parameters to the underlying cryptographic library but also a salt and a message authentication code (MAC). We refer to the combination of salt and MAC as the authentication token. The master password is used together with the salt to derive a key. The derived key and the invocation parameters are input to HMAC (RFC 2104¹⁰) and its output is compared with the MAC value supplied in the request. The request authentication process is illustrated in Figure 3. Key management operations are only performed, if the HMAC-based authentication scheme succeeds without errors. We also log authentication tokens in the trusted key management service to prevent replay attacks with previous key management requests. If a request contains a previously used authentication token, the request is automatically denied.

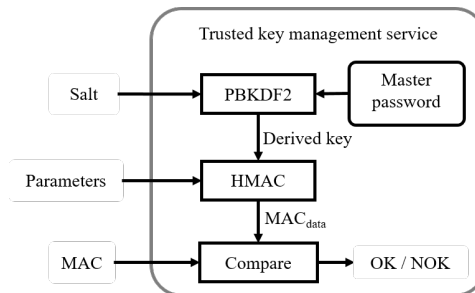


Figure 3. Process overview of authenticating requests from the key management client in the trusted key management service.

Any application (e.g., a web server that provides a remote configuration possibility for the operator of the device) that runs in the REE can be compiled with a cryptographic library that we modified such that private key cryptographic operations are delegated to the trusted key management service running in the TEE. In TEE terminology, the modified cryptographic library acts as a CA and the trusted key management service is a TA. From the application’s point of view, the cryptographic library exposes functions to encrypt and decrypt data, which can be invoked similarly to API functions, as shown in Figure 4. However, instead of supplying the key itself, the application provides a handle to the private key with which to perform the cryptographic operation. The modified cryptographic library serialized the key handle and the provided parameters as a message and passes

¹⁰<https://tools.ietf.org/html/rfc2104> (last accessed: Nov 12, 2020)

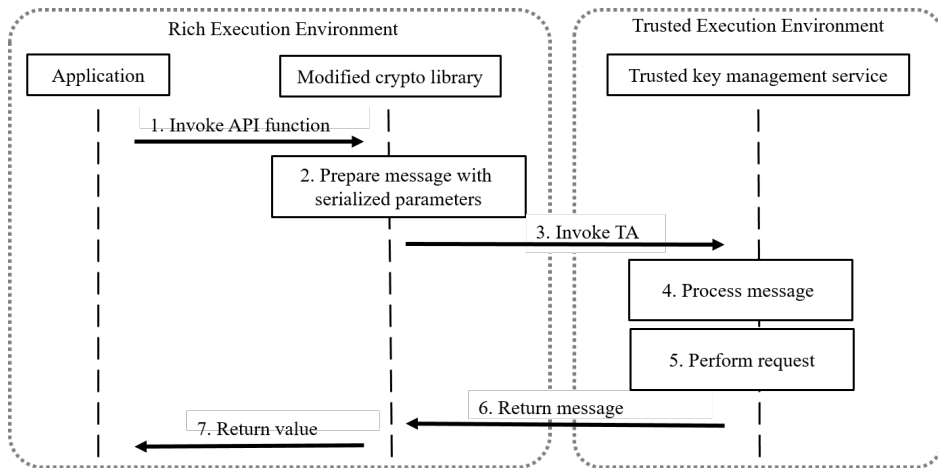


Figure 4. Interactions involved in performing cryptographic operations with private keys stored and managed in the TEE.

that to the trusted key management service. The trusted key management service, which is compiled with the original cryptographic library, processes the serialized parameters, retrieves the key referred by the provided key handle, and calls the original cryptographic library to execute the requested operation. The results are passed back to the modified cryptographic library and the modified cryptographic library provides the return value to the application.

The two components can pass parameters and values to each other via shared memory: a block of memory which is shared between the CA and the TA. Both the CA and the TA can read data from and write data to the shared memory, however, only the CA can allocate it. Therefore, the modified cryptographic library must allocate memory to hold the results of cryptographic operations. Knowing the requested operation and information about the key, the modified cryptographic library can estimate the necessary amount of memory. If the modified cryptographic library underestimated the amount of memory, the trusted key management service returns a special message requesting more memory to return the result.

4. Prototype implementation

We implemented the proposed TEE-based architecture for protecting long-term cryptographic keys using the Trusted Firmware¹¹ projects OP-TEE and mbedtls. Trusted Firmware provides a reference trusted code base for the ARM platform, a widely used platform in embedded devices. OP-TEE is an open source implementation of GlobalPlatform’s TEE specification, primarily maintained by Linaro, and it is usually used in conjunction with the Linux kernel in the REE. mbedtls is a

¹¹<https://www.trustedfirmware.org/> (last accessed: Nov 03, 2020)

cryptographic library written in C with a small code footprint. It can be used in both the REE and the TEE; OP-TEE can be compiled to use `mbedtls` as the default cryptographic library.

For our prototype implementation, we set `mbedtls` as OP-TEE's default cryptographic library. We implemented a Trusted Application which fulfills the role of trusted key management service and handles incoming requests for cryptographic and key management operations. The Trusted Application stores the key pair object in the secure storage and passes it to `mbedtls` whenever cryptographic operations are to be performed. We also compiled a modified version of `mbedtls`'s source code in the REE such that it includes wrapper functions to direct requests to our Trusted Application. Our prototype implementation consists of eight wrapper functions as follows:

- `tee_set_master_password`: Installs the specified master password into the trusted key management service to authenticate key management requests. This function can only be called once, we assume that it is done in a controlled environment by the device's operator.
- Key management functions: These functions perform privileged operations allowed only for the operator. Therefore, the Trusted Application performs the request authentication process described in Section 3 on their inputs.
 - `tee_generate_keypair`: Generates a long-term private-public key pair and stores it in the TEE. The function returns a handle to the key pair which can be later used for other cryptographic operations.
 - `tee_load_keypair`: Allows the operator to load an existing key pair into the TEE. The key pair must be encrypted and in PEM format. Similarly to `tee_generate_keypair`, this function also returns a handle to the key pair.
 - `tee_remove_keypair`: If a key pair becomes compromised or is considered weak, the operator can inactive it. We do not permanently delete keys because the attacker might try to reinstall old and weak keys. Instead, inactivating keys allows us to maintain a list of all previously and currently used keys. The list could be reviewed by the operator or used for attestation purposes.
- Functions available for all applications: All of these functions reference a key stored in the TEE with a key handle. In our prototype, handles are 32 bytes long and calculated as the SHA256 hash value of the key pair.
 - `tee_pk_decrypt`: Decrypts the supplied data with a given key.
 - `tee_pk_sign`: Digitally signs the input data with a given key.
 - `tee_get_keyinfo`: Returns the type, the size, and the intended usage of a given key.
 - `tee_get_publickey`: Returns the public key of a public-private key pair in plaintext.

For each function, we defined a custom message format which can hold the serialized parameters to be passed to the trusted key management service prototype. In all cases, messages start with an ID field identifying the operation requested, followed by the key handle. Depending on the function, the key handle can be an input parameter and an output parameter. For example, the function `tee_remove_keypair` expects a key handle as an input, while for the function `tee_load_keypair`, the field for the key handle is empty and must be filled with the handle assigned by the TA. The remainder of the message formats follow the length-value convention: first comes the length of the data as an 8-byte-long unsigned integer, then the data as a variable length field.

5. Evaluation

In order to measure the added overhead of TEE-based key protection, we conducted the following experiment. We set up a QEMU-based¹² environment for running our prototype implementation and manually saved an RSA long-term key pair in the TEE. We deployed two versions of `mbedtls`'s example web server with TLS capabilities in REE: one without any modifications and another with the modification to relay cryptographic operations to our trusted key management service prototype. We used `mbedtls`'s example client to test the connection to the web server and repeatedly send HTTP GET requests to both versions.

Our experiment was concerned with the amount of time required to perform cryptographic operations using our trusted key management service prototype. We sent 10 HTTP GET requests from the client to the webserver and measured the amount of time it took for the sign operation to complete. Communicating parties used the `TLS-ECDHE-RSA-WITH-CHACHA20-POLY1305-SHA256` cipher suite during the TLS Handshake. The communication between client and server succeeded in all 10 exchanges. In case of the unmodified `mbedtls` operations, all operations take place in REE memory. In case of our trusted key management service prototype, the measured amount of time includes the context switch between REE and TEE, as well as the time necessary to perform the requested operation and return the result.

The results of the experiment are shown in Table 1. Our trusted key management service prototype needed an average of 204 ms for the sign operation. This is 5x slower than `mbedtls`'s unmodified operations which take place in REE memory. However, it is worth noting that after the first run, `mbedtls`'s unmodified operations gain a performance boost: their required time to complete changes from 87 ms to ca. 30 ms. This performance boost is the result of `mbedtls`'s implementation to prevent timing attacks. The authors of [6] presented timing attacks in which they measured the amount of time required to perform private key operations, consequently finding fixed Diffie-Hellman exponents and factor RSA keys. The proposed protection against such attacks involves the use of *blinding values*, a pair of ran-

¹²<https://www.qemu.org/> (last accessed: Nov 10, 2020)

dom numbers (v_i, v_f) such that in the case of Diffie-Hellman, $v_f = (v_i^{-1})^x \bmod n$, and in the case of RSA, $v_i = (v_f^{-1})^e \bmod n$. The chosen numbers are then used similarly to blind signatures [4]: the input is multiplied by v_i and the result is corrected by multiplying it with $v_f \bmod n$. However, computing the inverses is slow, therefore, `mbedtls`'s implementation uses SSL session information to determine whether (v_i, v_f) has been chosen before and if yes, it updates their values by squaring. Unfortunately, our trusted key management service does not have access to SSL session information and must select a new random (v_i, v_f) pair for each computation.

Table 1. Comparisons between the performance of the unmodified `mbedtls` library and our trusted key management service prototype in the TEE. The first two columns show the performance of the operation on the server-side, while the last two columns show the amount of time required to build a secure communication channel and exchange an HTTP GET request and response between the client and the server.

	<code>mbedtls</code> 's sign operation	Our TEE-based sign operation	Communication using <code>mbedtls</code>	Communication using TEE
Run 1	87 ms	208 ms	410 ms	533 ms
Run 2	30 ms	204 ms	331 ms	505 ms
Run 3	30 ms	203 ms	341 ms	516 ms
Run 4	29 ms	206 ms	319 ms	501 ms
Run 5	29 ms	204 ms	305 ms	507 ms
Run 6	30 ms	203 ms	326 ms	522 ms
Run 7	29 ms	203 ms	312 ms	504 ms
Run 8	29 ms	206 ms	315 ms	511 ms
Run 9	38 ms	202 ms	388 ms	503 ms
Run 10	33 ms	204 ms	341 ms	500 ms
Mean	36 ms	204 ms	339 ms	508 ms
Std.dev	18 ms	2 ms	34.22 ms	10.59 ms

From the client's perspective, completing a full TLS handshake and exchanging an HTTP GET request and response over the secure channel is 1.49x slower, if cryptographic operations with the long-term key are performed in the TEE. In case of the unmodified `mbedtls` library, the exchange takes 339 ms on average, while in case of our trusted key management service prototype, the same exchange is completed in 508 ms on average. The results in Table 1 suggest that network latency and SSL session management in both cases accounts for ca. 300 ms. Thus, the increased time necessary to complete the exchange using our trusted key management service in the TEE is the result of the overhead caused by the TEE-based sign operation.

6. Conclusion and future work

Remote administration is one of the key enabling features of IoT devices. However, remote administration requires secure communication channels, which in turn require the protection of long-term cryptographic keys. Traditionally, such keys are protected using additional hardware components, however, the cost of including such components in IoT devices is economically unviable.

In this paper we proposed Trusted Execution Environments as alternative. Their main advantage is that they are mostly software components requiring minimal hardware support for isolation. Our basic idea is to use the TEE's secure storage to protect keys in rest and run cryptographic libraries in the TEE which can protect the keys during execution thanks to access to protected resources. Our architecture includes a trusted key management service in the TEE whose task is to handle the TEE's secure storage and invoke the cryptographic library inside the TEE. Applications not running in the TEE can request operations from the trusted key management service. We created a prototype implementation of the proposed architecture using OP-TEE, an open-source TEE implementation, and mbedtls, a cryptographic library designed to run on small devices. We measured the performance overhead of performing cryptographic operations in the TEE. While there certainly was an overhead due to context switches, the overhead we measured was bearable and did not threaten the communication between client and server. Thus, we can conclude that TEEs are indeed viable alternatives to HSMs and TPMs to protect long-term cryptographic keys.

Other security-critical operations could be implemented in the TEE, as well. Our current research ideas include integrity monitoring from the TEE and using the results for remote attestation of IoT devices. One of the main challenges of remote attestation is how to ensure the trustworthiness of attestation results in the presence of an attacker. TEEs can solve this problem: even if the attacker compromises the main operating system, the device's hardware support for TEEs isolates the attestation process and cryptographic keys from the attacker.

References

- [1] N. AARAJ, A. RAGHUNATHAN, N. K. JHA: *Analysis and Design of a Hardware/Software Trusted Platform Module for Embedded Systems*, ACM Trans. Embed. Comput. Syst. 8.1 (Jan. 2009), ISSN: 1539-9087, DOI: <https://doi.org/10.1145/1457246.1457254>.
- [2] A. AVIZIENIS, J. LAPRIE, B. RANDELL, C. LANDWEHR: *Basic concepts and taxonomy of dependable and secure computing*, IEEE Transactions on Dependable and Secure Computing 1.1 (2004), pp. 11–33, DOI: <https://doi.org/10.1109/TDSC.2004.2>.
- [3] T. BROSTRÖM, J. ZHU, R. ROBUCCI, M. YOUNIS: *IoT Boot Integrity Measuring and Reporting*, SIGBED Rev. 15.5 (Nov. 2018), pp. 14–21, DOI: <https://doi.org/10.1145/3292384.3292387>.

- [4] D. CHAUM: *Blind Signatures for Untraceable Payments*, in: *Advances in Cryptology*, ed. by D. CHAUM, R. L. RIVEST, A. T. SHERMAN, Boston, MA: Springer US, 1983, pp. 199–203, ISBN: 978-1-4757-0602-4.
- [5] D. KAPLAN, J. POWELL, T. WOLLER: *AMD Memory Encryption*, tech. rep., http://amd-dev.wpengine.netdna-cdn.com/wordpress/media/2013/12/AMD_Memory_Encryption_Whitepaper_v7-Public.pdf, Last visited: 13.10.2020, 2016.
- [6] P. C. KOCHER: *Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems*, in: *Advances in Cryptology | Crypto*, vol. 96, 1996, p. 104113.
- [7] B. MCGILLION, T. DETTENBORN, T. NYMAN, N. ASOKAN: *Open-TEE – An Open Virtual Trusted Execution Environment*, in: TRUSTCOM '15, USA: IEEE Computer Society, 2015, pp. 400–407, ISBN: 9781467379526, DOI: <https://doi.org/10.1109/Trustcom.2015.400>.
- [8] S. PINTO, T. GOMES, J. PEREIRA, J. CABRAL, A. TAVARES: *IloTEED: An Enhanced, Trusted Execution Environment for Industrial IoT Edge Devices*, *IEEE Internet Computing* 21.1 (2017), pp. 40–47.
- [9] M. SCHWARZ, D. GRUSS: *How Trusted Execution Environments Fuel Research on Microarchitectural Attacks*, *IEEE Security Privacy* 18.5 (2020), pp. 18–27.
- [10] D. J. SEBASTIAN, U. AGRAWAL, A. TAMIMI, A. HAHN: *DER-TEE: Secure Distributed Energy Resource Operations Through Trusted Execution Environments*, *IEEE Internet of Things Journal* 6.4 (2019), pp. 6476–6486.
- [11] C. SHEPHERD, R. N. AKRAM, K. MARKANTONAKIS: *EmLog: Tamper-Resistant System Logging for Constrained Devices with TEEs*, in: *Information Security Theory and Practice*, ed. by G. P. HANCKE, E. DAMIANI, Cham: Springer International Publishing, 2018, pp. 75–92, ISBN: 978-3-319-93524-9.
- [12] C. SHEPHERD, R. N. AKRAM, K. MARKANTONAKIS: *Establishing Mutually Trusted Channels for Remote Sensing Devices with Trusted Execution Environments*, in: *Proceedings of the 12th International Conference on Availability, Reliability and Security, ARES '17*, Reggio Calabria, Italy: Association for Computing Machinery, 2017, ISBN: 9781450352574, DOI: <https://doi.org/10.1145/3098954.3098971>.
- [13] C. SHEPHERD, R. N. AKRAM, K. MARKANTONAKIS: *Remote Credential Management with Mutual Attestation for Trusted Execution Environments*, in: *Information Security Theory and Practice*, ed. by O. BLAZY, C. Y. YEUN, Cham: Springer International Publishing, 2019, pp. 157–173, ISBN: 978-3-030-20074-9.
- [14] M. STRASSER, H. STAMER: *A Software-Based Trusted Platform Module Emulator*, in: *Trusted Computing - Challenges and Applications*, ed. by P. LIPP, A.-R. SADEGHI, K.-M. KOCH, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 33–47, ISBN: 978-3-540-68979-9.
- [15] F. ZHANG, H. ZHANG: *SoK: A Study of Using Hardware-Assisted Isolated Execution Environments for Security*, in: *Proceedings of the Hardware and Architectural Support for Security and Privacy 2016, HASP 2016*, Seoul, Republic of Korea: Association for Computing Machinery, 2016, ISBN: 9781450347693, DOI: <https://doi.org/10.1145/2948618.2948621>.

Compositional trend filtering*

Christopher Rieser, Peter Filzmoser

TU Wien

Institute of Statistics and Mathematical Methods in Economics

`christopher.rieser@tuwien.ac.at`

`P.Filzmoser@tuwien.ac.at`

Submitted: November 30, 2020

Accepted: February 17, 2021

Published online: May 18, 2021

Abstract

Trend filtering is known as the technique for detecting piecewise linear trends in univariate time series. This technique is extended to the setting of compositional data, which are multivariate data where only the relative information is of importance. According to this, we formulate the problem and present a procedure how to efficiently solve it. To show the usefulness of this method, we consider the number of COVID-19 infections in several European countries in a chosen time period.

Keywords: Trend filtering, compositional data, COVID-19

1. Introduction

Filtering linear trends of a univariate time series has been extensively investigated in the literature, and many methods exist for this purpose, see [7] or [11]. In the univariate context, estimating a piecewise linear trend and its change points can deliver valuable insights and serve as an analytical tool. The standard l_1 linear trend filtering estimator for given measurements $y_t \in \mathbb{R}$ with equidistant time stamp $t = 1, \dots, T$ is given as the solution of the following optimisation problem

$$\min_{a_t} \frac{1}{2} \sum_{t=1}^T \|y_t - a_t\|^2 + \frac{\lambda}{2} \sum_{t=3}^T |a_t - 2a_{t-1} + a_{t-2}|,$$

*This research was supported by the Austrian Science Fund (FWF) under the grant number P 32819 Einzelprojekte.

for a fixed positive tuning parameter λ . The term on the left controls the goodness of fit to the data, whereas the right penalty term enforces the parameters a_t to be close to a linear function in t . This follows from the fact that a lasso penalty [12] is used on the second differences, i.e. $a_t - 2a_{t-1} + a_{t-2}$, setting the latter for certain t – depending on λ – to zero. It is easy to see that when $a_{t_j} - 2a_{t_j-1} + a_{t_j-2} = 0$ holds for consecutive t_j , with $j = 1, \dots, K$, we get $a_{t_j} = a + bt_j$, for fixed $a, b \in \mathbb{R}$. Therefore, by using a lasso penalty on the second differences we get that a_t is forced to become piecewise linear with growing λ . Recently, trend filtering has also been extended to many other contexts keeping the property that for a growing penalty parameter λ linear functions in the appropriate setting are selected; e.g. graphs [15], vector-valued graphs [13] and additive models [10].

To the best of our knowledge, so far, trend filtering has not been extended to compositional data. Compositional data are in its nature multivariate and strictly positive, and for this type of data it is the relative rather than the absolute information which is of interest. As an example we might consider chemical concentrations for $D \geq 2$ different elements. An observation is written as a D dimensional vector $\mathbf{x} = (x_1, \dots, x_D)'$ with positive entries x_1 to x_D , where each entry is the concentration of a certain element. In a chemical setting it is the relative information between different elements which is of main interest. Relative meaning that the important information is contained in the ratios for two different elements, i.e. $\frac{x_i}{x_j}$, for $i \neq j \in \{1, \dots, D\}$. In the ground breaking work [1], not only has it been made clear how a compositional view can be advantageous, but also the mathematical foundations of compositional data have been laid out.

In this work we consider fitting linear trends to a time series of compositional data, i.e. we look at the multivariate linear trend of compositional data with an equidistant time stamp t . As an example, consider the case where we compare the number of healthy individuals to infected ones in the whole population of a country. Denoting the number of infected individuals at each time t as κ_t and the total number of individuals in a country by P , the infected vs. non-infected individuals in a population over time can be described by the two dimensional time series $(\kappa_t, P - \kappa_t)$. If we are interested in analyzing how the number of healthy vs unhealthy individuals behaves we can see this as a compositional time series.

This perspective is relevant in many other contexts, e.g. comparing the performance of different stocks relative to each other. This explains the success compositional data analysis has had in past applications. The method we propose in this work guarantees to find an estimator of piecewise compositional linear trends. The trend estimates will be strictly positive and sum up to a given total.

This paper is organized as follows. In Section 2 we will review some important compositional data analysis concepts. In Section 3 we introduce compositional trend filtering, and in Section 4 we present a procedure for the computation. Section 5 shows an application of the presented method to the number of COVID-19 infected individuals in various European countries, and the final Section 6 concludes.

2. Compositional data analysis concepts

In the following, consider a composition $\mathbf{x} = (x_1, \dots, x_D)'$ with D strictly positive entries, called compositional parts, which sum up to 1. This leads to the definition of compositional data as observations from the D part simplex \mathcal{S}^D ,

$$\mathcal{S}^D := \left\{ \mathbf{x} = (x_1, \dots, x_D)' \in \mathbb{R}_+^D, \sum_{i=1}^D x_i = 1 \right\}.$$

As established in [1], compositional data aims to capture the relative information between the entries of \mathbf{x} . This is done by identifying each point $\mathbf{x} \in \mathbb{R}_+^D$ with a whole ray starting at zero and going through \mathbf{x} , which means that we identify an element $\mathbf{x} \in \mathcal{S}^D$ with all elements $\gamma\mathbf{x}$ for any $\gamma > 0$. In other words, the constraint of sum equal to 1 can always be achieved by rescaling, see [5].

The simplex can be equipped with an addition, multiplication with a scalar, an inner product and a norm, which leads to the so-called Aitchison geometry on the simplex [1]. Consider the compositions $\mathbf{x} = (x_1, \dots, x_D)'$ and $\mathbf{y} = (y_1, \dots, y_D)'$:

- For $\mathbf{x}, \mathbf{y} \in \mathcal{S}^D$, perturbation is defined by $\mathbf{x} \oplus \mathbf{y} := (x_1 y_1, \dots, x_D y_D)'$
- For $\mathbf{x} \in \mathcal{S}^D$ and $\alpha \in \mathbb{R}$, powering is defined by $\alpha \odot \mathbf{x} := (x_1^\alpha, \dots, x_D^\alpha)'$
- For $\mathbf{x}, \mathbf{y} \in \mathcal{S}^D$, the inner product is defined as

$$\langle \mathbf{x}, \mathbf{y} \rangle_A := \frac{1}{2D} \sum_{i=1}^D \sum_{j=1}^D \log \left(\frac{x_i}{x_j} \right) \log \left(\frac{y_i}{y_j} \right).$$

Remark 2.1. The difference of \mathbf{x} and \mathbf{y} denoted by $\mathbf{x} \ominus \mathbf{y}$ is therefore $(\frac{x_1}{y_1}, \dots, \frac{x_D}{y_D})'$. The norm can be defined in the usual manner using the inner product defined above, i.e $\|\mathbf{x}\|_A = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle_A}$.

Interestingly, one can construct isometric mappings from \mathcal{S}^D into \mathbb{R}^{D-1} , using the so called (Centered Logratio Coefficients) clr-mapping, which is defined as

$$\text{clr} : \mathcal{S}^D \rightarrow \mathbb{R}^D, \quad \text{clr}(\mathbf{x}) := \left(\log \left(\frac{x_1}{\sqrt[D]{\prod_{i=1}^D x_i}} \right), \dots, \log \left(\frac{x_D}{\sqrt[D]{\prod_{i=1}^D x_i}} \right) \right)'.$$

This mapping fulfills important properties regarding addition, multiplication by a scalar and the inner product, namely

$$\text{clr}(\mathbf{x} \oplus \mathbf{y}) = \text{clr}(\mathbf{x}) + \text{clr}(\mathbf{y}) \tag{2.1}$$

$$\text{clr}(\alpha \odot \mathbf{x}) = \alpha \text{clr}(\mathbf{x}) \tag{2.2}$$

$$\langle \mathbf{x}, \mathbf{y} \rangle_A = \langle \text{clr}(\mathbf{x}), \text{clr}(\mathbf{y}) \rangle_E \tag{2.3}$$

where $\langle \cdot, \cdot \rangle_E$ denotes the standard inner product in \mathbb{R}^D . However, clr is not a one-to-one mapping onto \mathbb{R}^D as for each $\mathbf{x} \in \mathcal{S}^D$ we have $\sum_{i=1}^D \text{clr}(\mathbf{x})_i = 0$; meaning that the sum of the entries of $\text{clr}(\mathbf{x})$ is always zero and therefore the image of clr lies in the subspace $\{z \in \mathbb{R}^D \mid \sum_{i=1}^D z_i = 0\}$.

Nevertheless, fixing $D - 1$ orthonormal basis vectors $\mathbf{v}_1, \dots, \mathbf{v}_{D-1} \in \mathbb{R}^D$, denoted in the following in matrix form $\mathbf{V} \in \mathbb{R}^{D \times (D-1)}$, of the subspace $\{z \in \mathbb{R}^D \mid \sum_{i=1}^D z_i = 0\} \subset \mathbb{R}^D$ one can define an isometry from \mathcal{S}^D to \mathbb{R}^D by

$$\text{ilr}_{\mathbf{V}} : \mathcal{S}^D \rightarrow \mathbb{R}^{D-1}, \quad \text{ilr}_{\mathbf{V}}(\mathbf{x}) := \mathbf{V}' \text{clr}(\mathbf{x}),$$

where \mathbf{V}' denotes the transposed matrix, called Isometric Logratio. $\text{ilr}_{\mathbf{V}}$ naturally preserves the properties (2.1), (2.2) and (2.3) and is an isometry. This mapping will be used in the following for trend filtering.

For a more thorough explanation of compositional data and different coordinate representations we refer to [5].

In the following we will speak of a compositional time series when talking about a time series $\mathbf{s}_t \in \mathcal{S}^D$, with time index $t = 1, \dots, T$. It is interesting to note here that due to the compositional nature of \mathbf{s}_t we can multiply the latter with any univariate time series $P_t \in \mathbb{R}_+$ such that $P_t \mathbf{s}_t$ still lies in \mathcal{S}^D for each t . In a compositional setting, $P_t \mathbf{s}_t$ is therefore equivalent to \mathbf{s}_t . This means that if we would like to go back from a compositional view to absolute numbers one needs to prespecify P_t . We will see how to do that in the next section. In the following, we will write $\mathbf{x}_t = P_t \mathbf{s}_t$, where P_t is given by the user.

3. Compositional trend filtering

We will now show how to extend the linear univariate trend filtering framework to compositional time series $\mathbf{x}_t \in \mathcal{S}^D$ and discuss why the basic property of fitting piecewise linear trends is kept.

We define the trend filtering estimator of a compositional time series \mathbf{x}_t as:

$$(\hat{\mathbf{a}}_1, \dots, \hat{\mathbf{a}}_T)' := \arg \min_{\mathbf{a}_t \in \mathcal{S}^D} \frac{1}{2} \sum_{t=1}^T \|\mathbf{x}_t \ominus \mathbf{a}_t\|_A^2 + \frac{\lambda}{2} \sum_{t=3}^T \|\Delta^2 \mathbf{a}_t\|_A \quad (3.1)$$

where $\Delta^2 \mathbf{a}_t$ denotes $\mathbf{a}_t \ominus 2\mathbf{a}_{t-1} \oplus \mathbf{a}_{t-2}$, for a fixed $\lambda > 0$. This means that we fit T vectors $\hat{\mathbf{a}}_1, \dots, \hat{\mathbf{a}}_T \in \mathcal{S}^D$ to the observed data $\mathbf{x}_1, \dots, \mathbf{x}_T$, taking into account a given level of smoothness controlled by the penalty term. When λ goes to infinity we get $\Delta^2 \mathbf{a}_t = 0$ which can be shown to be equal to $\mathbf{a}_t = \mathbf{a} \oplus (t \odot \mathbf{b})$, for all t , for some \mathbf{a} and \mathbf{b} in \mathcal{S}^D ; i.e. \mathbf{a}_t is a linear function in the compositional sense. For $\lambda < \infty$ we will see that we usually get piecewise linear trends in the compositional sense.

Remark 3.1. Problem (3.1) can be extended to fit higher order polynomial trends by defining for all t firstly $\Delta^1 \mathbf{x}_t := \mathbf{x}_t \ominus \mathbf{x}_{t-1}$ resp. $\Delta^2 \mathbf{x}_t := \mathbf{x}_t \ominus 2\mathbf{x}_{t-1} \oplus \mathbf{x}_{t-2}$ and then higher order k -th finite differences incrementally by $\Delta^k \mathbf{x}_t := \Delta(\Delta^{k-1} \mathbf{x}_t)$.

To solve Problem (3.1) we will use isometric logratios. As mentioned in the last section, the mapping $\text{ilr}_{\mathbf{V}}$ is, for any fixed basis \mathbf{V} , an isometry and therefore the terms $\|\mathbf{x}_t \ominus \mathbf{a}_t\|_A^2$ resp. $\|\Delta^2 \mathbf{a}_t\|_A$ translate for any t into $\|\text{ilr}(\mathbf{x}_t) - \text{ilr}(\mathbf{a}_t)\|_E^2$ resp. $\|\Delta^2 \text{ilr}(\mathbf{a}_t)\|_E$, where in the latter Δ^2 is defined in the same way as before but for the usual addition and multiplication in \mathbb{R}^{D-1} .

Therefore we get that it suffices to solve the optimisation problem

$$(\hat{\mathbf{u}}_1, \dots, \hat{\mathbf{u}}_T)' := \arg \min_{\mathbf{u}_t \in \mathbb{R}^{D-1}} \frac{1}{2} \sum_{t=1}^T \|\text{ilr}(\mathbf{x}_t) - \mathbf{u}_t\|_E^2 + \frac{\lambda}{2} \sum_{t=3}^T \|\Delta^2 \mathbf{u}_t\|_E. \tag{3.2}$$

It is easy to see that the latter is strictly convex and therefore has a unique solution $(\hat{\mathbf{u}}_1, \dots, \hat{\mathbf{u}}_T)$. By using the inverse of $\text{ilr}_{\mathbf{V}}$, we can recover the solution to (3.1) by defining $\hat{\mathbf{a}}_t := \text{ilr}_{\mathbf{V}}^{-1}(\hat{\mathbf{u}}_t)$ for every t . This also shows that the solution to (3.1) is unique and not dependent on the choice of \mathbf{V} as any solution to (3.1) transformed by $\text{ilr}_{\mathbf{V}}$ is necessarily the unique one to (3.2). Therefore, the fit in ilr coordinates for any matrix \mathbf{V} , as well as in any special coordinate system like balances or symmetric pivot coordinates is immediately available [5].

Note that in problem (3.2) we use the l_2 norm as a penalty on the second differences. This corresponds to a group lasso penalty, see [16], on the latter. The interpretation of this is that we look for times t where the trends of \mathbf{x}_t change at once for all components together as the penalty automatically sets $\Delta^2 \mathbf{a}_t$ for certain t to zero.

Remark 3.2. It is also possible to extend this approach to compositional data with an index t_l , for $l = 1, \dots, T$, marking rather a position in space than time. In such a case the spacing of t_l is not equidistant. However, a generalisation is straightforward by taking the penalty terms $\left\| \frac{\Delta \mathbf{a}_{t_l}}{t_l - t_{l-1}} \ominus \frac{\Delta \mathbf{a}_{t_{l-1}}}{t_{l-1} - t_{l-2}} \right\|_A$ instead; compare with the univariate usual trend filtering extension to non-equidistant points discussed in [9].

We also want to remark that the estimator defined in (3.1) changes accordingly under rescaling of each coordinate. That is, assume that each coordinate of the time series $(x_i)_t$ is rescaled by α_i , meaning that we look at $\alpha_i(x_i)_t$. Then if $\hat{\mathbf{a}}_t$ is the solution to (3.1) for the data \mathbf{x}_t , $\hat{\mathbf{a}}_t \oplus \alpha$ is the solution to (3.1) for the data $\mathbf{x}_t \oplus \alpha$. The proof is trivial, as $\Delta^2 \alpha = 0$ and because $\mathbf{x}_t \ominus \hat{\mathbf{a}}_t = (\mathbf{x}_t \oplus \alpha) \ominus (\hat{\mathbf{a}}_t \oplus \alpha)$. Therefore, rescaling the data rescales the estimator accordingly. This is interesting when looking at log-ratios and other compositional tools. As, for example, when analysing the trend in log-ratio coordinates we get that after rescaling, the latter is only shifted by a positive constant.

Finally, when we want to go back to a non-compositional view, we simply multiply the estimator $\hat{\mathbf{a}}_t$ with P_t . Choices of P_t are case dependent. If a multivariate smooth to the original data is also of interest then a smoothed version of $\sum_{i=1}^p (\mathbf{x}_t)_i$ can make sense, see Section 5.

4. Computational considerations

4.1. The ADMM approach

As problem (3.2) has a very special structure, we use an ADMM (Alternating Direction Method of Multipliers) approach. The latter is very easy to implement and has been, with a slight modification, used for the univariate case [9], as well as for multivariate piecewise constant trend filtering before, see [14]. Naturally other, approaches could be used here, such as the Primal-Dual Interior-Point Method, as briefly mentioned in [7]. For a more thorough introduction to ADMM we refer to [3].

For given functions f, g , matrices \mathbf{A}, \mathbf{B} and a vector \mathbf{c} , the augmented Lagrangian of the optimisation problem

$$\min_{\mathbf{x}, \mathbf{y}} f(\mathbf{x}) + g(\mathbf{y}) \quad \text{s.t.} \quad \mathbf{Ax} + \mathbf{By} = \mathbf{c} \tag{4.1}$$

is defined as:

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) := f(\mathbf{x}) + g(\mathbf{y}) + \boldsymbol{\theta}'(\mathbf{Ax} + \mathbf{By} - \mathbf{c}) + \frac{\rho}{2} \|\mathbf{Ax} + \mathbf{By} - \mathbf{c}\|_E^2,$$

where $\boldsymbol{\theta}$ denotes the dual variable, and $\rho > 0$ is fixed. Solving (4.1) is then done by the following iterative scheme,

$$\mathbf{x} \leftarrow \arg \min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}), \quad \mathbf{y} \leftarrow \arg \min_{\mathbf{y}} \mathcal{L}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}), \quad \boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \rho(\mathbf{Ax} + \mathbf{By} - \mathbf{c}),$$

with given starting vectors for \mathbf{x}, \mathbf{y} and $\boldsymbol{\theta}$.

To be able to use ADMM for (3.2) we firstly need to reformulate the problem. For this, denote by \mathcal{I}_D the unit matrix of size D and by \mathcal{O} a matrix of only zeros. We define the second difference matrix as

$$\mathbf{D}^2 := \begin{bmatrix} \mathcal{I}_D & -2\mathcal{I}_D & \mathcal{I}_D & \mathcal{O} & \dots & \mathcal{O} \\ \mathcal{O} & \mathcal{I}_D & -2\mathcal{I}_D & \mathcal{I}_D & \mathcal{O} & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \mathcal{O} \\ \mathcal{O} & \mathcal{O} & \mathcal{O} & \mathcal{I}_D & -2\mathcal{I}_D & \mathcal{I}_D \end{bmatrix} \in \mathbb{R}^{(D-1)(T-2) \times (D-1)T}.$$

It is easy to see that for the stacked vector of all \mathbf{u}_t , i.e. $\mathbf{u} := (\mathbf{u}_1, \dots, \mathbf{u}_T)'$, we have

$$\mathbf{D}^2 \mathbf{u} = (\mathbf{u}_3 - 2\mathbf{u}_2 + \mathbf{u}_2, \dots, \mathbf{u}_T - 2\mathbf{u}_{T-1} + \mathbf{u}_{T-2})'$$

and therefore problem (3.2) can be written as

$$\begin{aligned} \arg \min_{\mathbf{u}_t \in \mathbb{R}^{D-1}} & \frac{1}{2} \sum_{t=1}^T \|\text{ilr}_V(\mathbf{x}_t) - \mathbf{u}_t\|_E^2 + \frac{\lambda}{2} \sum_{t=3}^T \|\boldsymbol{\eta}_{t-2}\|_E \\ \text{s.t. } & \mathbf{D}^2 \mathbf{u} = \boldsymbol{\eta} \in \mathbb{R}^{(D-1)(T-2)}, \end{aligned}$$

where $\boldsymbol{\eta}_l$ denotes the subvector $(\eta_{(l-1)(D-1)+1}, \dots, \eta_{l(D-1)})'$ of $\boldsymbol{\eta}$. In this form we can use ADMM as explained before, where f is simply the first sum and g the second one.

As outlined above, denote by $\boldsymbol{\theta} \in \mathbb{R}^{(p-1)(T-k)}$ the dual variable. The augmented Lagrangian is then given by

$$\begin{aligned} \mathcal{L}(\mathbf{u}, \boldsymbol{\eta}, \boldsymbol{\theta}) := & \frac{1}{2} \sum_{t=1}^T \|\text{ilr}_{\mathbf{V}}(\mathbf{x}_t) - \mathbf{u}_t\|_E^2 + \frac{\lambda}{2} \sum_{t=3}^T \|\boldsymbol{\eta}_{t-2}\|_E \\ & + \boldsymbol{\theta}'(\mathbf{D}^2\mathbf{u} - \boldsymbol{\eta}) + \frac{\rho}{2} \|\mathbf{D}^2\mathbf{u} - \boldsymbol{\eta}\|_E^2. \end{aligned}$$

From the latter we can easily deduct the ADMM updates by optimising in each variable at once holding the others fixed. Optimizing in \mathbf{u} is simple and can be done by setting the derivative to zero. Optimizing in $\boldsymbol{\eta}$ alone is a group Lasso problem with non-overlapping groups. Denoting the proximal operator of the group lasso with non-overlapping groups as $\mathcal{P}_{\frac{\lambda}{\rho}}$, see [8], and writting from now on $\bar{\boldsymbol{\theta}} := \frac{\boldsymbol{\theta}}{\rho}$, we get the following updates:

$$\mathbf{u} \leftarrow (\mathcal{I}_{(D-1)T} + \rho \mathbf{D}^{2'} \mathbf{D}^2)^{-1} (\text{ilr}_{\mathbf{V}}(\mathbf{x}_t) - \rho \mathbf{D}^{2'}(\bar{\boldsymbol{\theta}} - \boldsymbol{\eta})) \tag{4.2}$$

$$\boldsymbol{\eta} \leftarrow \mathcal{P}_{\frac{\lambda}{\rho}}(\mathbf{D}^2\mathbf{u} + \bar{\boldsymbol{\theta}}) \tag{4.3}$$

$$\bar{\boldsymbol{\theta}} \leftarrow \bar{\boldsymbol{\theta}} + \mathbf{D}^2\mathbf{u} - \boldsymbol{\eta}. \tag{4.4}$$

As we usually like to solve problem (3.1) for a whole set of given λ 's, e.g. $\lambda_1, \dots, \lambda_L$, it seems sensible to use as starting vectors $(\mathbf{u}, \boldsymbol{\eta}, \bar{\boldsymbol{\theta}})$ for the above iteration a warm start scheme, meaning that when solving problem (3.1) for λ_i we use the solutions obtained by (4.2)–(4.4) belonging to λ_{i-1} as a start for (4.2)–(4.4) belonging to λ_i .

4.2. Cross Validation (CV)

The optimal λ from a set of $\{\lambda_1, \dots, \lambda_L\}$ can be chosen by K-fold CV. More precisely, denote with $\mathcal{F}_1, \dots, \mathcal{F}_K$ the folds, i.e. a non-overlapping partition of $1, \dots, T$ into K sets. In the case of trend filtering these should be chosen in an interleaved way, which means that for any time point t belonging to a certain fold \mathcal{F}_k , the elements of the neighbouring time points belong to other folds.

For each pair $(\lambda_i, \mathcal{F}_k)$ we calculate $CV(\lambda_i, \mathcal{F}_k) := \sum_{t \in \mathcal{F}_k} \left\| \text{ilr}_{\mathbf{V}}(\mathbf{x}_t) - \hat{\mathbf{u}}_t^{-\mathcal{F}_k} \right\|^2$, where $\hat{\mathbf{u}}_t^{-\mathcal{F}_k}$ denotes the prediction, at $t \in \mathcal{F}_k$, of the estimator calculated on the subset of observations with indices $\{1, \dots, T\} \setminus \mathcal{F}_k$.

To chose λ , one can take the argmin in $\{\lambda_1, \dots, \lambda_L\}$ of

$$CV(\lambda) := \frac{1}{T} \sum_{k=1}^K CV(\lambda, \mathcal{F}_k).$$

Denoting with λ^* the argmin of the latter, another popular choice is to use the one standard error rule to choose the optimal λ , as often done for usual univariate trend filtering. Thus we choose the optimal λ as the maximal λ satisfying $CV(\lambda) \leq CV(\lambda^*) + \sigma(\lambda^*)$, where $\sigma(\lambda)$ is the standard deviation of the data points $CV(\lambda, \mathcal{F}_1), \dots, CV(\lambda, \mathcal{F}_K)$, see [4].

5. Coronavirus data

To illustrate the utility of the method presented above we will look at the number of COVID-19 infections in 9 different countries in the time period from 2020-03-01 until 2020-07-31. This data set is publicly available at <https://ourworldindata.org/coronavirus-testing>.

In Figure 1 we show the absolute number of reported infections per 100000 inhabitants. As for a very few days the reported number of positive cases are zero, we used time series imputation from the R package forecast [6] to replace the zeros by positive numbers. Most countries show a very similar pattern and reached their maximum number of cases around April; except for Sweden which reached it in June/July. We can see that some countries like Sweden, Spain and Belgium had much higher peaks. However, we can also see that the periods of high values around the peaks also differs a lot among the countries.

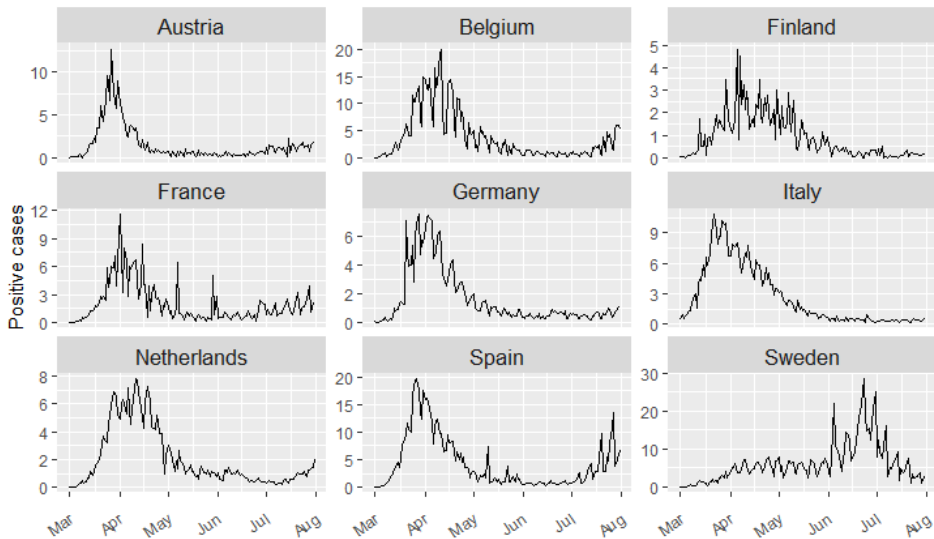


Figure 1. Number of reported COVID-19 infections per 100000 inhabitants from 2020-03-01 to 2020-07-31 for different countries.

To gain more insight into how one country performed in comparison to the whole group of countries we use the method described above. We estimate the multivariate trend filtering fits $\hat{\mathbf{a}}_t$ for \mathbf{x}_t , as described above, for λ chosen by $K = 10$

fold cross validation and the one standard error rule, see Subsection 4.2, with $\lambda \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30\}$. In Figure 2, we plot the first pivot coordinate $\sqrt{\frac{8}{9}} \log \left(\frac{(\mathbf{x}_t)_i}{\sqrt[8]{\prod_{j \neq i} (\mathbf{x}_t)_j}} \right)$, where i is the index corresponding to the i -th country in \mathbf{x}_t , and the estimated trend. Pivot coordinates can be used to compare the fit (or number of cases) of one country to the geometric mean of the fit (or number of cases) of the rest of the group, containing all relative information of a specific part to the remaining parts in the considered composition [5].

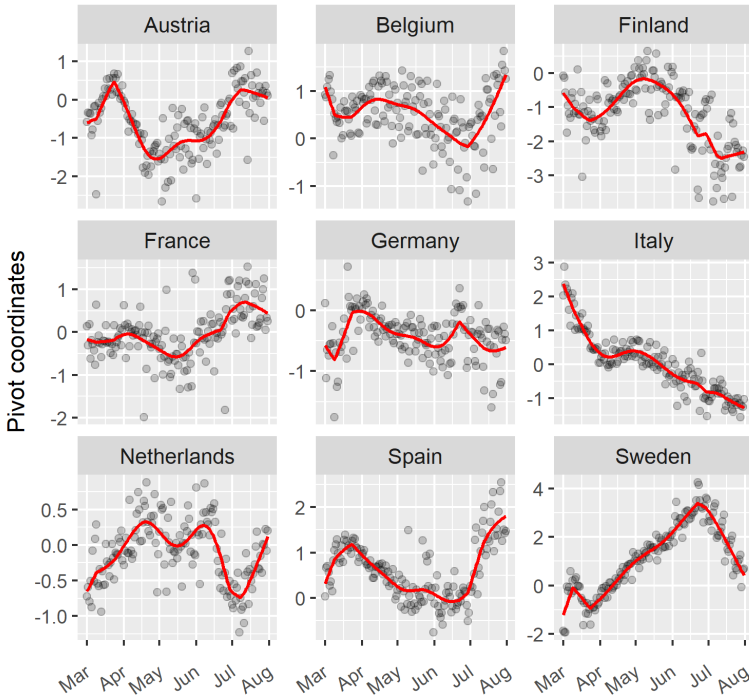


Figure 2. Positive COVID-19 cases in first pivot coordinate per country. The black dots are the measured number of cases in the first pivot coordinate for each country. Equally, the red lines are its compositional trend filtering fits for $\lambda = 10$.

It can be seen in Figure 2 that, compared to the whole group, Germany and Finland have had a comparatively low number of cases, as the values are mainly negative. Finland experienced a trend change in the middle/end of March with rising numbers compared to the geometric mean of the rest of the group, whereas Germany experienced a downward one. It is interesting to see that many countries have had a change in trend since mid/end June - e.g. Belgium, Germany and Sweden. For instance, since the absolute numbers in Spain and Belgium have been rising quickly in July, more than in other countries, also a relative increase com-

pared to the other countries is visible. The contrary behaviour can be seen for Sweden, with a decay in relative number in the beginning of July. It is also surprising to see that Italy, which had the highest infection numbers at the beginning all over Europe, has constantly been improving compared to the other countries, and is by the end of June doing better than average. The Netherlands have at the end of July had average numbers however, its trend might have been alarming.

Furthermore, for a fixed country it might also be interesting to see how the trend of positive cases behaves compared to one other country. For this we display in Figure 3 the log-ratios of Austria and each country present in the composition.

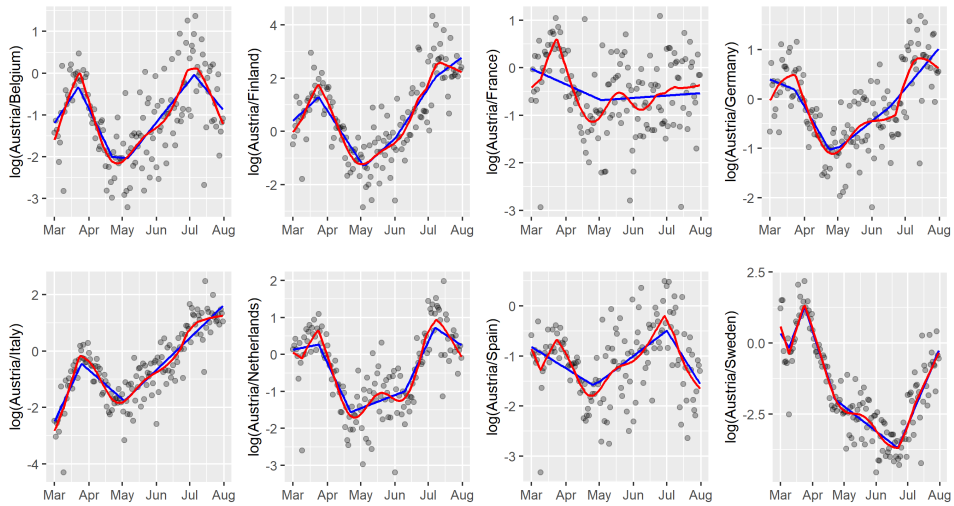


Figure 3. Log-ratios of Austria and all other countries of the composition. The black points are the observations, in red we display the compositional trends and in blue the regular univariate trend-filtering estimates. λ was chosen in both cases by cross validation with the one standard error rule.

We plot the compositional trend filtering fit described by the method above, which was also used for Figure 2, in log-ratios in red. Additionally, we show the univariate non-compositional trend filtering fits to each log-ratio pair in blue with λ obtained by cross validation with the one standard error rule as implemented in the R package [2]. We can see from the log-ratio between Austria and Germany that since July Austria has increasingly more cases than Germany. This upward trend has already started at the beginning of May when Austria still had less positive cases. At the same time the trend for the log-ratio between Austria and Italy started to change. This means that the positive cases per 100000 inhabitants in Austria is growing at a faster rate compared to each, Germany and Italy, since the beginning of July. Something similar holds for the pair Austria and Finland. At the end of July we can see that the cases in Austria compared to Italy or Finland

show a similar trend and numbers – around two. This is surprising as Italy notably started off with the highest infection numbers. At last, since July, Belgium and Spain seem do be doing worse and worse compared to Austria, indicated by a very steep downward trend. The non-compositional univariate trend filtering estimates (in blue) show a slightly different picture. The compositional approach differs sometimes vastly, e.g. for the log-ratio Austria-France, and seems to follow the data better where the non-compositional approach oversmooths, e.g. the log-ratio Austria-Spain in March, end of April and end of June. We also note that the compositional nature leads to trends which on the one hand are at times very smooth, and on the other, also display rapid changes.

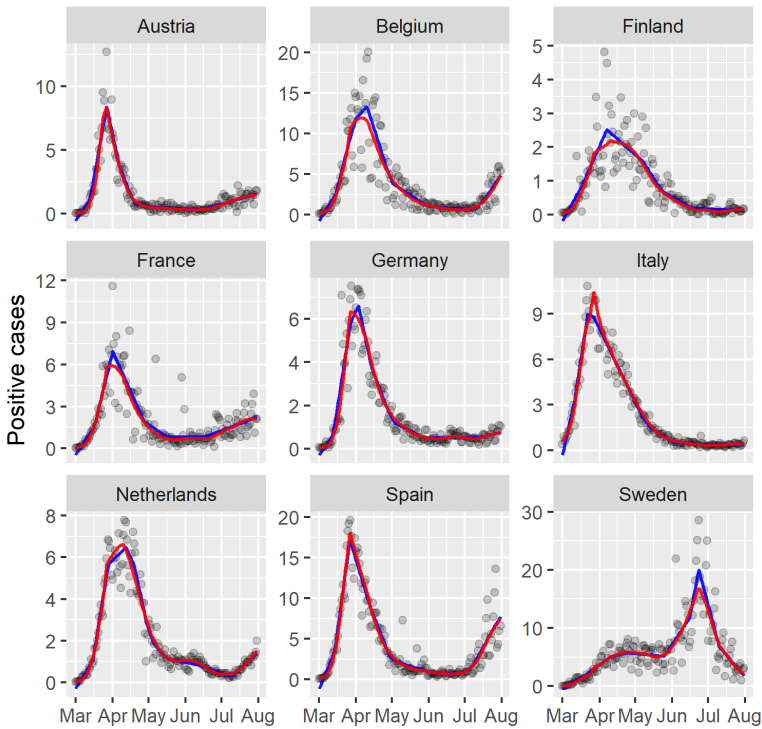


Figure 4. The black dots show for every country the recorded number of positive cases per 100000 inhabitants. The red lines display the estimator $P_i \hat{\mathbf{a}}_t$. The blue lines display the trends estimated by non-compositional trend filtering.

The fit $(\hat{\mathbf{u}}_1, \dots, \hat{\mathbf{u}}_T)'$ can also be back-transformed to $(\hat{\mathbf{a}}_1, \dots, \hat{\mathbf{a}}_T)'$ into the simplex, and the elements per time point sum up to 1. If we want to present those back-transformed fits together with the absolute numbers of infections per 100000 inhabitants, we need to multiply with the total P_t per time point. However, as we want to keep the smoothness of the fits, we will multiply with a smoothed total, as

described at the end of Section 3. To do this we sum up at each time point for all countries the number of positive cases, log-transform the latter and divide it by the absolute maximum, fit a smoother – in our case a univariate trend filter from the R package `genlasso` [2] with $\lambda = 1$ – and transform it back by multiplying it first with the absolute maximum of the log-transform and then taking its exponential. In Figure 4 we display the fit $P_t \hat{\mathbf{a}}_t$ to the original points in red. In blue we show the univariate non-compositional trend filtering fits for $\lambda = 1$, where the data were first divided by the maximum, then the trend filtering fit was obtained, and lastly, the latter was again multiplied by the maximum. Dividing the data by the absolute maximum before fitting puts the λ on a similar footing for comparison with $P_t \hat{\mathbf{a}}_t$. Note that the estimator $P_t \hat{\mathbf{a}}_t$ always sums up to the total P_t , as we are taking a compositional approach. It is interesting to see that for most countries the highest number of cases had been reached around mid/end of March with a change in trend since then. Also note that the fitted trend to Italy in the first half of April seems to be larger than one would expect from the data in Euclidean space. Examining the data at this time more closely we can see that the number of positive cases actually is very high for two consecutive days, before suddenly dropping and so the compositional fit reflects this better than the non-compositional one. Lastly, the non-compositional approach gives a negative estimate at the beginning of the measurements and shows a slightly different behaviour at the peaks.

6. Summary and conclusions

In this paper we presented a new method for fitting piecewise compositional linear trends to compositional time series. The method we proposed is a direct extension of univariate trend filtering to the compositional setting, which is multivariate by definition. This was done by reformulating the optimization problem in the appropriate Aitchison geometry on the simplex. We showed how to derive the problem in the usual Euclidean geometry, expressed in *ilr* coordinates, and that the solution does not depend on the specific choice of *ilr* coordinates. We proceeded by describing how to efficiently solve the problem through an ADMM algorithm.

To show the usefulness of our method, we looked at the number of COVID-19 cases in different European countries in the period from March to July 2020. Namely, once the compositional trend was fitted, we explored the trends in Pivot coordinates and log-ratio representations. This gave insights into how the COVID-19 infections in some countries behaved compared to the compositional mean of other countries, during the said time period.

The fitted trends have been back-transformed to the simplex. The results have to sum up to 1 per time point, and the values cannot be negative – which is often a desirable property. After multiplication by a smoothed total for every country, the total infection numbers can be compared to the smooth line, which still represents relative information to all other countries, and deviations might indicate interesting phenomena (higher variability, etc.).

Additionally to the compositional time series case presented in this paper, one

could look at various extensions. For example, instead of looking at compositional data with a time stamp, one could also look at compositional data with a geographic stamp. That means that we would look at a random variable $\mathbf{x}_{(u,s)} \in \mathcal{S}^{D-1}$ with (u, s) being a geographic location. Such a case might be interesting for geochemical exploration where at certain geographic points we measure the concentration of elements. For the latter a shift in concentrations might correspond to interesting areas, thus leading to piecewise constant trend filtering in the compositional sense. Another extension might consider robustification of the proposed method. Outliers are quite common in any statistical setting and compositional data are just as much affected by such. Therefore, investigating a compositionally robust definition of trend filtering might be interesting.

References

- [1] J. AITCHISON: *The statistical analysis of compositional data*, Journal of the Royal Statistical Society: Series B (Methodological) 44.2 (1982), pp. 139–160, DOI: <https://doi.org/10.1111/j.2517-6161.1982.tb01195.x>.
- [2] T. B. ARNOLD, R. J. TIBSHIRANI: *Package ‘genlasso’*, Statistics 39.3 (2020), pp. 1335–1371.
- [3] S. BOYD, N. PARIKH, E. CHU: *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Now Publishers Inc, 2011, DOI: <https://doi.org/10.1561/9781601984616>.
- [4] L. BREIMAN, J. FRIEDMAN, C. J. STONE, R. A. OLSHEN: *Classification and Regression Trees*, CRC Press, 1984.
- [5] P. FILZMOSER, K. HRON, M. TEMPL: *Applied Compositional Data Analysis*, Switzerland: Springer Nature (2018), DOI: <https://doi.org/10.1007/978-3-319-96422-5>.
- [6] R. HYNDMAN, G. ATHANASOPOULOS, C. BERGMEIR, G. CACERES, L. CHHAY, M. O’HARA-WILD, F. PETROPOULOS, S. RAZBASH, E. WANG, F. YASMEEN: *forecast: Forecasting functions for time series and linear models*, R package version 8.13, 2020, URL: <https://pkg.robjhyndman.com/forecast/>.
- [7] S. J. KIM, K. KOH, S. BOYD, D. GORINEVSKY: l_1 trend filtering, SIAM Review 51.2 (2009), pp. 339–360, DOI: <https://doi.org/10.1137/070690274>.
- [8] N. PARIKH, S. BOYD: *Proximal algorithms*, Foundations and trends in optimization 1.3 (2014), pp. 127–239, DOI: <https://doi.org/10.1561/2400000003>.
- [9] A. RAMDAS, R. J. TIBSHIRANI: *Fast and flexible ADMM algorithms for trend filtering*, Journal of Computational and Graphical Statistics 25.3 (2016), pp. 839–858.
- [10] V. SADHANALA, R. J. TIBSHIRANI: *Additive models with trend filtering*, arXiv preprint arXiv:1702.05037 (2017).
- [11] R. J. TIBSHIRANI: *Adaptive piecewise polynomial estimation via trend filtering*, The Annals of Statistics 42.1 (2014), pp. 285–323, DOI: <https://doi.org/10.1214/13-AOS1189>.
- [12] R. TIBSHIRANI: *Regression shrinkage and selection via the lasso*, Journal of the Royal Statistical Society: Series B (Methodological) 58.1 (1996), pp. 267–288, DOI: <https://doi.org/10.1111/j.2517-6161.1996.tb02080.x>.

-
- [13] R. VARMA, H. LEE, J. KOVAČEVIĆ, Y. CHI: *Vector-valued graph trend filtering with non-convex penalties*, IEEE Transactions on Signal and Information Processing over Networks 6 (2019), pp. 48–62,
DOI: <https://doi.org/10.1109/TSIPN.2019.2957717>.
- [14] B. WAHLBERG, S. BOYD, M. ANNERGREN, Y. WANG: *An ADMM algorithm for a class of total variation regularized estimation problems*, IFAC Proceedings Volumes 45.16 (2012), pp. 83–88,
DOI: <https://doi.org/10.3182/20120711-3-BE-2027.00310>.
- [15] Y.-X. WANG, J. SHARPBACK, A. J. SMOLA, R. J. TIBSHIRANI: *Trend filtering on graphs*, The Journal of Machine Learning Research 17.1 (2016), pp. 3651–3691.
- [16] M. YUAN, Y. LIN: *Model selection and estimation in regression with grouped variables*, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 68.1 (2006), pp. 49–67,
DOI: <https://doi.org/10.1111/j.1467-9868.2005.00532.x>.

Exploiting the structure of communication in actor systems

Krisztián Schäffer^a, Csaba István Sidló^b

^aMonotic Mo. Kft., Budapest, Hungary
krisztian.schaffer@monotic.com

^bInstitute for Computer Science and Control (SZTAKI), Budapest, Hungary
sidlo@sztaki.hu

Submitted: December 23, 2020

Accepted: March 8, 2021

Published online: May 18, 2021

Abstract

We propose a novel algorithm for minimizing communication costs of multi-threaded and distributed actor systems, to gain performance advantage by dynamically adapting to the structure of actor communication. We provide an implementation in Circo, an open source actor system, and show promising experimental results.

Keywords: Actor model, concurrent systems

1. Introduction and related work

Actor-based concurrency models [1] have been used for decades for scalable distributed applications [11]. Actors – the primitives of concurrency – encapsulate their state, communicate through asynchronous messaging and form arbitrary topological relations.

Various frameworks and languages permit actor programming, including Akka [15], CAF [7] and Pony [10]. Applications include banking and telecom transaction processing, complex event stream processing and large-scale analytical pipelines. The concurrency model of microservice architectures [8] corresponds with the actor model, and actor frameworks can be applied directly in cloud environments (e.g. Orleans [4]). Driven by the popularity of cloud and Internet of Things (IoT)

solutions and the stagnating performance of single CPU cores, the last few years has seen an increased interest in actor systems. We believe that actor systems also have a great potential for artificial intelligence, by providing an efficient tool to incorporate sparsity into deep learning.

1.1. Why actors?

Programs built using other programming models – especially the synchronous ones – may be easier to reason about, but the actor model allows unlimited scaling and a variety of performance optimizations thanks to a few key properties:

1. *No shared state*: An actor can access only its own state directly, and everything else must be done through messaging. Shared state is an abstraction famous for introducing hard to find bugs called data races in concurrent programs. Actor programming does not expose the programmer to the risks of shared memory, leaving shared memory to automatic performance optimizations.
2. *No global synchronization mechanism included*: Synchronization must be implemented on the actor level, using the fact that message processing of a single actor is serializable.
3. *Location transparency*: The act of sending a message does not depend on the location of the target actor – sending messages within a machine is the same as between machines.

Global synchronization performance degrades as the physical diameter of the system grows, because information cannot travel faster than light. Similarly, providing the illusion of synchronous shared state – which does not exist in reality – is only possible with introducing a latency proportional to the diameter of the subsystem containing the state. Not having these features allows the actor model to scale arbitrarily without performance loss. The third property, location transparency, allows the execution environment to optimize actor placement and message passing during run-time without actors noticing it.

1.2. Communication complexity

Communication is a common performance bottleneck of distributed systems, even on single-node multi-core systems, where shared-memory communication between cores works well, but brings in significant latency.

Communication is layered in modern hardware: network is slower than shared-memory which is slower than in-thread (cached) data passing. This layeredness of technology is a result of physical and technological constraints, namely the speed of light, maximal density of hardware elements, and manufacturing costs. It is reasonable to think that these constraints and the technology layers will not disappear soon. Even if the layers merge, messaging latency remains dependent on physical distance, because information cannot travel faster than light. Communi-

cation therefore will remain a performance limiting factor of distributed systems for long.

Large scale data processing systems apply data locality to minimize the cost of communication – bringing computation to the data [6]. Disk-based data locality was a key success factor of MapReduce, but as network technology outpaced local storage speed, memory locality became the primary goal. Data processing frameworks are now often aware of locality, with regard to NUMA (Non-Uniform Memory Access) patterns [13]. Actor systems are also employing techniques to deal with locality in non-uniform shared memory: a locality-guided scheduler for CAF was published in [14], and locality-aware work stealing scheduler methods was studied in [2].

The main goal of this paper is to provide a general method to reduce communication overhead in distributed systems. We formulate a solution in the context of the actor model: the decentralized “infoton optimization” algorithm is presented, which explores and exploits the structure of communication to minimize communication cost by co-locating actors during run-time. The computational cost of this algorithm is proportional to the number of actor messages.

2. Infoton optimization

To reduce communication costs during execution of an actor system, frequently communicating actors are to be moved to a common, or at least to nearby locations – e.g. to the same NUMA location, computer or data center.

Infoton optimization is a physics-inspired model, essentially a decentralized, scalable version of force-directed graph drawing [12] – a physical system of bodies with cohesive forces, where the energy of the system is to be minimized. Infoton optimization maps intensity of actor communication to forces of the physical system and approximates the behavior of the system in a way that needs no central coordination.

Actors and schedulers (threads executing actor code) are mapped to 3D Euclidean space: The main idea is that distance approximates communication cost, and actors move towards their communication partners to minimize communication cost.

Euclidean space is chosen on purpose as the model of the physical universe, where communication cost often depends on physical distance – even multicore CPUs evolve to be 3D structures [3]. However, as network and other communication costs don’t always match the strict Euclidean properties, other spaces might also be investigated.

Schedulers are embedded in a way that their distance represents communication overhead, either by static positioning, or by using network coordinates. Actors move in the space during optimization and are continuously migrated to the nearest scheduler.

2.1. Assumptions

Infoton optimization assumes the following properties of the actor system:

1. *Actors are significantly more numerous than threads.*
2. *Actor communication is structured:* Only a small, slowly changing portion of possible actor connections is used.
3. *Actors can be migrated:* Computational load with actors can be moved between schedulers, affecting future communication cost.

2.2. What is an infoton

The infoton is the quantum of actor forces, a force-carrying particle – like photons in physics – that:

1. Is coming from a *source location*.
2. Carries a positive scalar value called *energy*.
3. Has a *sign*.

2.3. Infoton action

When the infoton acts on an actor, it either pulls or pushes the actor toward/away from the source location of the infoton. The direction of the actor movement depends on the *sign* (positive pulls), while the distance is proportional to the *energy* of the infoton.

Actors have no inertia in this model, they only move when infotons act on them. This way inactive actors introduce no computational overhead. The physical analogy is that actors move in thick fluid.

2.4. The first force

We define two major forces of infoton optimization. The first force of infoton optimization brings communicating actors toward one another.

1. An infoton is attached to every message passed between actors, holding the *position of the source actor* and a unit of *energy* with positive sign.
2. When the message arrives at its destination actor, the infoton attached to it acts on that actor, *pulling it towards the source of the message*.

2.5. The second force

Another force spreads actors in the segment of the space near schedulers, avoiding all concentrating around a single point:

1. When a message arrives, the scheduler that executes the target actor creates a new “scheduler infoton”, with itself as source.
2. Scheduler infotons either pull or push actors toward or away from the scheduler, depending on the load of the scheduler.

3. Implementation details

We have added an experimental implementation to the Circo [9] actor system (where the main author and maintainer is the main author of this paper). Circo is written in Julia [5], a dynamically typed, garbage collected general-purpose language designed for numerical computing.

Unlike most contemporary actor systems, Circo implements multi-threading by running several single-threaded schedulers that communicate through shared memory and form the “host cluster”. Similarly, distribution of work between hosts is done in a separated cluster, which we call “the” Cluster, because we think that the Actors stick to schedulers by default, but can migrate between them by using the migration service.

For simplicity the current implementation of infoton optimization assumes that communication overhead between any pair of schedulers is fixed, thus schedulers are statically positioned. This, however, can be extended to be dynamically adjusted.

The algorithm can be customized with the following parameters:

1. I – A proportionality constant of actor forces, similar to the G gravitational constant in physics. It connects the energy of the acting infoton to the length of movement caused by the action. Higher values cause more intense actor movement.
2. $TARGET_DISTANCE$ – Force-directed graph drawing algorithms often use repulsive forces between every pair of nodes to avoid the concentration of nodes. The second force of infoton optimization has a similar goal, but we have found that the algorithm is more stable with a quirk that approximates a hidden repelling force acting only at low distances: When the source position of a pulling infoton is closer to the target actor than $TARGET_DISTANCE$, its effect is extinguished.
3. $SCHEDULER_TARGET_LOAD$ – We define the load of a scheduler as the total number of messages waiting to be processed. This parameter sets the load that every scheduler tries to maintain independently. Scheduler infotons emitted by a scheduler will pull actors when its load is lower and push when higher.
4. $SCHEDULER_LOAD_FORCE_STRENGTH$ – Proportionality constant of scheduler infoton energy.

Following is the Julia code that calculates the movement of an actor caused by an infoton acting on it (error handling is not shown):

```
function Circo.apply_infoton(targetactor, infoton)
    diff = infoton.sourcepos - targetactor.core.pos
    difflen = norm(diff)
    energy = infoton.energy
    if energy > 0 && difflen < conf[.TARGET_DISTANCE]
        return nothing
    end
    stepvect = diff / difflen * energy * conf[.I
```

```

targetactor.core.pos += stepvect
return nothing
end

```

The code to generate the “scheduler infoton” when delivering a message:

```

function Circo.scheduler_infoton(scheduler, actor)
    dist = norm(scheduler.pos - actor.core.pos)
    loaddiff = Float64(conf[].SCHEDULER_TARGET_LOAD - length(scheduler.msgqueue))
    if loaddiff == 0.0 # Nothing to do at target load
        return Infoton(scheduler.pos, 0.0)
    end
    energy = sign(loaddiff) *
             log(abs(loaddiff)) *
             conf[].SCHEDULER_LOAD_FORCE_STRENGTH
    return Infoton(scheduler.pos, energy)
end

```

Although Circo supports multi-threaded and distributed settings, for easier experimentation we have created a simulation environment¹ that starts several schedulers on the same thread and allows changing of optimization parameters during run-time.

4. Experiments

We have conducted experiments with two actor programs:

1. *Linked List*: Generates a linked list of actors, each storing a single scalar, then runs reduce (sum) operations on the list concurrently. When an operation finishes, immediately starts a new one, maintaining 100 concurrent operations.
2. *Search Tree*: Generates a binary search tree of actors, leafs hold 1000 scalars, inner nodes contain only a split value and addresses of two children. Fills the tree with random data and runs search operations concurrently (during and after filling, maintaining 500 concurrent searches).

In both cases a coordinator actor manages the creation of the data structure and sends the reduce/search operations to it. Results are sent back to the coordinator, so the computing graphs are cyclic: A single cycle containing every actor of the linked list, and a unique cycle for every leaf of the search tree.

We have introduced “domain knowledge” to the search tree through two simple actor behaviors, improving performance. We call these behaviors domain specific, because they reflect information about the structure of the actor system (that it is a tree). First, the coordinator actor goes back to the fixed position $(-10, 0, 0)$ every time it receives a search response. This helps stabilizing the tree layout. Second, tree nodes periodically send a negatively signed infoton to their siblings in order to

¹To reproduce the experiments, open <https://github.com/Circo-dev/ExploreInfotonOpt>.

repel each other. This is intended to open up the tree, making easier for siblings to separate.

Actors are continuously moving and occasionally changing schedulers when they get closer to another one. Messages are considered local if source and destination actors run on the same scheduler. As the direct indicator of optimization success we have measured the ratio of the number of local messages to the total number of messages.

To demonstrate that it is possible to find a single set of parameters for which infoton optimization yields good results for a wide variety of actor programs, we have run differently sized versions of the two programs with the same fixed infoton optimization parameters, selected manually:

- $I = 0.2$,
- `TARGET_DISTANCE` = 200.0,
- `SCHEDULER_TARGET_LOAD` = 13,
- `SCHEDULER_LOAD_FORCE_STRENGTH` = 0.02

Six schedulers were used, positioned at face-centers of a cube: $(-1000, 0, 0)$, $(1000, 0, 0)$, $(0, -1000, 0)$, $(0, 1000, 0)$, $(0, 0, -1000)$, $(0, 0, 1000)$.

Figures in this paper are screenshots of the Circo tool “Camera Diserta”, used to monitor and validate actor layout. Grey lines are local, orange lines are non-local connections between actors. Schedulers are drawn as blue cubes, test coordinator as a red sphere.

Figure 1 illustrates the layout of the linked list program with 200 (71%), 500 (83%), 1000 (85%), 2000 (89%), 4000 (89%) and 8000 (89%) (row major order) list item actors. Percents in parentheses are local message ratios of the last 10 seconds before taking the screenshots. (When actors are distributed randomly between six schedulers, local message ratio is $1/6$ (17%).)

Figure 2 illustrates the layout of the search tree program with 62 (51%), 126 (57%), 254 (64%), 506 (66%), 1018 (70%), 2028 (73%), 4046 (74%) and 8080 (76%) (row major order) tree node actors. Percents in parentheses are local message ratios of the last 10 seconds before taking the screenshots. Connections from leafs to the coordinator are not drawn.

Figure 3 illustrates the layout of the search tree without the domain-specific behaviors, with 62 (45%), 96 (46%), 254 (51%), 510 (50%), 1017 (51%), 2004 (56%) tree node actors. These layouts are not stable, they are slowly and continuously restructuring while maintaining high local message rate. Note that connections near the leafs have much less message traffic than near the root, so the optimization is still successful despite the high amount of non-local connections.

Infoton optimization radically improved message locality in all three experiments, reducing inter-scheduler communication by 43–87% compared to the random placement baseline. Figure 4 illustrates this by showing local message ratios achieved after optimization.

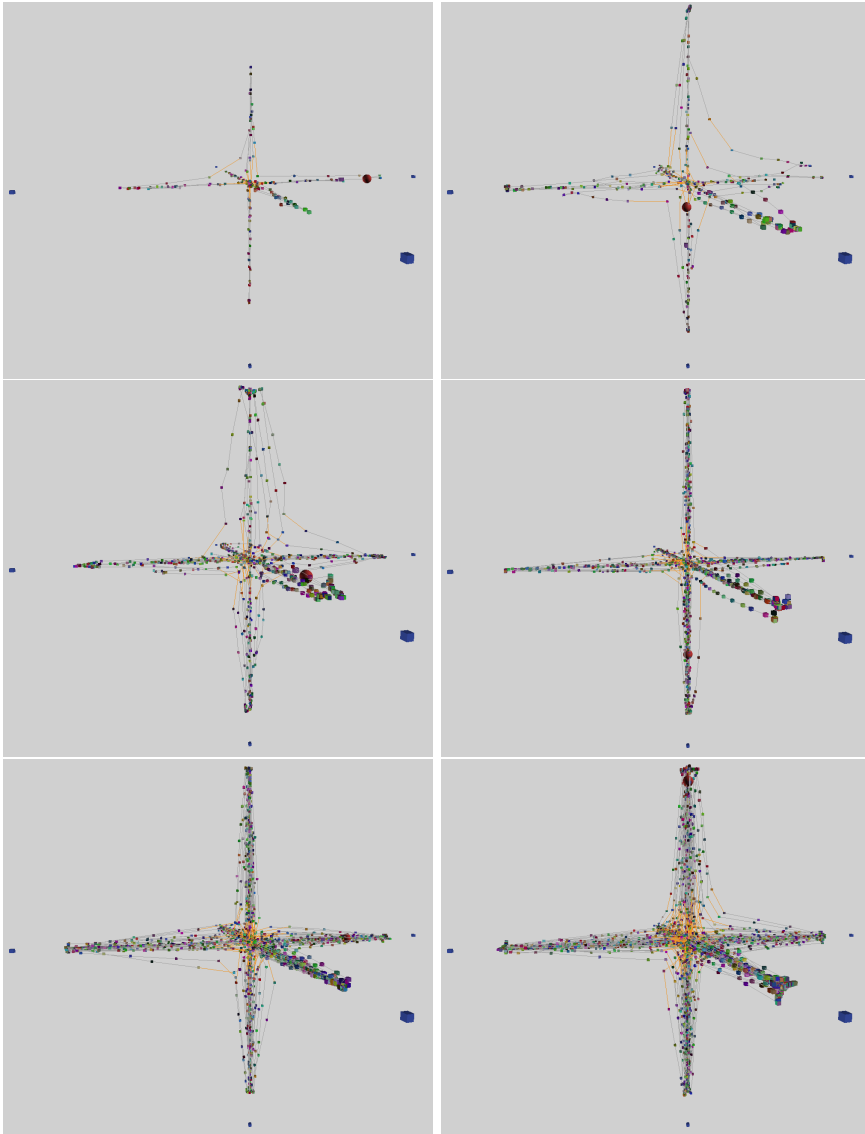


Figure 1. Optimized layouts of a linked list of 200, 500, 1000, 2000, 4000 and 8000 actors on 6 schedulers.

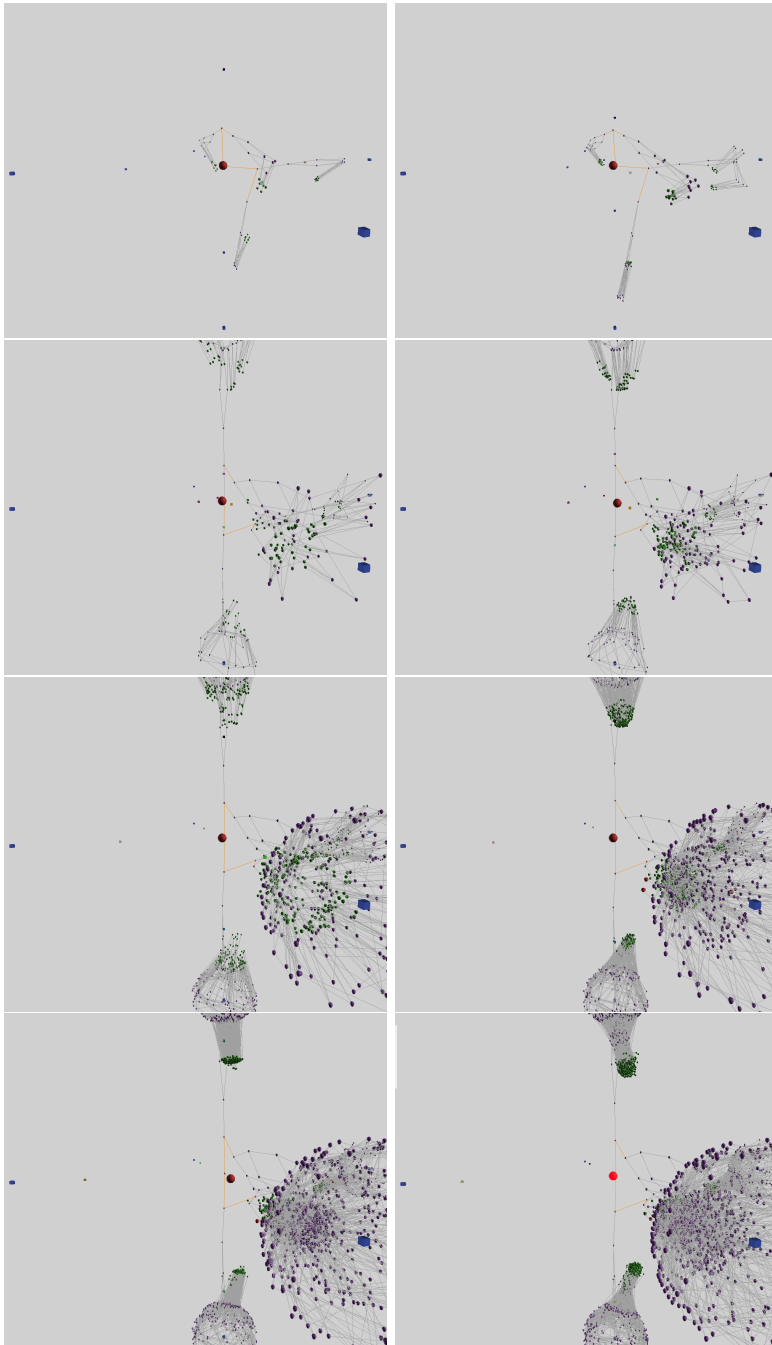


Figure 2. Optimized layouts of a binary tree built from 62, 126, 254, 506, 1018, 2028, 4046 and 8080 actors on 6 schedulers.

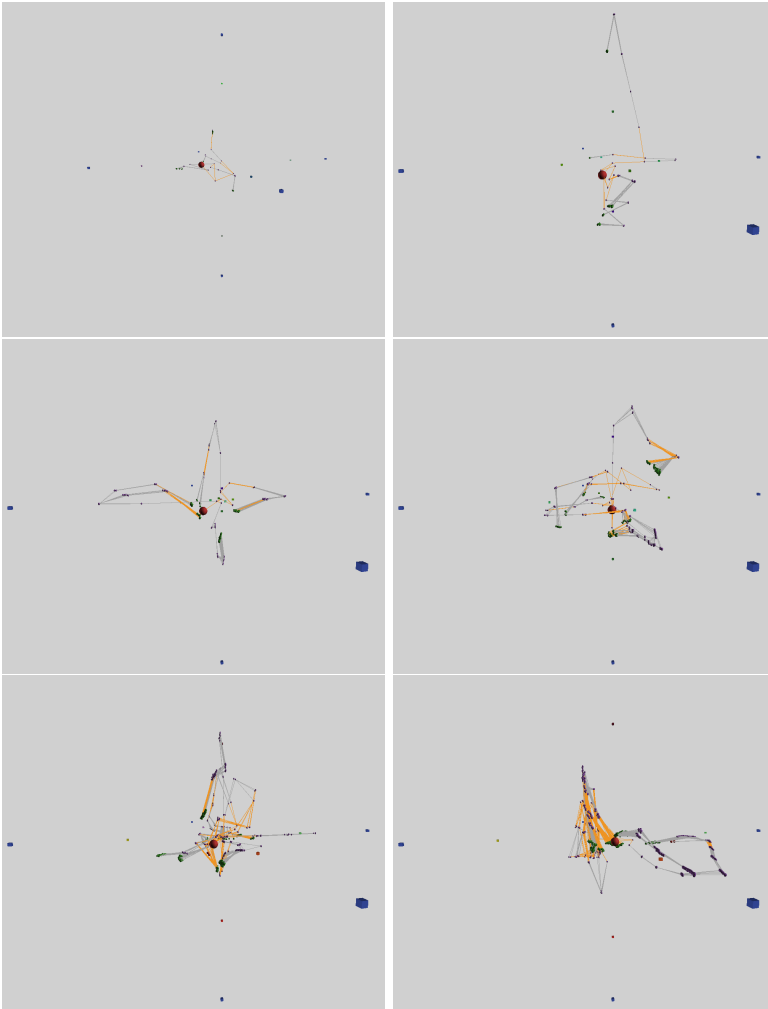


Figure 3. Optimized layouts of a binary tree without the domain-specific behaviors, built from 62, 96, 254, 510, 1017, 2004 actors on 6 schedulers.

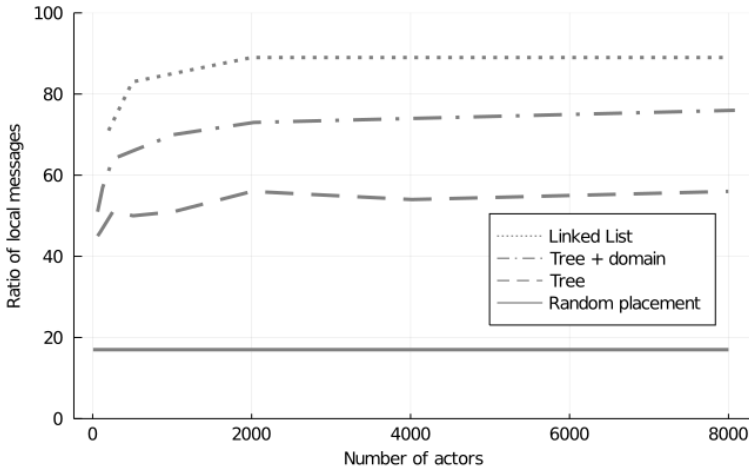


Figure 4. Local message ratios achieved at different program sizes on 6 schedulers, and random actor placement as baseline.

5. Conclusions and future work

We have introduced infoton optimization, and demonstrated in a limited scenario that it is capable of distributing computational load of actor systems while optimizing message locality. The algorithm is decentralized and its cost is proportional to the number of messages.

The algorithm has several parameters that need to be tuned manually. Manual tuning of large decentralized systems may not always be feasible, so future work should focus on meta-optimization or elimination of these parameters.

One of our examples introduces domain-specific constraints on how actors behave, which improves the efficiency of the optimization significantly. However, the optimization works well without these domain-specific behaviours too. This shows that the algorithm is easily customizable with (application-specific) domain knowledge, and for some actor programs such customization may result in significant performance gain.

In the simple version of the algorithm discussed in this paper, actors behave uniformly when infotons act on them. Introducing “mass” or “size” properties of actors to reflect the cost of migration is however a promising extension of the algorithm.

Several further aspects of infoton optimization are to be clarified and detailed as future work. For example, convergence criteria of infoton optimization and optimality of the results are studied in the context of stochastic optimization. Detailed benchmark experiments are also being performed, comparing common actor systems with Circo.

References

- [1] G. AGHA: *Actors: A Model of Concurrent Computation in Distributed Systems*, Cambridge, MA, USA: MIT Press, 1986, ISBN: 0262010925.
- [2] S. BARGHI, M. KARSTEN: *Work-Stealing, Locality-Aware Actor Scheduling*, in: 2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2018, pp. 484–494, DOI: <https://doi.org/10.1109/IPDPS.2018.00058>.
- [3] A. BEN ABDALLAH: *3D Integration Technology for Multicore Systems On-Chip*, in: Advanced Multicore Systems-On-Chip: Architecture, On-Chip Network, Design, Singapore: Springer Singapore, 2017, pp. 175–199, ISBN: 978-981-10-6092-2, DOI: https://doi.org/10.1007/978-981-10-6092-2_6.
- [4] P. A. BERNSTEIN, S. BYKOV: *Developing Cloud Services Using the Orleans Virtual Actor Model*, IEEE Internet Computing 20.5 (2016), pp. 71–75, DOI: <https://doi.org/10.1109/MIC.2016.108>.
- [5] J. BEZANSON, A. EDELMAN, S. KARPINSKI, V. B. SHAH: *Julia: A fresh approach to numerical computing*, SIAM review 59.1 (2017), pp. 65–98.
- [6] S. BONNER, I. KURESHI, J. BRENNAN, G. THEODOROPOULOS: *Chapter 14. Exploring the Evolution of Big Data Technologies*, in: Dec. 2017, ISBN: 9780128054673, DOI: <https://doi.org/10.1016/B978-0-12-805467-3.00014-4>.
- [7] D. CHAROUSSET, R. HIESGEN, T. C. SCHMIDT: *Revisiting Actor Programming in C++*, CoRR abs/1505.07368 (2015), arXiv: 1505.07368.
- [8] CHRIS RICHARDSON: *What are microservices?*, <https://microservices.io/>, [Accessed: 15 November 2020].
- [9] *Circo: A fast, scalable and extensible actor system*. <https://github.com/Circo-dev/Circo>, [Accessed: 22 December 2020].
- [10] S. CLEBSCH, S. DROSSOPOULOU, S. BLESSING, A. MCNEIL: *Deny Capabilities for Safe, Fast Actors*, in: AGERE 2015, Association for Computing Machinery, 2015, pp. 1–12, DOI: <https://doi.org/10.1145/2824815.2824816>.
- [11] J. DE KOSTER, T. VAN CUTSEM, W. DE MEUTER: *43 years of actors: a taxonomy of actor models and their key properties*, in: Oct. 2016, pp. 31–40, DOI: <https://doi.org/10.1145/3001886.3001890>.
- [12] T. M. FRUCHTERMAN, E. M. REINGOLD: *Graph drawing by force-directed placement*, Software: Practice and experience 21.11 (1991), pp. 1129–1164.
- [13] T. LI, Y. REN, D. YU, S. JIN: *Analysis of NUMA effects in modern multicore systems for the design of high-performance data transfer applications*, Future Generation Computer Systems 74 (2017), pp. 41–50, ISSN: 0167-739X, DOI: <https://doi.org/10.1016/j.future.2017.04.001>.
- [14] S. WÖLKE, R. HIESGEN, D. CHAROUSSET, T. C. SCHMIDT: *Locality-Guided Scheduling in CAF*, in: Proceedings of the 7th ACM SIGPLAN International Workshop on Programming Based on Actors, Agents, and Decentralized Control, AGERE 2017, Vancouver, BC, Canada: Association for Computing Machinery, 2017, pp. 11–20, ISBN: 9781450355162, DOI: <https://doi.org/10.1145/3141834.3141836>.
- [15] D. WYATT: *Akka Concurrency*, Sunnyvale, CA, USA: Artima Incorporation, 2013, ISBN: 0981531660.

SimBPDD: Simulating differential distributions in Beta-Poisson models, in particular for single-cell RNA sequencing data

Roman Schefzik

German Cancer Research Center (DKFZ),
Heidelberg, Germany

Current affiliation: Medical Faculty Mannheim,
Heidelberg University, Germany

Roman.Schefzik@medma.uni-heidelberg.de

Submitted: December 19, 2020

Accepted: March 8, 2021

Published online: May 18, 2021

Abstract

Beta-Poisson (BP) models employ Poisson distributions, where the corresponding rate parameter itself is a Beta-distributed random variable. They have been shown to appropriately mimic gene expression distributions in the context of single-cell ribonucleic acid sequencing (scRNA-seq), a breakthrough technology allowing to sequence information from individual biological cells and facilitating fundamental insights into numerous fields of biology. A prominent scRNA-seq data analysis task is to identify differences in gene expression distributions across two conditions. To validate new statistical approaches in this context, one typically has to rely on accurate simulations, as usually no ground truth for an assessment is available. We introduce several simulation procedures that allow to generate differential distributions (DDs) based on BP models. In particular, we describe how to create different types of DDs, mirroring various sources or origins of a difference, and different degrees of DDs, from a weak to a strong difference. The soundness of the simulation procedures is shown in a validation study in which theoretically expected model properties of the DD simulations are confirmed. The findings are in principle not restricted to the scRNA-seq context and may be gener-

ally applicable also to other application areas. The simulation approaches are implemented in the publicly available R package `SimBPDD`.

Keywords: Beta-Poisson model, differential distributions, single-cell RNA sequencing, Wasserstein distance

AMS Subject Classification: 62P10, 62-04, 62-08, 92-08

1. Introduction

Beta-Poisson (BP) models, sometimes also referred to as Poisson-Beta models, employ Poisson distributions, where the corresponding rate parameter itself is a Beta-distributed random variable [3, 4]. Thus, the BP distribution is an example of a mixed Poisson distribution [6] and a discrete compound distribution, respectively. It is used in various theoretical and practical applications [8, 13, 15, 17].

Specifically, the BP distribution has been recently used in the biological context to model single-cell ribonucleic acid sequencing (scRNA-seq) data [15, 17]. Due to major technological advances, it is nowadays possible to sequence information from individual biological cells. Such single-cell sequencing, and in particular the scRNA-seq, enables the quantification of cellular heterogeneity and provides new fundamental insights into various biological fields [16], thus being highly relevant. Along with the ever increasing amount of produced scRNA-seq data, there is a need to develop statistical methods for the analysis of such data [1]. The most striking difference compared to previous data obtained by bulk experiments is that gene expression in scRNA-seq data is available over multiple cells and not only as an average single point value. Consequently, models for scRNA-seq gene expression should take the form of distributions. Moreover, they should take account of the specific nature of scRNA-seq data (e.g. abundance of zero expression or increased variability). Besides other approaches, the BP distribution considered in this paper has been shown to model scRNA-seq data appropriately, where there are different procedures for model fitting and estimation of model parameters [2, 15].

To evaluate novel statistical methods in scRNA-seq data analysis, simulations play a very important role, as typically no ground truth is available for real data. For instance, to adequately test and validate differential expression methods for scRNA-seq data [2], it is important to simulate differential distributions (DDs) in a reliable way. While there are already methods to do so [18], we here explicitly focus on a specific procedure to generate DDs in the scRNA-seq context using BP models. In particular, we describe how to create different types of DDs, mirroring various sources or origins of a difference, and different degrees of DDs, from a weak to a strong difference.

While the focus of this paper is on using BP models in the context of scRNA-seq data, the generation of DDs for the BP models is generally applicable also to other application areas, for both theoretical and practical considerations.

2. Beta-Poisson models in scRNA-seq data

We here consider Beta-Poisson (BP) models as introduced in [15], which have been found to appropriately mimic scRNA-seq data. Precisely, in [15], three different BP models are considered: a three-parameter BP model (BP₃), a four-parameter BP model (BP₄) and a five-parameter BP model (BP₅).

The BP₃ model is a mixture of Poisson distributions $\text{Poi}(\lambda_1 u)$ with mean $\lambda_1 u$, where $\lambda_1 \in (0, \infty)$ denotes a scaling parameter and $u \sim \text{Beta}(\alpha, \beta)$ has a Beta distribution with parameters $\alpha \in (0, \infty)$ and $\beta \in (0, \infty)$:

$$X \sim \text{BP}_3(x|\alpha, \beta, \lambda_1) := \text{Poi}(x|\lambda_1 \text{Beta}(\alpha, \beta)).$$

Here, α is a shape parameter, where a large α indicates a high burst frequency (i.e. transcription rate, where transcription bursts correspond to an “on” state), and β is a scale parameter, with a large β indicating a high burst size [15, 17]. We can interpret this in the way that α may reflect among others the number of zero expression values (i.e. the proportion of zero expression), while β may mirror the size or magnitude of the non-zero expression values.

According to [15], the mean and the variance of the BP₃ model are given by

$$\mathbb{E}(X) = \lambda_1 \frac{\alpha}{\alpha + \beta}$$

and

$$\text{Var}(X) = \lambda_1 \frac{\alpha}{\alpha + \beta} + \lambda_1^2 \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)},$$

respectively.

As the BP₃ model can only account for count data (i.e. non-negative integers), the BP₄ model is proposed in [15], which employs an additional parameter $\lambda_2 \in (0, \infty)$ to allow for modeling non-negative real-valued data, i.e., the usual data format we have to deal with after normalization of the raw scRNA-seq count data:

$$Y \sim \text{BP}_4(x|\alpha, \beta, \lambda_1, \lambda_2) := \lambda_2 \text{BP}_3(x|\alpha, \beta, \lambda_1).$$

In addition to what has been done in [15], straightforward calculations yield

$$\mathbb{E}(Y) = \lambda_2 \lambda_1 \frac{\alpha}{\alpha + \beta}$$

and

$$\text{Var}(Y) = \lambda_2^2 \left(\lambda_1 \frac{\alpha}{\alpha + \beta} + \lambda_1^2 \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)} \right).$$

Finally, the BP₅ model has an additional parameter $p_0 \in [0, 1]$ explicitly capturing the proportion of cells with zero expression (besides the parameter α reflecting the burst frequency, as discussed before):

$$Z \sim \text{BP}_5(x|\alpha, \beta, \lambda_1, \lambda_2, p_0) := p_0 \mathbb{1}_{\{x=0\}} + (1 - p_0) \text{BP}_4(x|\alpha, \beta, \lambda_1, \lambda_2) \mathbb{1}_{\{x>0\}},$$

with $\mathbf{1}$ denoting the indicator function. In addition to what has been outlined in [15], by applying corresponding formulas for mixture distributions, it can be computed that

$$\mathbb{E}(Z) = (1 - p_0)\lambda_2\lambda_1 \frac{\alpha}{\alpha + \beta}$$

and

$$\text{Var}(Z) = (1 - p_0)(\mathbb{E}(Y)^2 + \text{Var}(Y)) - \mathbb{E}(Z)^2.$$

Note that for $\lambda_2 := 1$ and $p_0 := 0$, the BP₄ and the BP₅ models actually reduce to the BP₃ model.

3. Simulating differential distributions for Beta-Poisson models

The starting point of our procedure is a pre-processed (including quality control and normalization) real-experiment scRNA-seq data set in form of a $(G \times C)$ expression matrix, with G denoting the number of genes and C the number of cells. We first fit a BP₅ model to the expression data for each gene separately using the method provided by [15] in the R [12] package BPSC and obtain corresponding parameter estimates $\alpha, \beta, \lambda_1, \lambda_2$ and p_0 . Further, we test for each gene whether its distribution is indeed fitted well by the corresponding BP₅ model, using the procedure proposed in Section 3.2 in [15]. While filtering out low-quality fits, the cases (genes) that show a good fit, together with their corresponding parameter estimates, are kept in our pipeline and will be referred to as the controls in what follows.

We then simulate differential distributions (DDs) for each control Z separately by manipulating the corresponding parameters α, β and λ_1 . We do not explicitly consider a manipulation of the parameter λ_2 here, as λ_2 only controls the transformation from (a discrete spectrum of) non-negative integers (expression counts) to (a discrete spectrum of) non-negative real values (expression after normalization). Moreover, we do not consider a manipulation of the parameter p_0 at this point; however, this will be discussed at the end of the section, when we explicitly describe how to construct differential proportions of zero expression (DPZ) in the context of BP₅ models.

Here, we consider multiplicative manipulations of the parameters α, β and λ_1 as follows, where we set $\lambda := \lambda_1$ for simplicity (as λ_2 is anyway not explicitly considered): $\lambda \mapsto \Delta_\lambda \lambda$, $\alpha \mapsto \Delta_\alpha \alpha$, $\beta \mapsto \Delta_\beta \beta$. The parameters obtained by (one or multiple of) these transformations are then the corresponding parameters in the manipulated BP₅ model \tilde{Z} . As $\lambda \in (0, \infty)$, $\alpha \in (0, \infty)$ and $\beta \in (0, \infty)$, it must hold that $\Delta_\lambda \in (0, \infty)$, $\Delta_\alpha \in (0, \infty)$ and $\Delta_\beta \in (0, \infty)$, respectively, to get a reasonable model.

We not only want to create DDs, but also to incorporate different degrees θ of DD that range from weak to strong differences. Here, a degree θ of DD between a control BP₅ model Z and a manipulated BP₅ model \tilde{Z} is first introduced using a

multiplicative change (i.e., a fold change) with respect to the expected value:

$$\mathbb{E}(\tilde{Z}) = \theta \mathbb{E}(Z).$$

Inserting the corresponding expressions for the expected values and some algebra yields

$$\theta = \Delta_\lambda \cdot \frac{\Delta_\alpha \alpha + \Delta_\alpha \beta}{\Delta_\alpha \alpha + \Delta_\beta \beta}. \tag{3.1}$$

Hence, $\theta = \theta(\Delta_\lambda, \Delta_\alpha, \Delta_\beta)$ can be viewed as a function of $\Delta_\lambda, \Delta_\alpha$ and Δ_β , and the degree of DD can be specified by varying $\Delta_\lambda, \Delta_\alpha$ and Δ_β , respectively. Vice versa, we may consider Δ_λ as a function of θ in case Δ_α and Δ_β are fixed, i.e., $\Delta_\lambda = \Delta_\lambda(\theta)$. Analogously, $\Delta_\alpha = \Delta_\alpha(\theta)$ in case Δ_λ and Δ_β are fixed, and $\Delta_\beta = \Delta_\beta(\theta)$ in case Δ_λ and Δ_α are fixed. Note here again that θ generally refers to the degree of the DD (weak to strong), while $\Delta_\lambda, \Delta_\alpha$ or Δ_β represents the model manipulation that is necessary in order to achieve a DD of degree θ .

To get an understanding of the influence of the single parameter manipulations and to facilitate interpretability, we here first only consider those cases in which only one of the three original BP₅ model parameters is changed.

Case DLambda:

Here, the model is changed by manipulating the parameter λ only: $\lambda \mapsto \Delta_\lambda \lambda$, and $\Delta_\alpha = \Delta_\beta = 1$.

By inserting the corresponding expressions in (3.1), we get

$$\theta = \theta(\Delta_\lambda) = \mathbb{E}(\tilde{Z})/\mathbb{E}(Z) = \Delta_\lambda,$$

i.e.,

$$\Delta_\lambda = \Delta_\lambda(\theta) = \theta.$$

As $\theta(0) = 0$ and $\theta(\Delta_\lambda) \rightarrow \infty$ for $\Delta_\lambda \rightarrow \infty$, θ is bounded from below by zero, but has no upper bound. Hence, the range for possible values of θ is $\theta \in (0, \infty)$. DDs of arbitrary degree can be created using either positively oriented or negatively oriented fold changes. Here, a negatively oriented fold change means that $\mathbb{E}(\tilde{Z}) < \mathbb{E}(Z)$, hence, $\theta \in (0, 1)$. Conversely, a positively oriented fold change means that $\mathbb{E}(\tilde{Z}) > \mathbb{E}(Z)$, hence, $\theta \in (1, \infty)$. For instance, a positively oriented fold change of 3 in our setting here practically has the same effect as a negatively oriented fold change of 1/3, as we are only interested in the magnitude (i.e. the degree) of the difference here, and not in the direction of the change.

A manipulation of the scaling parameter λ in the BP model, while keeping Beta(α, β) unmodified, changes location (mean) and size (variance). In contrast, the shape should be affected only to a minor extent by a manipulation as here, if at all [15, 17]. Moreover, a change of λ should not affect the proportion of zero expression too much.

Case DAlpha:

Here, the model is changed by manipulating the parameter α only: $\alpha \mapsto \Delta_\alpha \alpha$, and $\Delta_\lambda = \Delta_\beta = 1$.

By inserting the corresponding expressions in (3.1), we get

$$\theta = \theta(\Delta_\alpha) = \mathbb{E}(\tilde{Z})/\mathbb{E}(Z) = \frac{\Delta_\alpha(\alpha + \beta)}{\Delta_\alpha \alpha + \beta},$$

i.e.,

$$\Delta_\alpha = \Delta_\alpha(\theta) = \frac{\beta\theta}{(\alpha + \beta) - \alpha\theta}.$$

As $\theta(0) = 0$ and $\lim_{\Delta_\alpha \rightarrow \infty} \theta(\Delta_\alpha) = 1 + \frac{\beta}{\alpha}$, θ is bounded from below by zero and has an upper bound $1 + \frac{\beta}{\alpha}$. Hence, the range for possible values of θ is $\theta \in (0, 1 + \frac{\beta}{\alpha})$. DDs can be generated using positively oriented (i.e. $\theta \in (1, 1 + \frac{\beta}{\alpha})$) or negatively oriented (i.e. $\theta \in (0, 1)$) fold changes. However, DDs of arbitrary degree can thus be created using negatively oriented fold changes only.

Here, location, size and shape can change. Also, a manipulation of α can affect the proportion of zero expression.

Case DBeta:

Here, the model is changed by manipulating the parameter β only: $\beta \mapsto \Delta_\beta \beta$, and $\Delta_\lambda = \Delta_\alpha = 1$.

By inserting the corresponding expressions in (3.1), we get

$$\theta = \theta(\Delta_\beta) = \mathbb{E}(\tilde{Z})/\mathbb{E}(Z) = \frac{\alpha + \beta}{\alpha + \Delta_\beta \beta},$$

i.e.,

$$\Delta_\beta = \Delta_\beta(\theta) = \frac{(\alpha + \beta) - \alpha\theta}{\beta\theta}.$$

As $\theta(0) = 1 + \frac{\beta}{\alpha}$ and $\lim_{\Delta_\beta \rightarrow \infty} \theta(\Delta_\beta) = 0$, θ is bounded from below by zero and has an upper bound $1 + \frac{\beta}{\alpha}$. Hence, the range for possible values of θ is $\theta \in (0, 1 + \frac{\beta}{\alpha})$. DDs can be generated using positively oriented (i.e. $\theta \in (1, 1 + \frac{\beta}{\alpha})$) or negatively oriented (i.e. $\theta \in (0, 1)$) fold changes. However, DDs of arbitrary degree can thus be created using negatively oriented fold changes only.

Here, location, size and shape can change. However, a manipulation of β should in principle not affect the proportion of zero expression too much.

We now consider a specific scenario, in which the expected value of the control BP₅ model is the same as that of the manipulated BP₅ model. The construction of such a type of DD may be relevant in case one wants to check whether an scRNA-seq differential expression analysis method is able to detect differences that are not caused by differences with respect to means [7].

Case DAlphaBeta:

Here, the model is changed by manipulating both the parameters α and β using a *common* parameter $\Delta := \Delta_\alpha = \Delta_\beta: \alpha \mapsto \Delta\alpha, \beta \mapsto \Delta\beta$, and $\Delta_\lambda = 1$.

As $\alpha, \beta \in (0, \infty)$, it must hold that $\Delta \in (0, \infty)$ to get a reasonable model. As discussed before, the expected value of the manipulated model \tilde{Z} in this setting is the same as the expected value of the control model $Z: \mathbb{E}(\tilde{Z}) = \mathbb{E}(Z)$. We therefore introduce DDs by considering a multiplicative manipulation (i.e., a fold change) θ of the variance instead of the expected value, with somewhat more complex formulas involved:

$$\text{Var}(\tilde{Z}) = \theta \text{Var}(Z).$$

Thus,

$$\begin{aligned} \theta &= \theta(\Delta) = \text{Var}(\tilde{Z}) / \text{Var}(Z) \\ &= \frac{1}{\text{Var}(Z)} \left[(1 - p_0) \left(\left(\lambda_1 \lambda_2 \frac{\alpha}{\alpha + \beta} \right)^2 + \lambda_2^2 \left(\lambda_1 \frac{\alpha}{\alpha + \beta} + \lambda_1^2 \frac{\alpha\beta}{(\alpha + \beta)^2 (\Delta(\alpha + \beta) + 1)} \right) \right) \right. \\ &\quad \left. - (1 - p_0)^2 \left(\lambda_1 \lambda_2 \frac{\alpha}{\alpha + \beta} \right)^2 \right] \\ &= \frac{1}{\text{Var}(Z)} \left[(1 - p_0) \left(\mathbb{E}(Y)^2 + \lambda_2 \mathbb{E}(Y) + \frac{\lambda_1 \lambda_2 \beta \mathbb{E}(Y)}{(\alpha + \beta)(\Delta(\alpha + \beta) + 1)} \right) - \mathbb{E}(Z)^2 \right], \end{aligned}$$

i.e., after some tedious calculations,

$$\begin{aligned} \Delta = \Delta(\theta) &= \frac{1}{\alpha + \beta} \left(\frac{\lambda_1^2 \lambda_2^2 \alpha \beta}{(\alpha + \beta)^2 \left(\frac{\text{Var}(Z)\theta + \mathbb{E}(Z)^2}{1 - p_0} - \mathbb{E}(Y)^2 - \lambda_2^2 \lambda_1 \frac{\alpha}{\alpha + \beta} \right)} - 1 \right) \\ &= \frac{1}{\alpha + \beta} \left(\frac{\lambda_1 \lambda_2 \beta \mathbb{E}(Y)}{(\alpha + \beta) \left(\frac{\text{Var}(Z)\theta + \mathbb{E}(Z)^2}{1 - p_0} - \mathbb{E}(Y)^2 - \lambda_2 \mathbb{E}(Y) \right)} - 1 \right). \end{aligned}$$

For the degree of DD θ , we have the upper bound

$$\begin{aligned} L_{\text{up}} &:= \theta(0) \\ &= \frac{1}{\text{Var}(Z)} \left[(1 - p_0) \left(\left(\lambda_1 \lambda_2 \frac{\alpha}{\alpha + \beta} \right)^2 + \lambda_2^2 \left(\lambda_1 \frac{\alpha}{\alpha + \beta} + \lambda_1^2 \frac{\alpha\beta}{(\alpha + \beta)^2} \right) \right) \right. \\ &\quad \left. - (1 - p_0)^2 \left(\lambda_1 \lambda_2 \frac{\alpha}{\alpha + \beta} \right)^2 \right] \\ &= \frac{1}{\text{Var}(Z)} \left[\mathbb{E}(Z) \left(\mathbb{E}(Y) + \lambda_2 \left(1 + \frac{\lambda_1 \beta}{\alpha + \beta} \right) - \mathbb{E}(Z) \right) \right] \end{aligned}$$

and the lower bound

$$L_{\text{low}} := \lim_{\Delta \rightarrow \infty} \theta(\Delta)$$

$$\begin{aligned}
 &= \frac{1}{\text{Var}(Z)} \left[(1 - p_0) \left(\left(\lambda_1 \lambda_2 \frac{\alpha}{\alpha + \beta} \right)^2 + \lambda_2^2 \lambda_1 \frac{\alpha}{\alpha + \beta} \right) - (1 - p_0)^2 \left(\lambda_1 \lambda_2 \frac{\alpha}{\alpha + \beta} \right)^2 \right] \\
 &= \frac{1}{\text{Var}(Z)} [\mathbb{E}(Z)(\mathbb{E}(Y) + \lambda_2 - \mathbb{E}(Z))].
 \end{aligned}$$

Hence, the range for possible values of θ is $\theta \in (L_{\text{low}}, L_{\text{up}})$, where $0 < L_{\text{low}} < 1 < L_{\text{up}}$. It is therefore not possible to create arbitrary degrees of DD in each case, be it for positively or negatively oriented fold changes with respect to the variance.

Note again that here, only size and shape change, but not the location. However, also the proportion of zero expression can change, as α varies, even though a variation of β should have no effect on this.

Finally, we consider the construction of manipulated BP₅ models with an explicit difference with respect to the proportion of zero expression, compared to the control model.

Table 1. Settings for the DD simulations based on BP models, where \times corresponds to “no” and \checkmark to “yes”.

case	differential distributions	changed parameter(s)			
		same location	same size	same shape	
DLambda	$Z \sim \text{BP}_5(x \alpha, \beta, \lambda_1, \lambda_2, p_0)$ vs. $\tilde{Z} \sim \text{BP}_5(x \alpha, \beta, \Delta_\lambda \lambda_1, \lambda_2, p_0)$	λ_1	\times	\times	\checkmark
DAlpha	$Z \sim \text{BP}_5(x \alpha, \beta, \lambda_1, \lambda_2, p_0)$ vs. $\tilde{Z} \sim \text{BP}_5(x \Delta_\alpha \alpha, \beta, \lambda_1, \lambda_2, p_0)$	α	\times	\times	\times
DBeta	$Z \sim \text{BP}_5(x \alpha, \beta, \lambda_1, \lambda_2, p_0)$ vs. $\tilde{Z} \sim \text{BP}_5(x \alpha, \Delta_\beta \beta, \lambda_1, \lambda_2, p_0)$	β	\times	\times	\times
DAlphaBeta	$Z \sim \text{BP}_5(x \alpha, \beta, \lambda_1, \lambda_2, p_0)$ vs. $\tilde{Z} \sim \text{BP}_5(x \Delta_\alpha \alpha, \Delta_\beta \beta, \lambda_1, \lambda_2, p_0)$	α, β	\checkmark	\times	\times
DPZ	$Z \sim \text{BP}_5(x \alpha, \beta, \lambda_1, \lambda_2, p_0)$ vs. $\tilde{Z} \sim \text{BP}_5(x \alpha, \beta, \lambda_1, \lambda_2, p_0 + \Delta_{p_0})$	p_0	\times	\times	\times

Case DPZ:

Here, the model is changed by manipulating the parameter p_0 of the control BP₅ model only: $p_0 \mapsto \tilde{p}_0 := p_0 + \Delta_{p_0}$, leading to differential proportions of zero expression (DPZ). While there is no intuitive feeling for the parameter ranges of the parameters λ, α and β , which is the reason why we used the models described above to construct different degrees of DDs for the other cases, we have an immediate and clear interpretability of the parameter p_0 .

As it has to hold that $p_0 + \Delta_{p_0} \in [0, 1]$ (since $p_0 \in [0, 1]$), we choose Δ_{p_0} as follows:

$$\Delta_{p_0} = \begin{cases} \theta, & \theta \leq 1 - p_0, \\ -\theta, & \theta < p_0, \end{cases}$$

where $\theta \in (0, 0.5]$. A change of p_0 should obviously affect the proportion of zero expression.

For an overview of all the considered settings described before, which are partly similar to those in [14], see the summaries in Tables 1 and 2. For the cases DLambda, DAlpha and DBeta, it is recommended to only consider negatively oriented fold changes θ (i.e. $\theta \in (0, 1)$), as all possible degrees of DDs can be achieved only using them. For the case DAlphaBeta, all possible degrees of DDs indeed cannot be achieved with negatively oriented fold changes, but neither this works for positively oriented fold changes. Hence, for reasons of consistency, we by default also focus on negatively oriented fold changes then.

Table 2. General overview of the different manipulated models \tilde{Z} of the control BP₅ models Z . Note that for the case DAlphaBeta, $L_{\text{low}} := \frac{1}{\text{Var}(Z)} [\mathbb{E}(Z)(\mathbb{E}(Y) + \lambda_2 - \mathbb{E}(Z))]$ and $L_{\text{up}} := \frac{1}{\text{Var}(Z)} \left[\mathbb{E}(Z) \left(\mathbb{E}(Y) + \lambda_2 \left(1 + \frac{\lambda_1 \beta}{\alpha + \beta} \right) - \mathbb{E}(Z) \right) \right]$. Here, Δ may refer to $\Delta_\lambda, \Delta_\alpha, \Delta_\beta$ or Δ_{p_0} , according to the descriptions of the corresponding cases in the main text.

case	manipulation	choice of Δ	possible values for θ
DLambda	$\mathbb{E}(\tilde{Z}) = \theta \mathbb{E}(Z)$	$\Delta = \theta$	$\theta \in (0, \infty)$
DAlpha	$\mathbb{E}(\tilde{Z}) = \theta \mathbb{E}(Z)$	$\Delta = \frac{\beta \theta}{(\alpha + \beta) - \alpha \theta}$	$\theta \in (0, 1 + \frac{\beta}{\alpha})$
DBeta	$\mathbb{E}(\tilde{Z}) = \theta \mathbb{E}(Z)$	$\Delta = \frac{\beta \theta}{(\alpha + \beta) - \alpha \theta}$	$\theta \in (0, 1 + \frac{\beta}{\alpha})$
DAlphaBeta	$\text{Var}(\tilde{Z}) = \theta \text{Var}(Z)$	$\Delta = \frac{1}{\alpha + \beta} \times$ $\left(\frac{\lambda_1 \lambda_2 \beta \mathbb{E}(Y)}{(\alpha + \beta) \left(\frac{\text{Var}(Z) \theta + \mathbb{E}(Z)^2}{1 - p_0} - \mathbb{E}(Y)^2 - \lambda_2 \mathbb{E}(Y) \right)} - 1 \right)$	$\theta \in (L_{\text{low}}, L_{\text{up}})$
DPZ	$\tilde{p}_0 = p_0 + \Delta$	$\Delta = \begin{cases} \theta, & \theta \leq 1 - p_0 \\ -\theta, & \theta < p_0 \end{cases}$	$\theta \in (0, 0.5]$

4. Validation study

4.1. Evaluation tools

To validate the soundness of our simulation procedures, we employ the waddR tool available at <https://github.com/goncalves-lab/waddR>. Specifically, a semi-parametric, permutation-based test using the 2-Wasserstein distance is applied to compare two distributions F_A and F_B [10].

In our validation study, for each instance (here, each gene), information about F_A (the control model) and F_B (the manipulated model) is available in the form of a sample $x_{A,1}, \dots, x_{A,C_A}$ from F_A , and $x_{B,1}, \dots, x_{B,C_B}$ from F_B , respectively, where in general, C_A does not need to equal C_B . In the context of scRNA-seq data, the sample sizes C_A and C_B correspond to the respective numbers of cells. Using the corresponding empirical cumulative distribution functions \hat{F}_A and \hat{F}_B as

approximations, the (squared) 2-Wasserstein distance d is then computed by

$$\begin{aligned}
 d(\hat{F}_A, \hat{F}_B) &\approx \frac{1}{K} \sum_{k=1}^K (Q_A^{\alpha_k} - Q_B^{\alpha_k})^2 \\
 &\approx \underbrace{(\hat{\mu}_A - \hat{\mu}_B)^2}_{\text{location}} + \underbrace{(\hat{\sigma}_A - \hat{\sigma}_B)^2}_{\text{size}} + \underbrace{2\hat{\sigma}_A\hat{\sigma}_B(1 - \hat{\rho}_{A,B})}_{\text{shape}}, \quad (4.1) \\
 &\hspace{15em} \underbrace{\hspace{10em}}_{\text{variability}}
 \end{aligned}$$

with $(Q_A^{\alpha_k})_{k=1,\dots,K}$ and $(Q_B^{\alpha_k})_{k=1,\dots,K}$ denoting the α_k -quantiles of \hat{F}_A and \hat{F}_B , respectively, where we use equidistant levels $\alpha_k = \frac{k-0.5}{K}$, $k = 1, \dots, K$. Here, $\hat{\mu}_A, \hat{\mu}_B$ denote the corresponding empirical means, $\hat{\sigma}_A, \hat{\sigma}_B$ the corresponding empirical standard deviations, and

$$\hat{\rho}_{A,B} := \text{Cor}((Q_A^{\alpha_1}, \dots, Q_A^{\alpha_K}), (Q_B^{\alpha_1}, \dots, Q_B^{\alpha_K}))$$

the sample Pearson correlation coefficient between $(Q_A^{\alpha_k})_{k=1,\dots,K}$ and $(Q_B^{\alpha_k})_{k=1,\dots,K}$. For the calculations, we use $K := 1000$ here.

For each instance separately, we calculate the corresponding 2-Wasserstein distance as a test statistic and obtain a p-value using a semi-parametric, permutation-based testing procedure involving a generalized Pareto distribution approximation to estimate very small p-values accurately. Along with the p-value, the decomposition of the 2-Wasserstein distance in (4.1) may help to judge whether overall differences between two distributions (i.e. BP models) are mainly due to differences with respect to location (referring to differences with respect to the expected values), size (referring to differences with respect to the standard deviations) and/or shape (referring to differences not mainly caused by differences with respect to expected values and/or standard deviations) [5, 9].

To explicitly test for DPZ, we use Fisher's exact test, applied to each instance separately.

4.2. Setting

In the following validation study, we start with the real-experiment scRNA-seq data set in [11], downloaded from <https://hemberg-lab.github.io/scRNA.seq.datasets/human/tissues>. The data set consists of log2-transformed TPM (transcripts per million) expression values, normalized for both gene length and sequencing depth, for 301 cells, where we only keep those genes from the original data set that are expressed in at least three cells. A BP₅ model is fitted to each gene in the data set using the R package BPSC [15]. Specifically, maximum likelihood estimation combined with a binning approach to reduce computation time is employed to estimate the model parameters, using the standard R function `optim` for optimization. For more details, in particular the choice of initial values for the BP model optimization, see Section 3 in [15]. To assess the quality of the BP₅ model fits, a goodness-of-fit test statistic comparing the observed and expected frequencies

from the model is considered, where a Monte-Carlo method is used to generate a suitable null distribution that is employed to derive a corresponding p-value P . A gene is then declared to be fitted well by the BP model if $P \geq 0.05$. For details, see Section 3.2 in [15]. In our study, 8773 genes are declared to be fitted well by the corresponding BP_5 model. For the further analyses, we keep only these well-fitted genes as controls, from which manipulated BP_5 models are then constructed according to the procedures discussed before. At this point, we emphasize that for our purposes here, the real data set in [11] is only used for obtaining reasonable BP model parameters in the scRNA-seq context, from which the control and manipulated BP models in our purely numerical experiments are constructed. However, no specific biological investigations or aspects are considered in our validation study.

For each case (see Tables 1 and 2), we here consider five different degrees θ of DD, ranging from weak to strong, where the explicit choices for θ are shown in Table 3. Note that for the cases DLambda, DAlpha, DBeta and DPZ, these degrees of DD can be achieved for all 8773 genes in the simulation study, and all these genes are used in the studies. In contrast, for the case DAlphaBeta, due to the existence of the lower bound L_{low} , we only keep those genes for the study for which the corresponding degree of DD can be achieved.

As representatives of the corresponding control and manipulated BP_5 models, for each gene, we draw a sample from each model. In this context, the samples from a BP distribution are independent random draws. Specifically, the function `rBP` from the `BPSC` package is used for drawing the samples from the BP distributions, which combines the classical `rpois` and `rbeta` functions for randomly drawing from Poisson and Beta distributions, respectively, in R. In our study, we for convenience consider situations in which the sample size (i.e. the number of cells) $C := C_A = C_B$ in both conditions (control and manipulated) is equal and cover a range $C \in \{25, 50, 75, 100, 500\}$ of examples for small to large sample sizes.

Table 3. Different degrees of DDs specifically chosen in the simulation study.

case \ degree	D1	D2	D3	D4	D5
DLambda, DAlpha, DBeta, DAlphaBeta DPZ	$\theta = 10/11$ $\theta = 0.05$ weak	$\theta = 2/3$ $\theta = 0.1$	$\theta = 1/2$ $\theta = 0.25$	$\theta = 2/5$ $\theta = 0.4$	$\theta = 1/3$ $\theta = 0.5$ strong

4.3. Results

We now discuss the results for the validation study in terms of detection power and the decomposition of the 2-Wasserstein distance in the `waddR` test. In this context, for each fixed case, degree of DD and number of cells, detection power is defined as

$$\text{detection power} = \frac{\# \text{ p-values} \leq \alpha}{\# \text{ tests (genes)}}$$

Table 4. Detection powers (in %), based on p-values at a 5% significance level, with varying degrees of DD (D1: weak to D5: strong), numbers of cells $C \in \{25, 50, 75, 100, 500\}$ and cases from Table 1.

		degree	D1	D2	D3	D4	D5
case							
$C = 25$	DLambda	waddR DD	1.39	8.48	21.86	32.06	38.31
		Fisher DPZ	0.25	0.40	0.48	1.03	1.87
	DAlpha	waddR DD	1.57	7.67	18.03	26.73	32.73
		Fisher DPZ	0.30	1.45	5.11	10.48	16.95
	DBeta	waddR DD	1.61	8.36	20.43	30.42	36.24
		Fisher DPZ	0.25	0.42	0.96	1.74	2.74
	DAlphaBeta	waddR DD	1.35	2.50	5.50	0.87	12.66
		Fisher DPZ	0.20	1.64	7.55	16.53	25.67
	DPZ	waddR DD	1.37	2.09	23.50	54.21	65.61
Fisher DPZ		0.26	0.39	12.71	68.97	77.03	
$C = 50$	DLambda	waddR DD	2.17	17.91	40.61	51.61	58.00
		Fisher DPZ	0.40	0.52	1.58	3.21	5.49
	DAlpha	waddR DD	2.14	14.89	34.22	44.88	50.95
		Fisher DPZ	0.59	3.65	14.97	29.02	38.32
	DBeta	waddR DD	2.47	16.77	37.25	48.93	55.08
		Fisher DPZ	0.34	0.80	2.43	5.32	8.96
	DAlphaBeta	waddR DD	1.62	4.09	10.92	19.49	23.96
		Fisher DPZ	0.51	4.29	19.48	37.09	53.11
	DPZ	waddR DD	1.99	6.62	50.83	75.56	81.33
Fisher DPZ		0.57	1.30	67.11	82.75	86.11	
$C = 75$	DLambda	waddR DD	2.83	27.62	53.06	63.75	69.63
		Fisher DPZ	0.54	0.97	2.58	5.04	8.36
	DAlpha	waddR DD	2.72	22.61	44.74	56.35	62.93
		Fisher DPZ	0.68	6.38	26.38	42.00	51.76
	DBeta	waddR DD	2.36	25.35	49.29	60.31	67.05
		Fisher DPZ	0.51	1.42	4.34	9.18	15.22
	DAlphaBeta	waddR DD	1.83	6.01	18.10	26.30	32.86
		Fisher DPZ	0.83	7.54	33.13	56.49	70.34
	DPZ	waddR DD	2.62	12.19	66.05	81.98	84.94
Fisher DPZ		0.81	7.61	79.41	86.78	88.93	
$C = 100$	DLambda	waddR DD	3.00	34.41	60.21	71.21	76.56
		Fisher DPZ	0.51	0.95	3.08	6.84	11.23
	DAlpha	waddR DD	2.90	28.78	52.10	63.48	69.96
		Fisher DPZ	0.88	9.80	34.42	50.88	60.58
	DBeta	waddR DD	3.05	31.63	56.41	67.54	74.75
		Fisher DPZ	0.59	1.89	6.19	12.90	20.81
	DAlphaBeta	waddR DD	1.62	9.05	26.51	35.93	40.39
		Fisher DPZ	0.98	11.42	45.10	69.32	81.52
	DPZ	waddR DD	3.64	17.85	74.09	83.98	86.91
Fisher DPZ		1.09	15.76	83.11	88.62	90.74	
$C = 500$	DLambda	waddR DD	11.67	76.78	93.35	96.82	97.61
		Fisher DPZ	0.81	5.30	15.05	30.35	46.37
	DAlpha	waddR DD	9.86	68.94	87.43	92.57	94.85
		Fisher DPZ	2.68	53.03	78.10	88.13	92.83
	DBeta	waddR DD	10.40	72.12	90.07	94.94	96.90
		Fisher DPZ	0.93	11.66	38.47	58.55	68.68
	DAlphaBeta	waddR DD	3.04	62.78	78.12	85.17	89.33
		Fisher DPZ	2.87	59.94	92.03	96.65	98.57
	DPZ	waddR DD	23.74	70.61	87.99	91.51	93.25
Fisher DPZ		27.50	82.00	92.10	96.57	98.34	

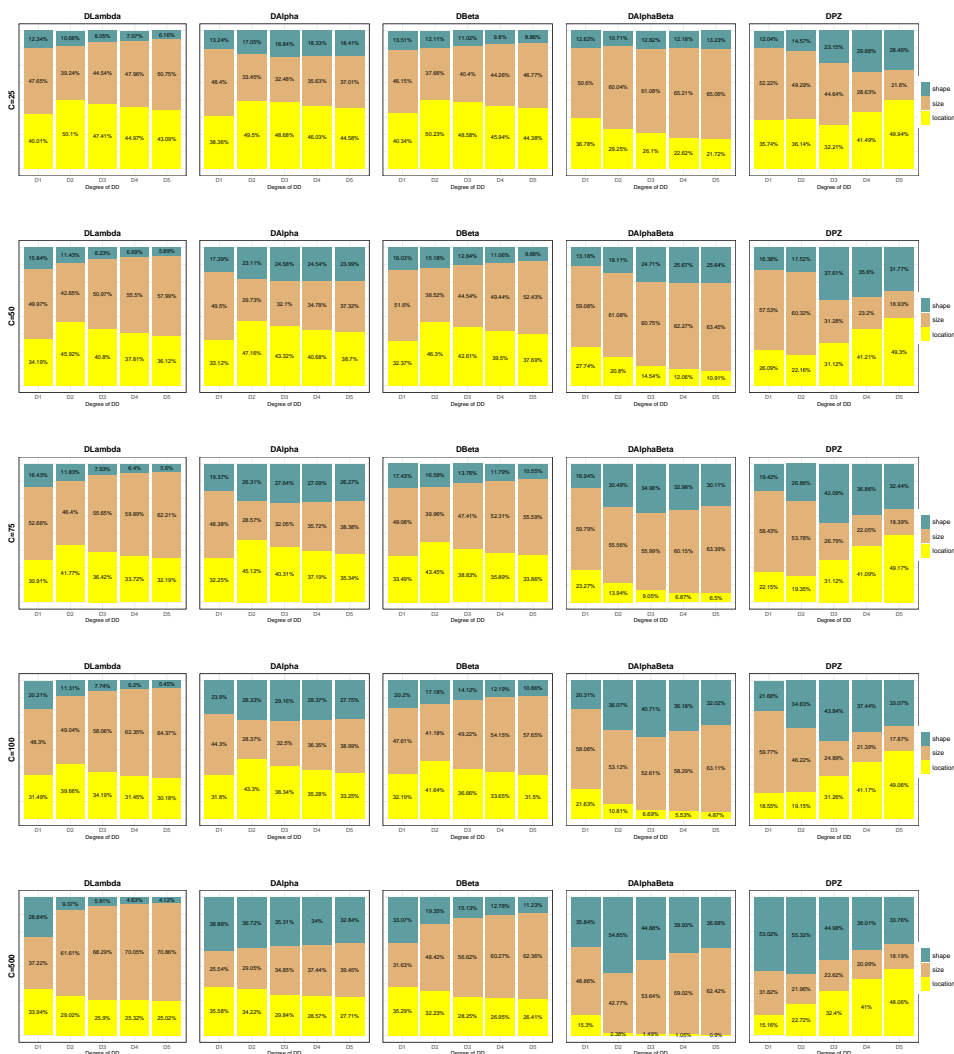


Figure 1. Decomposition results for the waddR test for varying degrees of DD (D1: weak to D5: strong) and numbers of cells $C \in \{25, 50, 75, 100, 500\}$, based on averages over those of the runs that are considered to show significant DDs in that the corresponding p-value is ≤ 0.05 , with cases according to Table 1.

with $\alpha \in (0, 1)$. Detection powers for the different numbers of cells for the standard level of $\alpha = 5\%$ are listed in Table 4. In general, for all cases, detection powers meaningfully increase with increasing numbers of cells. Moreover, they increase with increasing strength of the difference between the distributions (weak to strong degree of DD; D1 to D5). While only very little detection power can be

observed for the very weak degree of DD D1, the detection powers get bigger for the stronger degrees of DD, for which the differences become more and more obvious. This intuitively makes sense and confirms in particular that the implementation of the varying degrees of DD from weak to strong in our simulation procedure is valid. When comparing the p-values of the `waddR` test and the separate DPZ test, we observe that DPZ can mainly be detected when the parameters α or p_0 are changed (i.e. in the cases DAlpha, DAlphaBeta and DPZ). In contrast, DPZ typically plays only a minor role when the parameters λ or β are changed (i.e. in the cases DLambda and DBeta). This is in line with the theoretical properties and the interpretation of the parameters of the BP model.

A further confirmation that the simulation procedures are able to reflect what is to be expected from the underlying theory (Table 1) of the BP model is given by the decomposition of the 2-Wasserstein distance into location, size and shape parts within the `waddR` test. For the different degrees of DDs and numbers of cells, Figure 1 shows for all cases the average fractions of the location, size and shape parts with respect to the overall 2-Wasserstein distance for the `waddR` test based on those runs with a p-value less than or equal to 5%. Again, the respective decomposition patterns meaningfully become more and more obvious the larger the number of cells is and the stronger the degree of DD is. In particular, the shape and location component in the cases DLambda and DAlphaBeta, respectively, are minor to negligible compared to the corresponding other components, in line with the theoretical models according to Table 1. Moreover, for instance, the shape component is more pronounced in cases in which the shape parameter α is changed (i.e. in the cases DAlpha and DAlphaBeta) than in those where α is not changed (i.e. in the cases DLambda and DBeta). An explicit change of the proportion of zero expression by manipulating the parameter p_0 (case DPZ) can obviously also affect the shape.

5. Discussion

We have discussed how to create DDs of varying degrees, ranging from weak to strong differences, for BP models, using various manipulations of the BP model parameters. The soundness of our approaches has been shown in a validation study, in which theoretically expected properties of our procedures have been confirmed.

In particular, based on the construction of our simulations and their validation in the study, we can provide some guidance on how to generate DDs between two BP models when the difference shall be of a specific type. For instance, when no difference with respect to shape is desired, the DLambda simulation, in which only the BP model parameter λ is changed, can be used. Similarly, in case no difference with respect to location is desired, one can employ the DAlphaBeta simulation, in which the BP model parameters α and β are changed using a common manipulation parameter. In case there shall be no DPZ, one may rely on the DLambda or DBeta simulations, in which only the BP model parameters λ and β , respectively, are changed.

Despite the focus of this paper is on the application field of scRNA-seq data, the introduced procedures can in principle be applied also to settings in other research areas.

While we have presented first attempts to simulate DDs for BP models here, we by far did not consider all possible combinations of BP parameter manipulations. This provides opportunities for future work, in which in particular interactions of changes when multiple BP parameters are manipulated simultaneously could be investigated in more detail. Moreover, up to now, only univariate BP distributions, that allow for individual (gene-wise) modeling, have been considered in the models here. However, certain variables may be correlated (such as genes in the scRNA-seq context), and taking account of specific correlation structures is an important issue that could be addressed in future extensions of the models.

Software availability. The simulations of DDs based on BP models presented in this paper are implemented in the R package `SimBPDD`, which is publicly available at <https://github.com/RomanSchefzik/SimBPDD>, along with documentation of the functions.

Acknowledgements. Angela Goncalves and Marc Schwering are thanked for helpful discussions and useful comments. Moreover, thanks are given to two anonymous reviewers for valuable comments and suggestions. The work was funded by project VH-NG-1010 of the HGF.

References

- [1] R. BACHER, C. KENDZIORSKI: *Design and computational analysis of single-cell RNA-sequencing experiments*, Genome Biology 17 (2016), art. 63, DOI: <https://doi.org/10.1186/s13059-016-0927-y>.
- [2] M. DELMANS, M. HEMBERG: *Discrete distributional differential expression (D^3E) – a tool for gene expression analysis of single-cell RNA-seq data*, BMC Bioinformatics 17 (2016), art. 110, DOI: <https://doi.org/10.1186/s12859-016-0944-6>.
- [3] J. GURLAND: *A generalized class of contagious distributions*, Biometrics 14 (1958), pp. 229–249, DOI: <https://doi.org/10.2307/2527787>.
- [4] M. S. HOLLA, S. K. BHATTACHARYA: *On a discrete compound distribution*, Annals of the Institute of Statistical Mathematics 17 (1965), pp. 377–384, DOI: <https://doi.org/10.1007/BF02868181>.
- [5] A. IRPINO, R. VERDE: *Basic statistics for distributional symbolic variables: a new metric-based approach*, Advances in Data Analysis and Classification 9 (2015), pp. 143–175, DOI: <https://doi.org/10.1007/s11634-014-0176-4>.
- [6] D. KARLIS, E. XEKALAKI: *Mixed Poisson distributions*, International Statistical Review 73 (2005), pp. 35–58.
- [7] K. D. KORTHAUER, L.-F. CHU, M. A. NEWTON, Y. LI, J. THOMSON, R. STEWART, C. KENDZIORSKI: *A statistical approach for identifying differential distributions in single-cell RNA-seq experiments*, Genome Biology 17 (2016), art. 222, DOI: <https://doi.org/10.1186/s13059-016-1077-y>.

- [8] K. L. LEASK, L. M. HAINES: *The beta-Poisson distribution in Wadley's problem*, Communications in Statistics—Theory and Methods 43 (2014), pp. 4962–4971, DOI: <https://doi.org/10.1080/03610926.2012.744047>.
- [9] Y. MATSUI, M. MIZUTA, S. ITO, S. MIYANO, T. SHIMAMURA: *D³M: detection of differential distributions of methylation levels*, Bioinformatics 32 (2016), pp. 2248–2255, DOI: <https://doi.org/10.1093/bioinformatics/btw138>.
- [10] V. M. PANARETOS, Y. ZEMEL: *Statistical aspects of Wasserstein distances*, Annual Review of Statistics and Its Application 6 (2019), pp. 405–431, DOI: <https://doi.org/10.1146/annurev-statistics-030718-104938>.
- [11] A. A. POLLEN, T. J. NOWAKOWSKI, J. SHUGA, X. WANG, A. A. LEYRAT, J. H. LUI, N. LI, L. SZPANKOWSKI, B. FOWLER, P. CHEN, N. RAMALINGAM, G. SUN, M. THU, M. NORRIS, R. LEBOFKY, D. TOPPANI, D. W. KEMP II, M. WONG, B. CLERKSON, B. N. JONES, S. WU, L. KNUTSSON, B. ALVARADO, J. WANG, L. S. WEAVER, A. P. MAY, R. C. JONES, M. A. UNGER, A. R. KRIEGSTEIN, J. A. A. WEST: *Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex*, Nature Biotechnology 32 (2014), pp. 1053–1058, DOI: <https://doi.org/10.1038/nbt.2967>.
- [12] R CORE TEAM: *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2020, URL: <https://www.R-project.org/>.
- [13] J. M. SARABIA, E. GÓMEZ-DÉNIZ: *Multivariate Poisson-Beta distributions with applications*, Communications in Statistics—Theory and Methods 40 (2011), pp. 1093–1108, DOI: <https://doi.org/10.1080/03610920903537269>.
- [14] M. SCHWERING: *Batch effects in single cell RNA sequencing analysis*, MA thesis, Heidelberg University, 2017.
- [15] T. N. VU, Q. F. WILLS, K. R. KALARI, N. NIU, L. WANG, M. RANTALAINEN, Y. PAWITAN: *Beta-Poisson model for single-cell RNA seq data analyses*, Bioinformatics 32 (2016), pp. 2128–2135, DOI: <https://doi.org/10.1093/bioinformatics/btw202>.
- [16] Y. WANG, N. E. NEVIN: *Advances and applications of single-cell sequencing technologies*, Molecular Cell 58 (2015), pp. 598–609, DOI: <https://doi.org/10.1016/j.molcel.2015.05.005>.
- [17] Q. F. WILLS, K. J. LIVAK, A. J. TIPPING, T. ENVER, A. J. GOLDSON, D. W. SEXTON, C. HOLMES: *Single-cell gene expression analysis reveals genetic associations masked in whole-tissue experiments*, Nature Biotechnology 31 (2013), pp. 748–752, DOI: <https://doi.org/10.1038/nbt.2642>.
- [18] L. ZAPPIA, B. Phipson, A. Oshlack: *Splatter: simulation of single-cell RNA sequencing data*, Genome Biology 18 (2017), art. 174, DOI: <https://doi.org/10.1186/s13059-017-1305-0>.

Abstractive text summarization for Hungarian

Zijian Győző Yang^{abc}, Ádám Agócs^a,
Gábor Kusper^a, Tamás Váradi^c

^aEszterházy Károly University, Faculty of Informatics
agadam98@gmail.com

{yang.zijian.gyozo, kusper.gabor}@uni-eszterhazy.hu

^bMTA-PPKE Hungarian Language Technology Research Group
yang.zijian.gyozo@itk.ppke.hu

^cHungarian Research Centre for Linguistics
{yang.zijian.gyozo, varadi.tamas}@nytud.hu

Submitted: November 22, 2020

Accepted: April 12, 2021

Published online: May 18, 2021

Abstract

In our research we have created a text summarization software tool for Hungarian using multilingual and Hungarian BERT-based models. Two types of text summarization method exist: abstractive and extractive. The abstractive summarization is more similar to human generated summarization. Target summaries may include phrases that the original text does not necessarily contain. This method generates the summarized text by applying keywords that were extracted from the original text. The extractive method summarizes the text by using the most important extracted phrases or sentences from the original text. In our research we have built both abstractive and extractive models for Hungarian. For abstractive models, we have used a multilingual BERT model and Hungarian monolingual BERT models. For extractive summarization, in addition to the BERT models, we have also made experiments with ELECTRA models. We find that the Hungarian monolingual models outperformed the multilingual BERT model in all cases. Furthermore, the ELECTRA small models achieved higher results than some of the BERT models. This result is important because the ELECTRA small models have much fewer parameters and were trained on only 1 GPU within a couple of days. Another important consideration is that the ELECTRA

models are much smaller than the BERT models, which is important for the end users. To our best knowledge the first extractive and abstractive summarization systems reported in the present paper are the first such systems for Hungarian.

Keywords: BERT, huBERT, ELECTRA, HILBERT, abstractive summarization, extractive summarization

AMS Subject Classification: 68T07, 68T50, 68T09

1. Introduction

Processing large amounts of textual data in our everyday life with manual tools is proving increasingly difficult because of the scale of the data. For instance, any company or public institution typically has an enormous amount of text data. It may be especially important for them to extract the essence of data from the huge body of texts. Using automatic methods for extracting or summarizing can lead to significant saving of time and costs. Therefore, there is an increasing demand for automatic information extraction applications. Automatic text summarization is a particularly pressing, unsolved challenge for the Hungarian language.

Automatic text summarization is the process of shortening a text document using a system for prioritizing information. Technologies that generate summaries take into account variables such as length, style, or syntax. Text summarization from the perspective of humans is taking a chunk of information and extracting the most important parts from it. Automatic text summarization methods typically rely on the logical quantification of features of the text including weighting keywords, and sentence ranking.

There are two different machine summarization methods: extractive and abstractive summarization.

Abstractive text summarization can generate completely new pieces of texts while capturing the meaning of the original article. Abstractive methods are usually more complex because the machine has to analyze the text and the most important information from it, then learn the relevant concepts and construct cohesive summaries.

Extractive text summarization does not generate any new text, it only uses words already in the original article and combines the existing words, phrases or sentences that are the most relevant to the article. Extractive summarization techniques include ranking sentences and phrases in order of importance, and selecting the most important components of a document to create a summary.

In our research we have carried out both extractive and abstractive experiments for Hungarian.

2. Related work

The extractive method creates summarization by selecting the most important phrases or sentences from the original text. It involves a classification problem:

the task is to find which sentences should be selected for inclusion in the summary. One of the first neural network-based extractive summarization tool is SummaRuNer [13], which uses an RNN encoder to solve the problem. Another method called Refresh [14] is based on the Rouge metric, which is used to rank sentences in the text using the reinforcement learning method. The goal of Latent [25] was to propose a latent variable extractive model where sentences are viewed as latent variables and sentences with activated variables are used to infer gold summaries. Sumo [9] uses a method that builds on multi-root dependency tree structures that can be extracted from a document and predicts the possible form of the summary. NeuSum [26] approaches the problem by scoring and selecting sentences from the original text.

The abstractive summarization with neural network approaches the problem as a transformation from a sequence into another sequence. The encoder identifies tokens from the source document, then maps them onto target tokens, and finally generates new text from the decoder. The PTgen [19] tool generates pointers to identify words in the source text, then using a coverage mechanism keeps the words to generate the summary. Deep Communicating Agent [1] is an agent-based approach where the task of encoding a long text is shared among multiple collaborating agents, each in charge of a subsection of the input text. These encoders are connected to a single decoder, trained end-to-end using reinforcement learning to generate a focused and coherent summary. The Deep Reinforced Model [17] uses an intra-attention that attends to the input and the continuously generated output separately, as well as a new training method that combines standard supervised word prediction and reinforcement learning. The Bottom-Up [4] approach uses a data-efficient content selector to “over-determine” phrases in a source document that should be part of the summary. The method uses this selector as a bottom-up attention step to constrain the model to likely phrases.

The PreSumm [8] model was the state-of-the-art tool in 2019. It requires a pretrained BERT model to train extractive and abstractive summarization models. Pre-training a BERT model requires huge data and compute capacity. Fortunately, we can choose the PreSumm model because recently a number of BERT models have been created for Hungarian. We can use the multilingual BERT¹, which, of course, covers Hungarian. There are also two Hungarian monolingual BERT base models built by Nemeskey [16] that we could use for our research.

In the recent months, further models for Hungarian were successfully trained²: HIL-ELECTRA, HIL-RoBERTa, HIL-ALBERT and HILBERT [3]. For the purposes of the present research we experimented with the HIL-ELECTRA and the HILBERT models.

In recent months, autoregressive methods achieved the best results in the field of summarization. Autoregressive models rely on the decoder of the transformer model and use an attention mask on the top of the full sentence so the model can only look at the tokens before the current text. This method achieved higher results

¹<https://github.com/google-research/bert/blob/master/multilingual.md>

²<https://hilanco.github.io/>

on many text generation tasks [22]. The BART model[6] is a denoising autoencoder for pretraining sequence-to-sequence models, which is trained to corrupt text with arbitrary noising function and then to learn to reconstruct the original text. This model is effective for fine-tuning summarization tasks.

Currently, the state-of-the-art tool for summarization is the PEGASUS [24] system. In PEGASUS, important sentences are removed/masked from an input document and are generated together as one output sequence from the remaining sentences, similar to an extractive summary. For Hungarian, the OpinHu system has a summary function [10]. The system uses keywords and text context to extract information. Lengyelne Molnár Tünde [12] examined the possibilities and limitations of the automatic generation of research abstracts. Using the PreSumm [8] tool, Yang et al. built the first extractive summarization tool [23] for Hungarian. In this paper we present the first Hungarian abstractive summarization tool. It was built using the PreSumm system.

3. BERT and ELECTRA models

In our experiments, different kinds of BERT models were used for our summarization tasks, ELECTRA model was tested in addition to the BERT model for the extractive summarization.

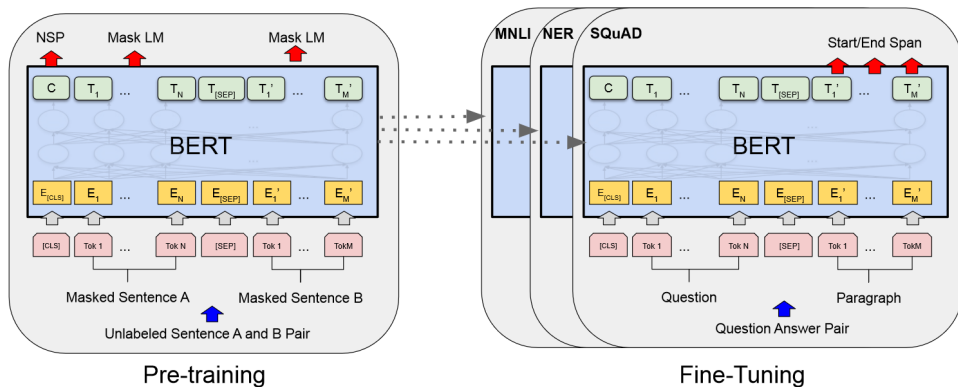


Figure 1. BERT model.

BERT (Bidirectional Encoder Representations from Transformer) is a multi-layer, bidirectional Transformer encoder [21]. The BERT model was trained on two language modeling tasks (see Figure 1): masking and next sentence prediction. During masking, 15% of the words in the corpus are randomly masked, then the system had to learn the correct word. During the next sentence prediction task, the model receives two sentences, the task is to guess whether the two received sentences are next to each other in the original text or two randomly selected sentences. To limit the size of the dictionary and to solve the out-of-vocabulary words problem,

WordPiece tokenizer [18] was used. After training BERT, the pretrained model is used to fine-tune to the target task. The BERT model is further trained on a specific downstream task with a feed-forward network in the fine-tuning process.

One of the advantages of BERT is that models have not only been trained on English. Google has trained two multilingual models³: lowercase and non-lowercase. The first 104 languages with the largest Wikipedia were selected to train the models. The size of Wikipedia varies greatly between the languages, the English Wikipedia accounting for nearly 20% of the data, so sampling was controlled by normalization to solve this problem. Then, all languages, same like English, were tokenized, which had four steps: lowercasing, accent removal, punctuation and whitespace handling. Training the non-lowercase model also followed these steps except lowercasing. WordPiece tokenization and dictionary can handle cased and unknown words. The Hungarian language is also part of this model.

The first Hungarian BERT model was published by Dávid Márk Nemeskey [16], which is called huBERT⁴. Three huBERT models were trained:

- huBERT: BERT base model trained on Hungarian Webcorpus 2.0⁵
- huBERT Wikipedia cased: cased BERT base model trained on Hungarian Wikipedia
- huBERT Wikipedia lowercased: lowercased BERT base model trained on Hungarian Wikipedia

Currently the huBERT models achieve state-of-the-art results in name entity recognition and noun phrase chunking tasks [15].

ELECTRA [2] is based on the GAN (Generative adversarial network) [5] method. The basis of the method (see Figure 2) is that there are two networks are trained, a generator and a discriminator. During training, the generator randomly generates vector representations from which it generates output. Then, real output data is shown, which can improve the performance of random vector generation. In this way, by the end of the training, the generator will become “smarter” and will be able to generate an output that closely matches the real output. The discriminator is trained to predict whether a particular word is the original word or a replacement. During training, the discriminator gets data from the real corpus, and also gets data that generated by the generator. By the end, the generator can generate content that similar to a real content, the discriminator can distinguish between a fake/erroneous content and a real/correct content.

ELECTRA is a modified GAN method for training language model (see Figure 2). The difference compared to the BERT model (and the original GAN) is that ELECTRA does not try to predict the original word behind the masked word but instead the generator randomly generates words for the masked words and then the discriminator has to guess if the words given by the generator are the original words

³<https://github.com/google-research/bert/blob/master/multilingual.md>

⁴<https://hlt.bme.hu/en/resources/hubert>

⁵<https://hlt.bme.hu/en/resources/webcorpus2>

or randomly generated words. Thus, the generator slowly learns what actual words match to the place of the masked words, while the discriminator learns whether the given input text are built with real words or fake words. After training, the generator is discarded and only the discriminator is retained for fine-tuning.

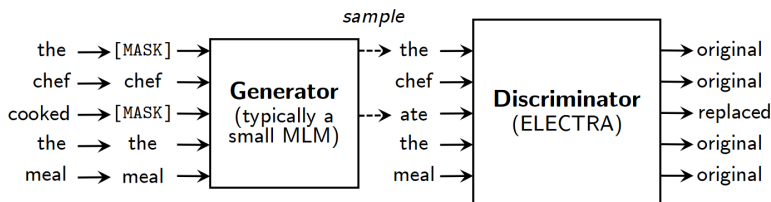


Figure 2. ELECTRA model.

4. Corpora

For building the summarization corpora for fine-tuning, we used 3 different kinds of resource: HVG, index.hu and the Hungarian MARCELL corpus [20]. Table 1 displays the main characteristics of the corpora.

Table 1. Main characteristics of the corpora.

	HVG	index.hu	H+I	MARCELL
year	2012 - 2020	1999 - 2020	-	1991 - 2019
documents	480,660	183,942	559,162	24,747
token	129,833,741	104,640,902	159,131,373	28,112,090
type	5,133,030	3,921,893	3,053,703	450,115
avg tokens in src	246,27	496,27	265,17	1124,82
avg tokens in tgt	12,43	22,33	29,97	11,22
avg sents in src	23,74	35,76	11,40	49,26
avg sents in tgt	1,46	2,23	1,57	1,00

In the case of HVG⁶ and index.hu⁷, the body of the articles taken from the daily online newspaper, as well as the corresponding leads, representing the summaries. We have built two corpora from them. In the first version, we used only the HVG documents. In the second version (H+I corpus) we merged the HVG and the index.hu articles. In the case of MARCELL, we used the legal documents as source and each of these have one short sentence topic description that we used for target summary.

A BERT model has a maximum 512 sequence length (after BERT subword tokenization). Therefore in our research we used only the online daily articles and

⁶<https://www.hvg.hu>

⁷<https://www.index.hu>

its leads, because the articles of the weekly newspaper (HVG) are much longer. In the case of MARCELL, the average sentence length is 1124,82, which is much longer than 512, but the median is: 340, which is short enough for this task.

We did three different tasks. In the first two task, we used the HVG and MARCELL corpora on their own, without any cleaning and normalizing processes. In the third task, we merged the HVG and the index.hu corpora, and we also made cleaning processes on it. The cleaning and normalizing aspects are as followed:

- removed the long ($500 < \text{token}$) documents from the corpora
- removed the short ($5 > \text{token}$) documents from the corpora
- removed documents, that articles were shorter than its' lead (e.g., See Table 7)
- removed irrelevant articles or text parts: e.g. "Follow us on facebook", "Edited: [NAME]", "Click for more details", "Start a Quiz", etc.

5. Pretrained language models

In our abstractive summarization experiments we used 4 different kinds of pre-trained BERT models: huBERT, huBERT Wikipedia cased, HILBERT, BERT-Base-Multilingual-Cased.

huBERT [15] is the state-of-the-art Hungarian cased (not lowercased) BERT-base model that trained on Webcorpus 2.0⁸ (9 billion token) with 110 million parameters, 12-layer, 768-hidden, 12-heads.

huBERT Wikipedia cased [15] is a Hungarian cased BERT-base model that trained on Hungarian Wikipedia (170 million token) with 110 million parameters, 12-layer, 768-hidden, 12-heads.

HILBERT [3] is a Hungarian cased BERT-large model that trained on NYTK v1 corpus (3.7 billion token) with 340 million parameters, 24-layer, 1024-hidden, 16-heads.

BERT-Base-Multilingual-Cased⁹ is a cased BERT-base model that trained on 104 languages of Wikipedia, with 110 million parameters, 12-layer, 768-hidden, 12-heads.

In the extractive summarization experiments, we used the 2 kinds of huBERT, the multilingual and 4 kinds of ELECTRA models. In the case of ELECTRA, there were no pretrained models for Hungarian, thus we did experiments to create them.

For training ELECTRA models, we have used three different corpora:

⁸<https://hlt.bme.hu/en/resources/webcorpus2>

⁹<https://github.com/google-research/bert/blob/master/multilingual.md>

- Hungarian Wikipedia (wiki): 13,098,808 segments; 163,772,783 tokens;
- NYTK corpus (NYTK): 283,099,534 segments; 3,993,873,992 tokens; (contains Hungarian Wikipedia)

The vocabulary size was 64,000. This relatively large size was deemed justified in view of the agglutinative nature of the rich morphological system of Hungarian resulting in an almost open-ended stock of wordforms. The ratio of number of subword tokens per surface words was 1.15707, which can be considered good.

To train ELECTRA models, we used the code¹⁰ published by Google. We trained six different ELECTRA models for Hungarian, which we named as HIL-ELECTRA (HILANCO¹¹ ELECTRA):

- HIL-ELECTRA small wiki: trained on Hungarian Wikipedia. Training time: ~5 days
- HIL-ELECTRA small NYTK: trained on Hungarian Research Centre for Linguistics corpus v1. Training time: ~7 days
- HIL-ELECTRA base wiki: trained on Hungarian Wikipedia. Training time: ~5 days
- HIL-ELECTRA base NYTK: trained on Hungarian Research Centre for Linguistics corpus v1 corpus. Training time: ~7 days

In Table 2, we can see the training hyper-parameters of the ELECTRA small and base models.

Table 2. The hyper-parameters of the training of the ELECTRA models.

	Learning rate	Weight decay	Layers	Embedding size	Batch size	Training step
small	5e-4	0.01	12	128	80	1 million
base	5e-4	0.01	12	768	2	1 million

Each model was trained on 1 single GeForce RTX 2080 Ti type video card. The training took about 5-7 days. The run time is also affected by dictionary size, it can be accelerated with a smaller dictionary.

6. Experiments

Using the pretrained language models, we fine-tuned summarization models for Hungarian. The first step of our research was to pre-process the original text.

¹⁰<https://github.com/google-research/electra>

¹¹<https://hilanco.github.io>

The articles and their leads were tokenized with the e-magyar¹² tokenizer module, the quntoken [11] tool. Then, we converted the tokenized text to JSON format for the summarization system. The system then inserts two special elements, the first one indicating the beginning of the text and the other one marks the sentence boundaries. After pre-processing, we trained different summarization models.

We trained and compared the following models in the different tasks:

- Abstractive summarization:
 - BERT Base Multilingual Cased (multi-BERT)
 - huBERT Wikipedia cased (huBERT wiki)
 - huBERT (huBERT web)
 - HILBERT
- Extractive summarization:
 - BERT Base Multilingual Cased
 - huBERT Wikipedia cased
 - huBERT
 - HIL-ELECTRA base Hungarian Wikipedia (HIL-ELECTRA base wiki)
 - HIL-ELECTRA base NYTK (HIL-ELECTRA base NYTK)
 - HIL-ELECTRA small Hungarian Wikipedia (HIL-ELECTRA small wiki)
 - HIL-ELECTRA small NYTK (HIL-ELECTRA small NYTK)

To train abstractive and extractive models, we used the PreSumm [8] tool¹³. In the Table 3 you can see the differences of BERT-base and BERT-large training (fine-tuning) hyper-parameters and characteristics. All other hyperparameters were set to default of experiments of Yang et al. [8].

Table 3. Differences of BERT-base and BERT-large training hyper-parameters.

	learning rate	lr decrease	batch size	hardware
BERT-base	1e-03	0.1	20	4x GeForce RTX 2080
BERT-large	5e-05	0.02	10	4x Tesla V100

In our experiments, the larger the corpus the more training steps are required. Accordingly, the following training steps were used:

- Abstractive summarization
 - MARCELL: 50,000

¹²<https://e-magyar.hu>

¹³<https://github.com/nlpyang/PreSumm>

- HVG: 200,000
- H+I: multi and huBERT: 600,000; HILBERT: 800,000
- Extractive summarization: 50,000

7. Results and Evaluation

The ROUGE [7] method was used for evaluation. ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a coverage-based method based on BLEU metrics used in machine translation. ROUGE itself contains several methods, of which ROUGE-1, ROUGE-2 and ROUGE-L methods were used for our measurements. ROUGE-1 is a unigram, while ROUGE-2 is a bigram coverage calculation algorithm. ROUGE-L examines the longest common word sequence at the paragraph and sentence level.

In Table 4, 5 and 6, we can see the ROUGE recall results of our abstractive and extractive experiments. Since the HILBERT model needs huge amount of resources, we used it only in the experiment of H+I and in this task we did not use the huBERT wiki, because the huBERT web contains the wiki.

Table 4. ROUGE recall results of abstractive summarization of MARCELL, HVG and H+I tasks.

		ROUGE-1	ROUGE-2	ROUGE-L
MARCELL	multi	87.37	77.38	84.97
	huBERT wiki	89.37	79.91	86.14
	huBERT web	89.64	80.29	86.46
HVG	multi	47.02	19.72	39.29
	huBERT wiki	49.49	21.62	41.46
	huBERT web	51.47	23.27	43.82
H+I	multi (550k)	51.85	23.22	43.45
	huBERT web (450k)	57.07	26.97	48.28
	HILBERT (800k)	44.98	14.22	37.06

Table 5. ROUGE F1 results of the first generated sentence of MARCELL tasks.

	ROUGE-1	ROUGE-2	ROUGE-L
multi	72.99	65.38	71.53
huBERT wiki	74.23	66.56	72.90
huBERT web	75.85	68.35	74.61

In Table 4, you can see only the recall results, because of the number of generated sentences are more than the reference sentences, the precision of ROUGE cannot show evaluable performance. In the case of MARCELL task, the result

Table 6. ROUGE recall results of extractive summarization.

	ROUGE-1	ROUGE-2	ROUGE-L
multi-BERT	48.58	20.12	39.42
huBERT wiki	48.86	20.45	39.60
huBERT web	49.45	21.07	40.14
ELECTRA base wiki	48.83	20.37	39.53
ELECTRA base NYTK	49.04	20.53	39.76
ELECTRA small wiki	49.02	20.52	39.74
ELECTRA small NYTK	49.04	20.53	39.76

should be exactly one sentence, thus, in Table 5, you can see the F1 scores of the original (orig) and the first generated sentence of the models.

In the case of abstract summarization (see Table 4), the integration of Hungarian models achieved higher performance than the multilingual model. In all cases, the huBERT web gained the best results. As you can see in Table 4, adding index.hu data and applying cleaning methods leads to a performance increase of about 6%.

In the case of H+I, we can see the steps numbers in parentheses that achieved the best results. According to the steps, the huBERT at 450,000 step achieved the best results, much earlier than the other models. In the case of HILBERT, we did not achieve the theoretical optimum, because the rouge values are increasing continuously. As we can see in Table 4, the performance of HILBERT is much lower than the other models. Since the HILBERT is BERT-large, with twice as many parameters as a BERT-base, the model is more robust and the fine-tuning is more difficult. After 47 failed experiments, we could find a set of hyperparameters (see Table 3) that the model could converge with. We believe, that the HILBERT could gain higher results, but we need more experiments to find the best set of hyperparameters to achieve the highest result.

In the case of extractive summarization (see Table 6), all Hungarian models have scored higher than the multi-BERT. As was expected, the Hungarian huBERT web scored the best results. The interesting fact in the results is that our ELECTRA models, which were trained with modest compute, could achieve higher results than huBERT wiki. The ELECTRA models could not outperform huBERT web, just as we expected it would not, after all the huBERT web model was trained on over 9 billion tokens. The result that our ELECTRA models outperformed the huBERT wiki model is significant as the ELECTRA models have much fewer parameters than the BERT base models and can be trained on a single GPU, ideally, within as little as 5 days. It should be noted that training time can be even shorter if the dictionary size is reduced.

We can see some samples in Table 7–10 (see Appendix) which were generated by our abstractive summarization models. Analyzing the samples, we can notice some common features of our models. When the article is long (see Table 8 and Table 9), all of our models extract phrases from the original article, then combine them to

generate new sentences. It is similar to extractive models, the difference is that our extractive models choose full sentences from the article and after ranking, give them as results to the user. Generally, the sentences produced by the abstractive models are mostly grammatically correct. All the models generate several sentences, but by the end they “run out” and may leave sentence fragments (see Table 8).

When the article is short (see Table 7 and Table 10), the models show their real abstractive feature, which is to generate passages that the original article did not contain. But in this case, there is too little information in the original article, thus the performance of the output is lower.

Following the samples, we can see the disadvantages of the automatic evaluation metric, such as ROUGE, as well as the problem of using lead as summarization. The ROUGE metric shows only how the generated output is similar to the lead. However, often the function of the lead is to attract attention and not to summarize. In Sample 1. (see Table 7), the article is about damages caused by storms and the payments by the insurers. The lead was only about the insurers it did not even mention the storm and the damages, whereas our models described both topics. This is one of the reasons that in the results (see Table 4) we can see only about 50% recall results.

For more examples visit our demo site¹⁴.

8. Summary

It is concluded that we have created various text summarization tools for the Hungarian language. For building the summarization models, we used different kinds of BERT-based models. For abstractive models, we used the pretrained multilingual cased BERT model as well as the Hungarian monolingual huBERT base and the HILBERT large models.

For extractive summarization, besides the BERT models, we trained our own ELECTRA models. To fine-tune the BERT-based models for summarization tasks, we used the PreSumm tool. The results show that the monolingual Hungarian models outperformed the multilingual model in all cases. The huBERT web model that was trained on 9 billion words could gain the best results both in abstractive and in extractive tasks. Another important result is that our ELECTRA models were trained with less computational demand and they have much less parameters, could gain better results than the huBERT wiki. Another important point of view is that the ELECTRA models are much smaller than the BERT models, which is important for the end users.

This is the first automatic abstractive and extractive text summarization tool for Hungarian that is based on BERT-based neural network technology.

In the future, we would like to experiment with autoregressive methods, such as BART or PEGASUS.

¹⁴<http://nlpg.itk.ppke.hu/projects/summarize>

References

- [1] A. CELIKYILMAZ, A. BOSSELOT, X. HE, Y. CHOI: *Deep Communicating Agents for Abstractive Summarization*, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 1662–1675, DOI: <https://doi.org/10.18653/v1/N18-1150>.
- [2] K. CLARK, M.-T. LUONG, Q. V. LE, C. D. MANNING: *ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators*, in: International Conference on Learning Representations, 2020.
- [3] Á. FELDMANN, R. HAJDU, B. INDIG, B. SASS, M. MAKRAI, I. MITTELHOLCZ, D. HALÁSZ, Z. G. YANG, T. VÁRADI: *HILBERT, magyar nyelvű BERT-large modell tanítása felhő környezetben*, XVII. Magyar Számítógépes Nyelvészeti Konferencia (2021), pp. 29–36.
- [4] S. GEHRMANN, Y. DENG, A. RUSH: *Bottom-Up Abstractive Summarization*, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 4098–4109, DOI: <https://doi.org/10.18653/v1/D18-1443>.
- [5] I. GOODFELLOW, J. POUGET-ABADIE, M. MIRZA, B. XU, D. WARDE-FARLEY, S. OZAIR, A. COURVILLE, Y. BENGIO: *Generative Adversarial Nets*, in: Advances in Neural Information Processing Systems, ed. by Z. GHAFRAMANI, M. WELLING, C. CORTES, N. LAWRENCE, K. Q. WEINBERGER, vol. 27, Curran Associates, Inc., 2014, pp. 2672–2680.
- [6] M. LEWIS, Y. LIU, N. GOYAL, M. GHAVININEJAD, A. MOHAMED, O. LEVY, V. STOYANOV, L. ZETTLEMOYER: *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2020, pp. 7871–7880.
- [7] C.-Y. LIN: *ROUGE: A Package for Automatic Evaluation of Summaries*, in: Text Summarization Branches Out, Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 74–81.
- [8] Y. LIU, M. LAPATA: *Text Summarization with Pretrained Encoders*, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China: Association for Computational Linguistics, 2019, pp. 3730–3740.
- [9] Y. LIU, I. TITOV, M. LAPATA: *Single Document Summarization as Tree Induction*, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 1745–1755, DOI: <https://doi.org/10.18653/v1/N19-1173>.
- [10] M. MIHÁLTZ: *OpinHu: online szövegek többnyelv véleményelemzése*, VII. Magyar Számítógépes Nyelvészeti Konferencia (2010), ed. by A. TANÁCS, V. VARGA, V. VINCZE, pp. 14–23.
- [11] I. MITTELHOLCZ: *emToken: Unicode-képes tokenizáló magyar nyelvre*, XIII. Magyar Számítógépes Nyelvészeti Konferencia (2017), ed. by A. TANÁCS, V. VARGA, V. VINCZE, pp. 61–69.
- [12] T. MOLNÁR LENGYELNÉ: *Automatic abstract preparation*, 10th International Conference On Information: Information Technology Role in Development (2010), pp. 550–561.
- [13] R. NALLAPATI, F. ZHAI, B. ZHOU: *SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents*, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI’17, San Francisco, California, USA: AAAI Press, 2017, pp. 3075–3081.

- [14] S. NARAYAN, S. B. COHEN, M. LAPATA: *Ranking Sentences for Extractive Summarization with Reinforcement Learning*, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 1747–1759, doi: <https://doi.org/10.18653/v1/N18-1158>.
- [15] D. M. NEMESKEY: *Egy emBERT próbáló feladat*, in: XVI. Magyar Számítógépes Nyelvészeti Konferencia, Szeged: Szegedi Tudományegyetem, 2020, pp. 409–418.
- [16] D. M. NEMESKEY: *Natural Language Processing Methods for Language Modeling*, PhD thesis, Eötvös Loránd University, 2020.
- [17] R. PAULUS, C. XIONG, R. SOCHER: *A deep reinforced model for abstractive summarization*, in: Proceedings of the 6th International Conference on Learning Representations, Vancouver, Canada, 2018.
- [18] M. SCHUSTER, K. NAKAJIMA: *Japanese and Korean voice search*. In: ICASSP, IEEE, 2012, pp. 5149–5152, ISBN: 978-1-4673-0046-9.
- [19] A. SEE, P. J. LIU, C. D. MANNING: *Get To The Point: Summarization with Pointer-Generator Networks*, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, Canada: Association for Computational Linguistics, July 2017, pp. 1073–1083, doi: <https://doi.org/10.18653/v1/P17-1099>.
- [20] T. VÁRADI, S. KOEVA, M. YAMALOV, M. TADIĆ, B. SASS, B. NITOŃ, M. OGRODNICZUK, P. PEŽIK, V. BARBU MITITELU, R. ION, E. IRIMIA, M. MITROFAN, V. PÁIŞ, D. TUFIŞ, R. GARABÍK, S. KREK, A. REPAR, M. RIHTAR, J. BRANK: *The MARCELL Legislative Corpus*, English, in: Proceedings of the 12th Language Resources and Evaluation Conference, Marseille, France: European Language Resources Association, May 2020, pp. 3761–3768, ISBN: 979-10-95546-34-4.
- [21] A. VASWANI, N. SHAZEER, N. PARMAR, J. USZKOREIT, L. JONES, A. N. GOMEZ, Ł. KAISER, I. POLOSUKHIN: *Attention is All you Need*, in: Advances in Neural Information Processing Systems 30, ed. by I. GUYON, U. V. LUXBURG, S. BENGIO, H. WALLACH, R. FERGUS, S. VISHWANATHAN, R. GARNETT, Curran Associates, Inc., 2017, pp. 5998–6008.
- [22] Z. YANG, Z. DAI, Y. YANG, J. CARBONELL, R. R. SALAKHUTDINOV, Q. V. LE: *XLNet: Generalized Autoregressive Pretraining for Language Understanding*, in: Advances in Neural Information Processing Systems, ed. by H. WALLACH, H. LAROCHELLE, A. BEYGEZIMER, F. D’ALCHÉ-BUC, E. FOX, R. GARNETT, vol. 32, Curran Associates, Inc., 2019, pp. 5753–5763.
- [23] Z. G. YANG, A. PERLAKI, L. J. LAKI: *Automatikus összefoglaló generálás magyar nyelvre BERT modellel*, XVI. Magyar Számítógépes Nyelvészeti Konferencia (2020), pp. 343–354.
- [24] J. ZHANG, Y. ZHAO, M. SALEH, P. LIU: *PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization*, in: Thirty-seventh International Conference on Machine Learning, 2020.
- [25] X. ZHANG, M. LAPATA, F. WEI, M. ZHOU: *Neural Latent Extractive Document Summarization*, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 779–784, doi: <https://doi.org/10.18653/v1/D18-1088>.
- [26] Q. ZHOU, N. YANG, F. WEI, S. HUANG, M. ZHOU, T. ZHAO: *Neural Document Summarization by Jointly Learning to Score and Select Sentences*, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 654–663, doi: <https://doi.org/10.18653/v1/P18-1061>.

Appendices

Examples of abstractive models

Table 7. 1. example of abstractive models (HVG experiment).

<p>Article</p> <p>Ha a legjobb képes mémek is érdeklik , ide kattintson . Az internetadóról ebben a cikkünkben olvashat bővebben . Ha arra kíváncsi , mit gondol róla a világ , itt megtalálja .</p>
<p>Lead</p> <p>Az internetadóról szóló híreket olvassa el a hvg.hu-n . A magyar internetadóról szóló cikkünkre kattintva olvashatjátok a legfrissebb mémeket , amelyekről itt találjátok a választ . Íme , a legjobb mémek , a mémek és a mémek is . Ha nem akarunk lemaradni a neten , itt találjátok meg a legjobb mémeket . A Az internetadóról szóló mémek mellett az internetadóról is megemlékezett az internetadóról a mémgyár . A mémek mellett a mémek is felkerültek a netadóra . Íme , a legjobbak . Videó az internetadóról . A legjobbak között van az internetadó , a legjobb képes mémek is . a legjobbak között . Videós összefoglaló . hu</p>
<p>huBERT web</p> <p>Az internetadóról szóló mémek mellett az internetadóról is megemlékezett az internetadóról a mémgyár . A mémek mellett a mémek is felkerültek a netadóra . Íme , a legjobbak . Videó az internetadóról . A legjobbak között van az internetadó , a legjobb képes mémek is . a legjobbak között . Videós összefoglaló . hu</p>
<p>huBERT wiki</p> <p>Az internetadóról szóló híreket olvassa el a hvg.hu-n . A magyar internetadóról szóló cikkünkre kattintva olvashatjátok a legfrissebb mémeket , amelyekről itt találjátok a választ . Íme , a legjobb mémek , a mémek és a mémek is . Ha nem akarunk lemaradni a neten , itt találjátok meg a legjobb mémeket . A</p>
<p>multi-BERT</p> <p>Az internetadóról szóló cikkünkben olvashat bővebben az internetadóról . Mutatjuk , mire érdemes figyelni az internetadót . Íme a legújabb adatok . Az internetadóról ebben a cikkben olvashat bővebben . A legjobb képes mémek is érdekesek . És persze nem is gondolnánk , hogy mit gondolnak róla . Írásunkban megmutatjuk , mit gondol a világról . Írja a cikkünket . Teszteljétek magatokat ezzel a kví .</p>

Table 8. 2. example of abstractive models (HVG experiment).

<p>Article</p> <p>Az idei júliusi viharokat követően csaknem hétezer esetet regisztráltak a biztosítók . A két héttel ezelőtti viharokban sok ingatlan sérült meg a fővárosban , Pest megyében , illetve Somogy megyében . Különösen a Balaton déli partja volt érintett , s sokan csak napokkal később fedezték fel , hogy nyaralójuk károsodott . Több társaságnál is arról számoltak be , hogy átcsoportosították kárfelmérőiket a különösen érintett területekre , a szakemberek a kárszempléket elvégezték , s már a vihar másnapján megkezdődtek a kárkifizetések - írja a portfolio.hu . Magyarországon a legtöbb , ingatlanokkal kapcsolatos káresemény a nyári , május- augusztusi hónapokban történik . A biztosítókhoz 2010-2016 között az évnek ebben a periódusában összesen 988 599 kárbejelentés érkezett , amelyekre több mint 73 milliárd forintot fizettek ki a társaságok . A nyári viharok átlagosan mintegy 115 ezer forint összegű kárt tesznek a biztosított ingatlanokban és egyéb javakban , de e téren igen nagy a szórás : sok a kis összegű , úgynevezett „ bagatell-kár ” , de előfordulnak 10 milliós nagyságrendű káresemények is .</p>
<p>Lead</p> <p>2010 és 2017 között , a május-augusztusi időszakokban csaknem egymillió kárt térítettek a biztosító társaságok . A kifizetett kárösszeg meghaladja a 73 milliárd forintot - tette közzé a Magyar Biztosítók Szövetsége (MABISZ) .</p>
<p>huBERT web</p> <p>Az idei júliusi viharokat követően csaknem hétezer esetet regisztráltak a biztosítók . A viharokban sok ingatlan sérült meg , s sokan csak napokkal később fedezték fel , hogy nyaralójuk károsodott . Több biztosítónál is arról számoltak be , hogy átcsoportosították kárfelmérőiket a különösen érintett területekre , s már a vihar másnapján megkezdődtek a kárkifizetések - írja a portfolio.hu . A viharok átlagosan mintegy 115 ezer forint összegű kárt tesznek a biztosított ingatlanokban és egyéb javakban , de előfordulnak 10 milliós nagyságrendű káresemények is . Magyarországon a legtöbb , ingatlanokkal kapcsolatos káresemény a nyári , május- augusztusi időszakban .</p>
<p>huBERT wiki</p> <p>Az idei júliusi viharokat követően csaknem hétezer esetet regisztráltak a biztosítók . A Balaton déli partjának volt érintett , s sokan csak napokkal később fedezték fel , hogy nyaralójuk károsodott . Több társaságnál is arról számoltak be , hogy átcsoportosították kárfelmérőiket a különösen érintett területekre , a szakemberek a kárszempléket elvégezték , s már a vihar másnapján megkezdődtek a kárkifizetések . Magyarországon a legtöbb , ingatlanokkal kapcsolatos káresemény a nyári , május- augusztusi hónapokban történik .</p>
<p>multi-BERT</p> <p>Az idei júliusi viharokban sok ingatlan sérült meg a fővárosban , Pest megyében , illetve Somogy megyében . Különösen a Balaton déli partja volt érintett , s sokan csak napokkal később fedezték fel , hogy nyaralójuk károsodott . Több társaságnál is arról számoltak be , hogy átcsoportosították kárfelmérőiket a különösen érintett területekre , a szakemberek pedig a kárszempléket elvégezték , s már a vihar másnapján megkezdődtek a kárkifizetések .</p>

Table 9. 3. example of abstractive models (H+I experiment).

<p>Article</p> <p>Kedd délután a rendőrség megerősítette az Index nek , hogy reggel elfogtak a rendőrök egy férfit Budapesten , aki lőfegyverrel a kezében álldogált a Nagyvárad téren a 24-es villamos megállójában . A lapnak több szemtanú is beszámolt az esetről , a rendőrség pedig azt közölte , hogy kedd reggel 7 óra 12 perckor bejelentést kaptak arról , hogy a Nagyvárad tér 1. szám alatti villamosmegállóban egy fegyver van egy ember kezében . Lövés nem dördült , és senki nem sérült meg , a járőrök percekben belül a helyszínre érkeztek , és intézkedés alá is vontak egy 49 éves budapesti férfit . P. Lászlónál egy gáz-riasztó fegyvert találtak , ezért garázdaság és lőfegyverrel kapcsolatos szabálysértés miatt előállították , és eljárást kezdeményeztek vele szemben .</p>
<p>Lead</p> <p>Lövés nem dördült , a férfit elfogták .</p>
<p>huBERT web</p> <p>Garázdaság és lőfegyverrel kapcsolatos szabálysértés miatt előállították P. Lászlót , akinél gáz-riasztó fegyvert találtak , ezért garázdaság és lőfegyverrel kapcsolatban szabálysértés miatt eljárást kezdeményeztek vele szemben . Lövés nem dördült , senki nem sérült meg , de a rendőrök percekben belül elfogták a férfit . P. Lászlónál egy gáz-riasztó fegyvert is találtak . Garázdaság miatt előállították , és eljárást kezdeményeztek ellene . Garázdaság és lő fegyverrel kapcsolatos szabálysértése miatt előállították . A férfi ellen eljárást kezdeményeztek , és eljárás indult ellene . Videó</p>
<p>HILBERT</p> <p>A férfi fegyverrel a kezében sétált a 24-es villamosmegállóban , a rendőrök percekben belül elfogták , és kezdeményezik előzetes letartóztatásukat . A rendőrség eljárást indított vele szemben , és eljárást indítottak vele szemben . A férfi ellen eljárás indult . A rendőrség szerint egy gáz-riasztó fegyvert is találtak a rendőrök , és elrendelték az előzetes letartóztatását . Az MTI úgy tudja , hogy a férfi egy gáz-riasztó fegyverrel is felszerelkezett a 24-es villamos megállójában . Videó is van a helyszínről , a rendőrség a hvg.hu-nak arról számolt be , hogy kedd reggel fogtak el egy férfit , aki lőszerral a rendőrök elől menekülve próbált meg egy embert a 24-es villamoson . hu hu</p>
<p>multi-BERT</p> <p>Kedd reggel elfogták a rendőrök azt a férfit , aki lőfegyverrel a kezében álldogált a Nagyvárad téren a 24-es villamos megállójában . Lövés nem dördült , és senki nem sérült meg , a járőrök percekben belül a helyszínre érkeztek , és intézkedés alá is vontak egy 49 éves budapesti férfit . P. Lászlónál egy gáz-riasztó fegyvert találtak , ezért garázdaság és lőfegyverrel kapcsolatos szabálysértés</p>

Table 10. 4. example of abstractive models (H+I experiment).

<p>Article</p> <p>A Daily Mail Twitter-oldalán közzétett videó szerint most épp nem történt nagy baj , de ez könnyen történhetett volna másképp is .</p>
<p>Lead</p> <p>Valószínűleg nem gondolták át igazán ezek a fiatalok mit tesznek .</p>
<p>huBERT web</p> <p>A Daily Mail Twitter-oldalára került fel egy videó arról , ahogy az egyik brit autópályán egy motorcsónaknak csapódott . A videón jól látható , hogy a sofőrnek nem sok esélye van a balesetre , de a videón jól látszik , hogy nem is kell aggódnia . És nem is ez volt az első ilyen eset . A videóban rögzítették , ahogy a britek egy hatalmasat ugranak a fejük előtt . Az alábbi videón látható , hogyan történt a dolog , és hogyan lett volna ez a videóból . A videón látható .</p>
<p>HILBERT</p> <p>A Magyar Kétfarkú Kutya Párt szerint most már nem is történt nagy baj , de ez így is elég lett volna . nem ez volt az első eset , hogy nem lett komoly baj . „ Ez történt volna ” - reagált a párt . Videón az is látszik , hogy a párt szerint nem történt baj . Igaz , ez még így is így lett volna , ha nem is lett volna baj , az eredmény nem lett volna ennyire egyszerű .</p>
<p>multi-BERT</p> <p>A Daily Mail videója szerint nem történt nagy baj . A videó könnyen előfordulhatott volna , hogy egy hatalmas bajba került . Nem kellett volna . Videó . Szerencsére nem történt baj , de ez könnyen történhetett volna másképp is . De nem . Videón . Mutatjuk . Szerencsésen volt . Videót . És . Munkatársunk . Sőt . Mutatja !Videó !Szavazzon !! A hvg.hu hu . A</p>

