Proceedings in
Informatics and Information Technologies

Student **R**esearch **C**onference **2020**

Ján Lang and Rastislav Bencel (Eds.)

Keynote by Markus Rupp

Proceedings in
Informatics and Information Technologies

IIT.SRC 2020
Student Research Conference

Ján Lang and Rastislav Bencel
(Eds.)

IIT.SRC 2020:
Student Research Conference

16th Student Research Conference
in Informatics and Information Technologies
Bratislava, April 22, 2020
Proceedings

IEEE
Czechoslovakia Section

acm
Slovakia Chapter

STU
FIIT

SLOVAK UNIVERSITY OF
TECHNOLOGY IN BRATISLAVA
FACULTY OF INFORMATICS
AND INFORMATION TECHNOLOGIES

Proceedings in
Informatics and Information Technologies

IIT.SRC 2020
Student Research Conference

Editors
Ján Lang and Rastislav Bencel
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 2
842 16 Bratislava, Slovakia

Visit IIT.SRC on the Web: http://iit-src.stuba.sk/2020/

Executive Editors: Rastislav Bencel

Copy Editor: Ján Lang
FIIT STU

FIIT STU
Ilkovičova 2, Bratislava, Slovakia
Published in Slovak University of Technology in Bratislava in SPEKTRUM STU Publishing
2021
ISBN 978-80-227-5075-2

Preface

The criteria of universities quality appraisal are primarily focused on their research activities. One of the crucial indicators is the research results publication. So, it is obvious a university activity setting – not except our faculty – the Faculty of Informatics and Information Technologies of the Slovak University of Technology in Bratislava – oriented to the closest possible research and education connection. There is a natural effort of researchers to name research problems, analyse existing solutions and approaches in close confrontation with the design and implementation of their own solution. State of the art in a domain, as well as their own approach in solving problems naturally together with evaluation, are usually part of the final work of students at all three levels of university study. Joint research activities of students and their teachers, researchers are also documented by other publications. Partial results are often published at thematically related conferences in the form of research articles. The research orientation degree of a student is directly proportional to the university study degree. The highest rate is expected for students at the doctoral level of study, which is primarily focused on research in the field. The lowest but not negligible is also clear at the bachelor study level of university study. Bachelor students also use research techniques although their final theses are more of an implementation nature. The documented research output, exposed in the community of stakeholders, presupposes factual criticism in the form of a review. Assessment of papers is a natural process of selecting the relevance of the issue and the solution offered at conferences as well. The IIT.SRC Student Scientific Conference is an opportunity for students to experience the feeling of attending the conference, to experience the flavour of an international scientific conference and to receive feedback on published research results. The conference is an opportunity for these comments to lead to a real improvement in the scientific and research nature of the article. That is why it was considered particularly important to prepare at least two or more reviews for each article. Both internal researchers and external researchers were involved in the assessment, and finally, the students themselves, who had at least one such experience of publishing at another conference.

A total of 56 papers were submitted to the conference. At least two reviews per paper were applied. There were 32 program committee members and 16 additional reviewers. Finally, 47 papers were accepted and so the acceptance rate is 83.93%. The conference was attended by 75 authors - students at all three levels of study. Exactly there were 3 doctoral students, 25 master students, 18 bachelor students, and one high school student. Given the number of articles published in this collection, the experienced feeling and feeling in the context of the provided feedback has shifted the quality of articles so that they are published on other forums. Selected young researchers were supported by special travel grants that cover partially their travel expenses to conferences. The best paper awards were also presented. The conference was opened by Professor Markus Rupp from Vienna University of Technology, Austria, and his keynote entitled: Wireless Communications of the Future.

Bratislava, April 2020

Ján Lang and Rastislav Bencel

Conference Organisation

The 16th Student Research Conference in Informatics and Information Technologies (IIT.SRC), held on April 22, 2020 in Bratislava, was organised by the Slovak University of technology (and, in particular, its Faculty of Informatics and Information Technologies) in Bratislava. All from FIIT STU in Bratislava, Slovakia.

Steering Committee

Valentino Vranić (Steering Committee Chair)
Katarína Jelemenská
Ivan Kotuliak
Peter Lacko
Lukáš Šoltés

Program Committee

Ján Lang (Program Chair)
Rastislav Bencel (Program Vice-Chair)
Vanda Benešová
Mária Bieliková
Miroslav Blšták
Pavel Čičák
Pavol Helebrandt
Andrea Hrčková
Ján Hudec
Ladislav Hudec
Vladimír Janiš
Katarína Jelemenská
Peter Kapec
Ivan Kotuliak
Tomáš Kováčik
Tibor Krajčovič
Štefan Krištofík
Eduard Kuric
Peter Lacko
Ján Laštinec
Fedor Lehocki
Dominik Macko
Irina Malkin Ondik
Ivan Polášek
Michal Ries
Viera Rozinajová
Jakub Šimko
Matúš Tomlein
Peter Trúchly
Jozef Tvarožek
Petra Vrablecová
Valentino Vranić

Local Organizing Committee

Zuzana Marušincová (Local Chair)
Zuzana Horniaková
Ľubica Jančaťová
Zuzana Macková
Katarína Mršková
Ľubica Palatinusová
Branislav Steinmüller
Roman Stovíček

Additional Reviewers

Peter Bakonyi
Tomáš Frťala
Marek Galinski
Štefan Grivalský
Lukáš Hudec
Marek Jakab
Radovan Kohút
Matej Kompánek
Kristián Košťál
Vladimír Kunštár
Miroslav Laco
Lukáš Mastiľak
Ján Skalný
Waheedullah Sulaiman Khail
Martin Tamajka
Michal Valiček

Supporting Professional Societies and Foundations

IEEE
Czechoslovakia Section

Industry sponsors

AT&T

eset

QBSW
QUALITY BUSINESS SOFTWARE

SIEMENS
Healthineers

SOFTEC
BUSINESS & IT INTEGRATION

Wireless Communications of the Future

Markus Rupp

Vienna University of Technology, Austria
markus.rupp@tuwien.ac.at

Abstract. The UN predicts cities to increase dramatically in size in the coming years. As more and more shares of the human population will live in cities, such new cities face plenty of novel challenges: public transportation next to individual traffic will increase, requiring more efficient planning and scheduling as well as safety regulations. More energy demands require efficient control of limited and expensive resources. All of these demands require a novel wireless infrastructure. As a consequence of such dramatic changes that we will face in the coming two decades, the constraints on wireless networks will also need to be changed considerably. My presentation will exemplarily show various research activities in our labs concerning NOMA techniques that are intended to solve many of such new problems.

Optimization of LoRa Devices Communication for Applications in Healthcare

Alexander Valach* [0000-1111-2222-3333]

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
alexander.valach@gmail.com

Abstract. Number of embedded devices connected to the Internet is increasing rapidly. Many of these devices have never meant to be connected to the Internet. Especially, wireless ones that face a communication parameters selection problem to avoid collisions due to inability to listen before a transmission, limited power supply or duty-cycle restrictions. Another problem arises when devices are operating in densely populated areas where license-free ISM band is jammed with many different technologies. Therefore, a network controller does not have exact information about channel congestion and has to determine it for each node from observations. Among the most promising solutions for long-distance and low-power IoT networks is LoRa. In this paper, we propose a modified LoRa architecture using energy-wise LoRa@FIIT protocol to optimize communication parameters selection and ensure QoS for application in healthcare devices, where critical information must be properly delivered and acknowledged.

Keywords: LoRa, LoRaWAN, LoRa@FIIT, IoT, Health Care, Low Power

1 Introduction

IoT end devices are usually small embedded computers that can be wearable. They have less computational power, less memory capacity, limited power supply and less bandwidth than conventional IP and Ethernet network devices. Those factors should be considered during protocol or network architecture design.

Vendors, like Ericsson or Cisco, estimate that by the end of 2020 there will be approximately 28 billion devices connected to the Internet [1]. To meet the industrial requirements, modern IoT networks should be scalable and automatically respond to network changes, e.g. congestion, duty-cycle limitations, interference from other technologies and interference within the same technology when two or more devices

* Master study programme in field: Computer Engineering
Supervisor: Dr. Dominik Macko, Institute of Computer Engineering and Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

transmit at the same time using the same communication parameters (CP), e.g. spreading factor (SF) and power.

IoT devices have potential utilization in healthcare. The typical scenario assumes a patient wearing a battery supplied device that would measure a blood pressure, a heart rate or an oxygen saturation. Based on collected data, recommendations would be given, or early disease detection might be possible, e. g. detection of heart arrhythmia using heart rate long-term measurements.

Another constraint of communication in license-free bands is a duty-cycle. It is defined as the maximum percentage of time during which an end-device can transmit on a selected channel [2]. In Europe, duty-cycle is regulated by ETSI EN300.220 and is limited to less than 36s time on air per hour for each node [3].

The rest of the paper is organized in the following way. Section 2 summarizes latest research in optimization of LoRa technology. Section 3 covers a reliable LoRa network architecture proposal using LoRa@FIIT and STIoT protocol with QoS. In Section 4, we propose a network architecture to evaluate scalability and reliability of a proposed network design. Section 5 summarizes the paper.

2 Related works

One of the toughest challenges in LoRa networks is a collision-free communication in densely populated areas. In recent years, there has been an intensive research in this field [4][5][6].

A lot of work has been done in the field of computing an optimal channel selection in the network server or end nodes [7]. Specifically, in the problem of end nodes energy optimization [8][9] a channel selection [10], a collision-free communication in densely populated areas [11], or an optimization of communication in general [5]. In [12], a reinforcement learning (RL) proved to be a huge advantage even if non-stationary settings are present in the environment.

However, only the research in [13] has taken into account a dissemination, i.e. a process of distributing gained knowledge from the network server to the end nodes. All gateways are bound to duty-cycle limitations, since it is the key constraint in Lo-Ra networks. For a real-world use case with hundreds of nodes, a proper way and time to disseminate a network-wise model to end nodes is an important requirement.

Authors of [4] proposed a usage of Markov Decision Process to find optimal communication parameters from the end node perspective. However, as the results show in [13], it is important to find and test proposed solution in a real-world, where physical obstacles and interference are present.

In [13], the researchers have further presented a mathematical model aimed to help distribution of the network-wise configuration. It is very important that computation of the current network state is done in the network server, as there is a need to constantly update statistical model using the values of RSSI and SNR (or Sequence Number when using LoRa@FIIT protocol) from LoRa gateways.

To conclude, a network-wise approach instead of a node-wise approach has been used in [13]. Both in terms of a proper communication parameters selection and a message importance. A message importance can be considered a form of Quality of Service (QoS) as in [10][13]. The main idea behind this solution is to update nodes information based on the overall network congestion. Nodes prefer empty SFs or lower SF values to other SF values, as it means less time on air, which leads to lower probability of collisions [13]. The main goal was to increase network throughput in heterogenous IoT networks, which resemble a real-world scenario much more comparing to research, where a homogenous IoT network is presented [14][15]. Heterogenous IoT network assumes that nodes send messages on an event-driven basis or with different periods for each node and messages also have a different length. A message importance, resembling is a recurring factor, but to the authors' best knowledge there is no built-in mechanism for QoS.

To the authors' best knowledge, most of the research done in the field was performed with only in a form of a computer simulation as in [10][14]. They did not really consider an overhead needed for LoRa gateways to update CP on end devices. Except for [13], none of the above mentioned have been dealing with duty-cycle limitations and an efficient way to transfer a control plane data from a network server to an end device or a mathematical formula to find a proper timing when to update end node (EN) configuration.

3 A design of scalable and reliable IoT network

LoRa, also called LoRa PHY, is a physical layer modulation which uses proprietary technique called *chirp spread spectrum* [2][16]. This mechanism gives LoRa devices ability to communicate with minimal power consumption over long distances with high resistance to interference due to the usage of lower frequencies (sub-GHz band). LoRa uses license-free industrial, scientific and medical (ISM) bands and operates at 868 MHz in Europe. Frequencies may vary depending on the region [16].

Typical LoRa architecture consists of several end nodes (EN), wireless access points (AP), also called gateways, and a central point of management called a network server (NS). End nodes are usually embedded devices with ability to measure or evaluate certain characteristics and send them via LoRa technology. LoRa frames are received by single or multiple APs in the node's surrounding. A NS controls network traffic and is responsible for a duty-cycle computation, communication parameters selection, and a most adequate AP selection in case a message requires an acknowledgement and had been received by multiple APs.

It is not practical to send RAW data using LoRa technology [2][16]. For tasks like device addressing, encryption, and a message acknowledgment, a link layer protocol is required. The most popular and widely spread protocol is LoRaWAN [16], maintained by LoRa Alliance. It was designed to support roaming and device manipulation using so-called MAC commands. In this case a roaming is an important concept, because LoRaWAN networks are primarily owned by Internet Service Provider (ISP) and the service is provided to customers for certain fees [16].

LoRaWAN encrypts messages using AES-128 algorithm, which was not designed to be used in IoT devices and can be difficult to implement on memory and energy constrained nodes. To support a number of features, a LoRaWAN header is quite complex. For 1 B of useful information a device sends 29 B overhead [2][16]. The protocol was also not designed with QoS in mind and is impractical to send an emergency message. LoRaWAN networks use Adaptive Data Rate (ADR) to update communication parameters of end nodes if required [16]. The ADR algorithm simply compares the average RSSI value (computed from N latest measurements) with a predefined threshold [16][17]. If the value is below or above this threshold, a downlink message is scheduled and sent to the end node during the next uplink message.

3.1 Overcoming drawbacks of LoRaWAN protocol design

LoRa@FIIT is a link layer protocol created to overcome drawbacks of LoRaWAN protocol stack. It was not designed to support roaming, so we need to make sure that owner of the network is also owner of the devices. This could be considered a disadvantage when compared to LoRaWAN protocol. However, it is difficult to compare these protocols, since they were designed with different goals in mind. Not all use cases require use of roaming and LoRa@FIIT was designed to provide more efficient communication for such scenarios [17].

On the other hand, LoRa@FIIT has some advantages over LoRaWAN:

1. It uses optional acknowledgments for messages sent. Therefore, it is not mandatory to open a receiving window for an end device after each message is sent. The values for some message types could be customized depending on application needs.
2. Uses XXTEA algorithm [18] for encryption. The algorithm needs smaller blocks (64-bit alignment) and was designed for IoT devices with memory constraints in mind. It uses Diffie-Hellman [19] mechanism for key exchange during registration process or key update.
3. Energy efficient communication using shorter headers compared to LoRaWAN. For 1 B of payload there is only 12 B overhead. This leads to approximately 42% less battery and duty-cycle usage [17].
4. It uses sequence number field to achieve reliable delivery. Network server also benefits from it, as it can evaluate a pseudo-link quality to specific device.
5. It has built-in mechanism for QoS support called emergency message. It must be acknowledged and is transmitted using maximum power.

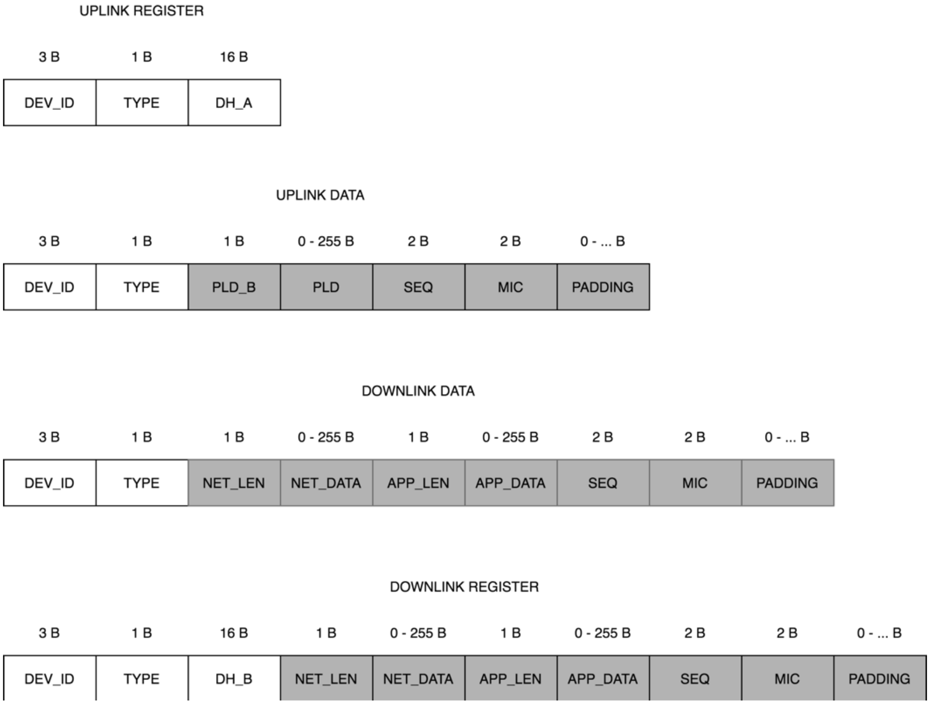
LoRa@FIIT differentiates several message types. There are *register messages* sent during initial EN registration process, *hello messages* which serve as a connection keepalive and health check mechanism, already mentioned *emergency messages* used for critical information transfer and *data messages* used for regular data transfer, e.g. battery level or oxygen saturation. Hello and data messages are presented in Fig. 1. Register and emergency messages are sent using full transmission power (14 dB) to ensure a successful delivery.

LoRa@FIIT networks do not use Message Queuing Telemetry Transport (MQTT) message broker for communication between APs and NS in comparison to known open-source LoRaWAN implementations [20]. It uses STIoT protocol [17], which is an acronym of Secured TCP for IoT.

Rather than using subscriber-publisher model, it focuses on reliable and secure information exchange between APs and the NS. This architecture is simple to deploy as it is client-server based. The further details about STIoT can be found in [17].

Both LoRa@FIIT and LoRaWAN are ALOHA-based protocols and do not have any collision detection or avoidance mechanism. This responsibility is transferred from end nodes to network server or even APs [13].

When updating a selection algorithm on an end node, the energy efficiency must be considered. Many IoT devices have very simple firmware implemented and adding additional overhead, when proper SF and power must be selected and statistical model updated, could lead to higher power consumption.



UPLINK REGISTER

3 B	1 B	16 B
DEV_ID	TYPE	DH_A

UPLINK DATA

3 B	1 B	1 B	0 - 255 B	2 B	2 B	0 - ... B
DEV_ID	TYPE	PLD_B	PLD	SEQ	MIC	PADDING

DOWNLINK DATA

3 B	1 B	1 B	0 - 255 B	1 B	0 - 255 B	2 B	2 B	0 - ... B
DEV_ID	TYPE	NET_LEN	NET_DATA	APP_LEN	APP_DATA	SEQ	MIC	PADDING

DOWNLINK REGISTER

3 B	1 B	16 B	1 B	0 - 255 B	1 B	0 - 255 B	2 B	2 B	0 - ... B
DEV_ID	TYPE	DH_B	NET_LEN	NET_DATA	APP_LEN	APP_DATA	SEQ	MIC	PADDING

Fig. 1: Different LoRa@FIIT packets along with header fields and their respective lengths. All grey fields are encrypted during transmission.

3.2 Effective selection of communication parameters

Current implementations of communication parameters selection (CPS) are mostly based on ADR algorithm [20] As a research have proven, it performs not very well when number of devices is increasing [21][13]. It also strongly depends on threshold values that has to be pre-calculated or updated according to results from observations. These values are also globally set in the current implementation of the LoRa@FIIT network server [22].

We propose a more modern approach suitable for a dynamic environment, which is exactly the case for LoRa devices supporting a mobility. The network server algorithm is based on Thompson Sampling with Switching Environments. The algorithm was primarily developed as a RL technique to solve a Multi-Armed Bandit Problem (MABP), well-known from a recommendation process. It performs well in case of CP selection according to performed computer simulations [14] along with Upper Confidence Bound (UBC) [12].

These algorithms are suitable for a selection of CPs (combination of SF and transmission power, where lower values are preferable) when a success of a message delivery or signal strength are not predefined. However, we cannot let the end nodes simply select these parameters. There are two main reasons for this. A statistical model, which both algorithms are based on, has to be constantly updated using RSSI, SNR or Sequence Number values, which are not known to end nodes. Also, energy consumption comes to mind when considering such a solution. All the IoT devices have less memory, computational power and have a limited power supply, and are not ready to select parameters with limited knowledge of a network state.

Another option is to let a NS maintain a statistical model for EN and AP and send it only when the environment changes or a link quality has been degraded. ENs are responsible only for CP selection based on the updated statistical model. The main problem of the proposed solution is maintaining this model on ENs. The model is easily updated on NS using data from APs (SNR, RSSI and Sequence Number). SNR and RSSI are used to determine a channel quality and Sequence Number serves as a mechanism for early detection of link congestion (missing values) or signal loss for certain ENs.

MABP algorithm calculations are based on provided reward when a certain arm (CS selection in our case) is pressed (chosen) by a bandit (device). Due to duty-cycle limitations, it is not possible to provide immediate feedback to ENs after an uplink message has been sent. This is a reason, why we need to modify and maintain statistical model for each node and determine a moment in which it needs to be updated.

The proposed statistical model consists of spreading factor, power and a probability of successful packet delivery when using combination of both parameters. It is stored in the JSON format and could be encoded to base64 string or even BSON (Binary JSON) to minimize the size of downlink messages. When a significant change occurs in a network, a model intended for certain EN or AP has to be updated, a NS schedules a downlink message with the updated model and sends it during the next opportunity (immediately for AP, after the next uplink message for EN). During an initial device registration, a statistical model of AP that received the message is sent to the device.

3.3 Ensuring Quality of Service

LoRa@FIIT protocol has a built-in mechanism to ensure Quality of Service and optional acknowledgement. There is no need for a device to open a receiving window when no acknowledgment from network server is expected. However, there are some messages that require acknowledgement. Message acknowledgement can vary depending of application needs. This is also an advantage, because non-critical systems do not need their *hello messages* to be acknowledged by server.

One example of critical communication is an emergency message. It is sent using maximum transmission power to make sure it is received by one or more nearby APs. This special type of message also ensures QoS, but only on ENs [17].

To the authors best knowledge, there is no implementation of QoS on other network components, such as APs or NS. If the message is not prioritized on each component, the QoS is not ensured at all.

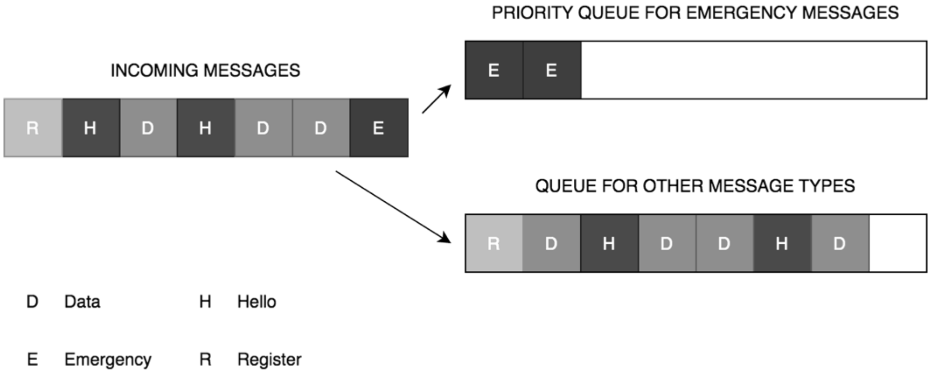


Fig. 2: Incoming messages are placed into separated queues depending on their type

In this paper, we propose a priority queue to be implemented in the wireless access point. Its aim is to handle messages from the priority queue prior to any other message type as presented in Fig. 2.

Another technique which might be considered a bad practise for QoS is the fact that a network server waits predefined time in seconds for another LoRa packets to arrive. The reason behind this is a duty-cycle limitation on APs (also called gateways or concentrators). The network server also manages a duty-cycle of APs. It waits for replicas of the received message to select an AP with the highest duty-cycle available, even if it was received only by a single AP, which is unknown to network server until the waiting timer expires. This leads to a delay, which is not tolerated in QoS ensured environment.

We assume there will be no waiting timer for the emergency queue, as it is of a high importance. If an emergency message is received by the network server, it does not wait for other packets to come. It immediately processes the message and send it to the AP from which the message originated.

4 Network evaluation using load tests

To test our proposed solution, we have designed the network architecture presented in Fig. 3. Both physical and simulated devices are present in this architecture. The devices have been divided into three groups. Groups A and B consist of simulated end nodes and access points. However, main difference between these groups is a network delay. The fact that LoRa messages could be decoded by every single access point in the nodes' range is taken into account. As a result, the network server should wait for any other duplicate messages to properly select an access point with the highest duty-cycle available. A network delay is simulated by connecting to a remote VPN server to ensure the traffic from group B will come later than packets from Group A.

We also need physical devices to compare energy efficiency with the currently implemented LoRa@FIIT library on end nodes [23]. We use the nodes based on 8 MHz ATmega328P processor, powered by 3.3V $LiSOCl_2$ battery. The selected power supply is preferred for long-life (several years) ENs, because it has a very low discharging rate, less than 1% per year. A RFM95W LoRa communication module is also used. To simulate a heart-rate measurement a MAX3015 particle-sensing board is used rather than heart-rate monitoring sensor MAX30101. However, it is sufficient to simulate the measurements and drain a battery power, for the scenario to be more realistic.

Our real-world LoRa AP in group C is based on Raspberry Pi 3 model B and iC880A concentrator and connected to the Internet. To test the scalability of the proposed solution, we use a single LoRa@FIIT NS implemented in Java. We plan to register thousands of simulated LoRa ENs and hundreds of APs to simulate the situation which might occur during this decade.

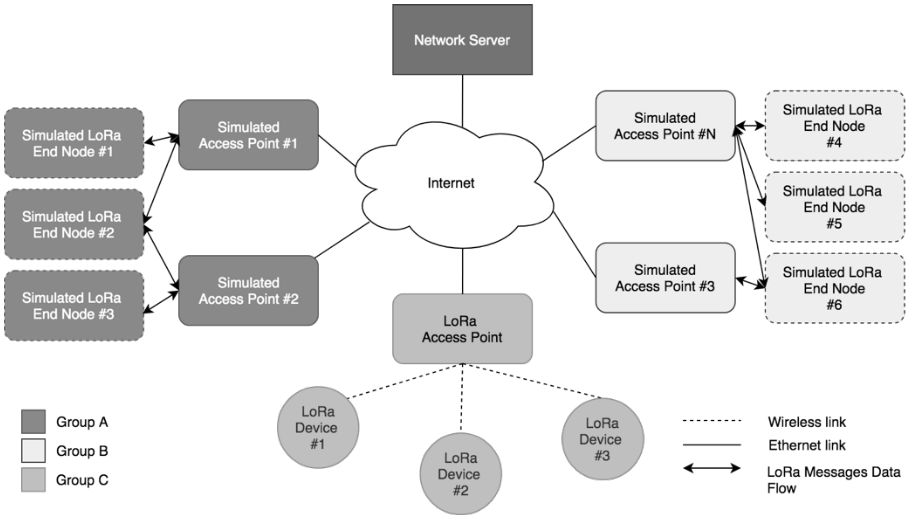

Network Server
Internet
Simulated LoRa End Node #1
Simulated LoRa End Node #2
Simulated LoRa End Node #3
Simulated Access Point #1
Simulated Access Point #2
Simulated Access Point #N
Simulated Access Point #3
Simulated LoRa End Node #4
Simulated LoRa End Node #5
Simulated LoRa End Node #6
LoRa Access Point
LoRa Device #1
LoRa Device #2
LoRa Device #3
Group A
Group B
Group C
Wireless link
Ethernet link
LoRa Messages Data Flow

Fig. 3: Proposed architecture for evaluation of network reliability and scalability

In a real-world scenario, there would be interference and packet collisions. In our simulated environment, a collision occurs on APs when two ENs transmit using the same frequency and the same SF at the same time. Interference and signal loss during a movement of patient wearing a sensor device are simulated by pseudo-randomly decreasing SNR values of received packets on APs.

At the time of writing this paper, we have successfully deployed our own LoRa@FIIT network with 10 ENs and 1 AP. We are now collecting data from stationary (not moving) nodes. One of them has a MAX30105 [24] sensor connected. Other nodes just simulate the process of heart-measuring by generating pseudo-random sensor-measured values and calculations based on the heart-rate calculation algorithm currently implemented in the library [24].

At this state, we are currently developing a console application written in Python to simulate a daily routine of ENs. The application is heavily inspired by the LoRa@FIIT library [23]. It simulates the functions of both ENs and APs.

5 Conclusion

One of the most promising technology for IoT devices communicating over long distances with minimal power consumption is LoRa. The potential of this technology could be lost when not handling with care. An increasing number of connected devices, duty-cycle limitations and unoptimized communication parameters selection threaten a real-world deployment with thousands of connected devices.

Based on the work of other researchers in this field, we used a network-wise statistical model for each network device maintained by a NS using information gathered from APs. ENs acquire this model during a registration process and use it as a knowledge base for optimal communication parameters selection. The model is further updated by NS when significant network changes occurs. As a channel selection based on statistical model can be energy-consuming, we intend to test the process of communication selection in a real-world scenario using ATmega328P based ENs.

For evaluation purpose, we have proposed a network architecture consisting of three groups of devices. In group A, there are only simulated devices with direct Internet connection. Group B also consists of simulated devices; however, they use slower VPN connection to provide higher latency than group A. The last one, group C, consists of real ATmega328P-based nodes powered by batteries.

In the future, we plan to implement modified Thompson Sampling and Upper Confidence Bound on the network server to compare the results with the currently implemented Adaptive Data Rate algorithm. There is also a possibility to choose not only between Spreading Factor and transmission power, but also using different frequencies. This can lead to higher required maintenance for the statistical model in compensation for higher throughput or collision-free communication.

Acknowledgement: This research was supported by the Ministry of Education, Science, Research and Sport of the Slovak Republic, Incentives for Research and Development, Grant No.: 2018/14427:1-26C0. This work was also partly sponsored by the Eset Research Centre.

References

1. D. Celebucki, M. A. Lin and S. Graham, "A security evaluation of popular Internet of Things protocols for manufacturers," 2018 *IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, 2018, pp. 1-6.
2. F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui and T. Watteyne, "Understanding the Limits of LoRaWAN," in *IEEE Communications Magazine*, vol. 55, no. 9, pp. 34-40, Sept. 2017.

3. ETSI, "EN300.220" [Online] Available: https://www.etsi.org/deliver/etsi_en/300200_300299/30022001/02.04.01_40/en_30022001v020401o.pdf
4. R. M. Sandoval, A. Garcia-Sanchez, J. Garcia-Haro and T. M. Chen, "Optimal Policy Derivation for Transmission Duty-Cycle Constrained LPWAN," in *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 3114-3125, Aug. 2018.D. Zorbas, G. Z. Papadopoulos, P. Maille,
5. N. Montavont and C. Douligeris, "Improving LoRa Network Capacity Using Multiple Spreading Factor Configurations," 2018 25th *International Conference on Telecommunications (ICT)*, St. Malo, 2018, pp. 516-520.
6. G. A. Akpakwu, B. J. Silva, G. P. Hancke and A. M. Abu-Mahfouz, "A Survey on 5G Networks for the Internet of Things: Communication Technologies and Challenges," in IEEE Access, vol. 6, pp. 3619-3647, 2018.
7. M. Parvin and M. R. Meybodi, “MABRP: A multi-armed bandit problem-based energy-aware routing protocol for Wireless Sensor Network,” in *The 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP 2012)*, 2012.
8. N. Michelusi and M. Levorato, "Energy-based adaptive multiple access in LPWAN IoT systems with energy harvesting," 2017 *IEEE International Symposium on Information Theory (ISIT),* Aachen, 2017, pp. 1112-1116.
9. J. Zhang, H. Jiang, Z. Huang, C. Chen, and H. Jiang, “Study of Multi-Armed Bandits for Energy Conservation in Cognitive Radio Sensor Networks,” Sensors, vol. 15, no. 4, pp. 9360–9387, Apr. 2015.
10. A. Hoeller, R. D. Souza, O. L. Alcaraz López, H. Alves, M. de Noronha Neto and G. Brante, "Analysis and Performance Optimization of LoRa Networks with Time and Antenna Diversity," in IEEE Access, vol. 6, pp. 32820-32829, 2018.
11. J. Pullmann and D. Macko, "A New Planning-Based Collision-Prevention Mechanism in Long-Range IoT Networks," in IEEE Internet of Things Journal, vol. 6, no. 6, pp. 9439-9446, Dec. 2019.
12. R. Bonnefoi, L. Besson, Ch. Moy, E. Kaufmann and J. Palicot, “Multi-Armed Bandit Learning in IoT Networks: Learning helps even in non-stationary settings,” *CROWNCOM 2017 - 12th EAI International Conference on Cognitive Radio Oriented Wireless Networks*, Sep 2017, Lisbon, Portugal.
13. R. M. Sandoval, A. Garcia-Sanchez and J. Garcia-Haro, "Optimizing and Updating LoRa Communication Parameters: A Machine Learning Approach," in *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 884-895, Sept. 2019.
14. Raouf Kerkouche, Réda Alami, Raphaël Féraud, Nadège Varsier, Patrick Maillé. Node-based optimization of LoRa transmissions with Multi-Armed Bandit algorithms. ICT 2018 - 25th International Conference on Telecommunications, Jun 2018, Saint Malo, France. pp.1-6.
15. D. Bankov, E. Khorov and A. Lyakhov, "Mathematical model of LoRaWAN channel access with capture effect," 2017 *IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Montreal, QC, 2017, pp. 1-5.
16. LoRa Alliance, "LoRaWAN™ White Paper," [Online]. Available: https://lora-alliance.org/sites/default/files/2019-11/dlms-lorawan-whitepaper_v1.pdf
17. O. Perešíni and T. Krajčovič, "More efficient IoT communication through LoRa network with LoRa@FIIT and STIOT protocols," *2017 IEEE 11th International Conference on Application of Information and Communication Technologies (AICT)*, Moscow, Russia, 2017, pp. 1-6.

18. S. Vishwakarma and Sh. Khare, "XXTEA an Optimized Encryption Design with High Feedback Substitution Box Architecture," in International Journal of Modern Engineering & Management Research, vol. 2, issue. 3, pp. 12-16, Sep. 2014.
19. M. Ahmed, B. Sanjabi, D. Aldiaz, A. Rezaei and H. Omotunde, "Diffie-Hellman and Its Application in Security Protocols," International Journal of Engineering Science and Innovative Technology (IJESIT), Nov 2012, pp. 69-73.
20. GitHub, "Chirpstack Network Server," [Online]. Available: https://github.com/brocaar/chirpstack-network-server
21. S. Li, U. Raza and A. Khan, "How Agile is the Adaptive Data Rate Mechanism of LoRaWAN?," 2018 *IEEE Global Communications Conference (GLOBECOM), Abu Dhabi*, United Arab Emirates, 2018, pp. 206-212.
22. GitHub, "LoRa Network Server," [Online]. Available: https://github.com/alexandervalach/lora-network-server
23. GitHub, "LoRa@FIIT Library," [Online]. Available: https://github.com/HalfDeadPie/LoRa-FIIT
24. GitHub, "MAX30105_Particle_Sensor_Breakout," [Online]. Available: https://github.com/HalfDeadPie/LoRa-FIIT

Applying Artificial Intelligence in the Optimization Problem

Roman Slonskyi*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
xslonskyi@stuba.sk

Abstract. *Job Shop Scheduling* (JSS) problem is a hard combinatorial optimization problem in the fields of computer science and operations research. It is *NP-Hard* and, therefore, heuristics are used for solving large-size cases of the problem. One of the real-world JSS examples is minimizing makespan of an assembly line finding an appropriate sequence of orders to be processed by operators.

Although many approaches of optimization can solve JSS, metaheuristic algorithms appeared to be the most effective. However, it is recommended to use neighbourhood-based approaches for solving JSS or hybrid algorithms with a local search element.

Genetic Algorithm, Simulated Annealing, Electrical Particle Swarm Optimization and *Particle Swarm Optimization with Multitype Individual Enhancement* were implemented and preliminary results were presented.

Keywords: metaheuristics, job shop scheduling, artificial intelligence, combinatorial optimization problem.

1 Introduction

1.1 Scheduling of Conveyor Systems

Complex conveyor systems are commonplace in the factories and delivery companies. And they usually tackle with a problem of scheduling of those systems so that they can receive the most profitable outcome of their work. However, scheduling is a hard combinatorial optimization problem and because of its huge state space an optimal solution is rarely found.

The problem, that we were tackling with, related to optimization of employees' work at the conveyor system in the warehouse. There was a list of orders which should be rearranged in such a way that the work was done in a minimal period of

[*] Bachelor study programme in field: Informatics
Supervisor: Assoc. Professor Peter Lacko, Institute of Informatics, Information Systems and Software Engineering, Faculty of Informatics and Information Technologies STU in Bratislava

time. In this case, state space is ***N!*** big, where ***N*** is a number of orders, if all the orders are unique.

Containers were conveyed to a fixed number of picking points, where an operator put appropriate items into it according to an order. But containers did not visit all the operators, because an order might not need items that some operator possessed. In addition, there were several starting points of containers: further or closer to a delivery point.

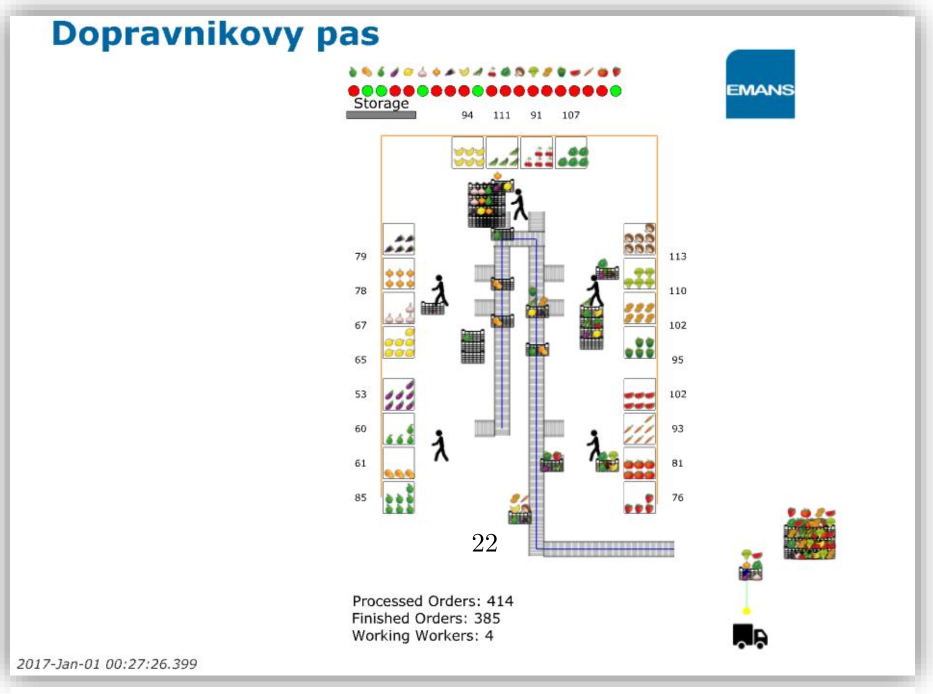
In our case, we had a conveyor with 5 workers, who were responsible for the order picking. An order consisted of a list of products to be picked. There were 20 types of products in an order and totally there were 1000 orders. An order could not include more than one equivalent product, so the maximum order size was equal to 20. Each worker had only 4 unique product types at his disposal. Hence, if an order did not need a product type, that some worker did not have, then the worker was skipped.

Order processing took time for each employee, but different products had different processing times and they were stochastic. The time was calculated according to normal distribution $N(\mu, \sigma^2)$, where a mean value was equal to 2, standard deviation was equal to 2, minimum and maximum values were equal to 1 and 4, respectively.

The conveyor was not fed with new containers continually. If any of workers had more than 4 orders in a queue to be processed, then supply of new containers was stopped. In addition, workers did not have an infinite number of products at his disposal, but they were periodically supplied. The period was 20 seconds after the moment when less than 5 products of a particular type had left in worker's possession. After that period, 4 new products were delivered to the operator.

This type of a scheduling problem appeared to be equivalent to the *Job Shop Scheduling (JSS)* problem.

For the visualization of such a conveyor system and the evaluation of its efficacy we used JaamSim simulator. It was shown in the **Fig. 1.**



Processed Orders: 414
Finished Orders: 385
Working Workers: 4

2017-Jan-01 00:27:26.399

Fig. 1. Visualization of the problem in the JaamSim simulator

1.2 Job Shop Scheduling

Job Shop Scheduling problem is a task of minimization of the execution period of jobs that are carried out on machines. Suppose we have a set of jobs J, a set of machines M and a set of operations O. Each job is represented as a sequence of operations that have to be performed on definite machines. Machines perform only unique types of operations, so if M_1 can perform the operation O_1 then M_2 must not perform O_1. Jobs visit machines and are processed there for a predefined period of time $\mu(J_x, M_y)$. If we have the job $J_a = \{O_2, O_4, O_5\}$, the machine $M_2 = \{O_2, O_5\}$ and $M_3 = \{O_4\}$, then J_a must be performed on machines M_2 and M_3. However, there are several constraints in the process. Machines can perform only one job at the same time, but it cannot be preempted and be processed twice on the same machine. If a job visits a busy machine, it will wait in a queue until the machine is free. All types of machines must be unique, but it is optional for the job types. Besides the general constraints the JSS problem can be expanded with some technological restrictions (i.e. time of travelling from $M_1\ to\ M_2$). Rearranging the jobs order and the operations order in every job we need to minimize the time of accomplishment of all the jobs. This time is called a makespan. The JSS problem was visualized in the **Fig. 2**.

In our case, a worker was interpreted as a machine that performed order picking. Each order was considered as a job and products were operations of the jobs. However, we omitted permutations of operations in jobs because machines were connected sequentially and therefore a job J_a, that should be fulfilled on the machines M_2 and M_3, could not visit the M_3 before the M_2.

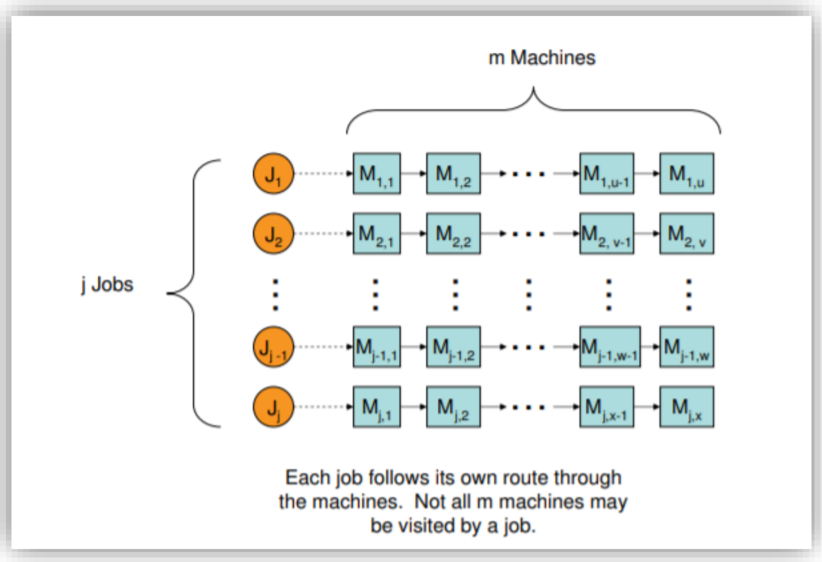


Fig. 2. Diagram of the generalized job-shop problem. The figure was taken from [1]

2 Related Work

Metaheuristic techniques are usually applied for the JSS problem. They showed fast convergence to near-optimal solutions. The most popular algorithms appeared to be Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Simulated Annealing (SA). However, not all pure algorithms have a local-search element, which is efficient for converging to near-optimal results. Hence, hybrid algorithms are common for that type of a problem. Several algorithms that showed promising results in solving the JSS problem were introduced.

2.1 **Hybrid Evolutionary Algorithm**

This Hybrid Evolutionary Algorithm (HEA) [2] is a combination of particle swarm optimization and simulated annealing algorithms. As the PSO could be easily stuck in a local optimum, utilizing SA helped to jump over an obstacle and to continue searching for better solutions. Moreover, in the proposed algorithm SA was modified by λ search method. That has been designed for searching not for the one solution but a set of solutions that are compared according their fitnesses and the best one is chosen. Tests showed better results in comparison with the other algorithms mentioned in the paper.

2.2 Multitype Particle Swarm Optimization

PSO with Multitype Individual Enhancement Scheme (MPSO) is an algorithm presented by Lin et. al. [3] to solve the JSS problem. Combined random-key encoding for transforming a continues space into a discrete one, multi-type individual enhancement scheme for adding local search feature and the PSO algorithm, they presented a robust algorithm for solving the combinatorial optimization problem. The algorithm was tested on 43 instances that are a part of a benchmark in the OR-Library. Research showed that the proposed MPSO can find near-optimal solutions with a smaller population size and less iterations, comparing to the HIA [4], HEA [2] and HGA [5] algorithms. In addition, for 35 instances the best-known results were reached.

2.3 Genetic Algorithm for Stochastic JSS

Deming Lei in his work [6] presented an algorithm based on GA, that was used for the stochastic variation of the JSS problem, where time of processing of a job on a machine was uncertain. GA was constructed of encoding, decoding, binary tournament selection, generalized order crossover (GOX) and swap mutation. The proposed algorithm showed better results in comparison with the Simulated Annealing and Similar Particle Swarm Optimization [9].

2.4 Hybrid Genetic Algorithm

Hybrid genetic algorithm (HGA), that was proposed by Qing-dao-er-ji and Wang [10], showed promising results according to compared algorithms. From 43 instances of the benchmark the hybrid genetic algorithm found the best-known solutions in 33 instances. The main emphasis was on population diversity and improving quality of solutions. Presented local search operator provided the desired quality improvement and new selection operator kept individuals being unique. In addition, the convergence of the proposed approach to the global optimum was proven by the authors.

2.5 Electrostatic Particle Swarm Optimization

Electrostatic Particle Swarm Optimization (EPSO) is modification of the canonical PSO that can deal with multimodal optimization types of problems. It does not require any additional parameters to tune by a user as other variants of PSO multimodal optimization. Comparing to the standard PSO algorithm, EPSO does not have a global best-found particle for the whole population, which is determined according to fitnesses of individuals. Each particle in a population has its partner to which it gravitates. The force of gravitation is computed according to the Coulomb's law in physics. EPSO showed good results in testing: it has found all the optima and the number of evaluations of a fitness functions were less comparing to the other tested algorithms.[8]

Originally, EPSO was not intended for the JSS problem. However, since it showed promising results with multimodal optimization problems and high-dimensional JSS problem can be interpreted as multimodal problem, presumably EPSO can show good results for the JSS problem, as well.

2.6 Summary

According to the aforementioned descriptions of algorithms we can say that MPSO seems to be the best-suited algorithm for our case, because considering that one evaluation of the fitness function lasts approximately for 4 seconds, a small population size and a low number of fitness function evaluations are desired.

3 Method Proposal

For a solution of the problem we decided to use several metaheuristic algorithms for finding the most appropriate approach. The first two implemented algorithms were Genetic Algorithm and Simulated Annealing. The algorithms were chosen because they appeared to be common solutions for optimization problems. In addition, they were trivial to implement. After analysing state-of-the-art algorithms for solving optimization problem and, particularly, JSS problem, Electrostatic Particle Swarm Optimization and Particle Swarm Optimization with Multitype Individual Enhancement Scheme appeared to be decent approaches for our case.

For all the algorithms we used equal representation of individuals/particles in a population. Each individual/particle was described as a list of 1000 orders:

$$i_n = \{o_1, o_2, \ldots, o_{1000}\} \tag{1}$$

where orders were represented as lists of unique numbers from the range [1, 20]. The minimal size of an orders could be equal to 1 and the maximal size could be equal to 20. Hence, the order o_1 could be as follows:

$$o_1 = \{2, 5, 20\} \tag{2}$$

3.1 Genetic Algorithm

The GA was implemented in a classical way [7]. It consists of several operations: *crossover*, *mutation* and *selection*. For the crossover process we chose one-point crossover. Crossover is performed every iteration. Mutation was defined as a swap of two elements in a population. The chance of the operation was set to 10%. For the selection process was used the roulette-wheel method and the principle of elitism was added – the best several chromosomes are always selected for the next population. The whole population consisted of 30 individuals.

3.2 Simulated Annealing

Simulated annealing was implemented in a canonical variant [11]. Starting temperature was set to 15. Cooling speed was equal to 0,09. The temperature was continually decreasing by the value of the cooling speed every 5 iterations. Stop condition of the algorithm was the moment when temperature is less of equal to 0,1.

3.3 EPSO

The Electrical Particle Swarm Optimization algorithm was implemented according to Barrera's and Coello's description [8]. The only difference was that we had a discrete type of the problem, so several transformations of the algorithms were made.

A distance between two particles are calculated as a difference of order sequences. Consider two particles p_1 and p_2 that are represented as follows:

$$p_1 = \{o_1, o_2, o_3\} \tag{3}$$

$$p_2 = \{ o_3, o_2, o_1\} \tag{4}$$

The distance between these particles is:

$$||p_1 - p_2|| = 2 \tag{5}$$

Velocity of a particle is represented as a number of swaps of orders to be made. Swaps are made in such a way that a sequence of orders becomes more similar to the other sequence that has been chosen as a particle with the biggest force.

Parameters c_1 and c_2 were set to 0.3 and 0.2, respectively. A size of the swarm consisted of 30 particles.

3.4 MPSO

The implementation of the Particle Swarm Optimization with Multitype Individual Enhancement Scheme (MIE) algorithm was almost equivalent to the description of the authors [3]. The distinction is in generating new particles in the MIE. Instead of generating only one particle, we generated 8 new particles and a particle with the best fitness was chosen. All the parameters of the algorithm were set to the values chosen by authors.

In addition, for this algorithm we have tried two approaches for applying it on our problem. The first approach was a standard one – manipulating all the items in a list of orders to find a good solution. During the second approach we divided the algorithm into three steps: manipulating all the orders, then manipulating the first half of the orders and after all manipulating the second half of the orders. Using this technique, we were trying to increase the quality of the local search.

4 Preliminary Results

All the algorithms were being tested for 5 times with a random permutation of orders. A quantity of iterations of implemented algorithms were set in such a way that a total number of fitness function evaluations was equal to approximately 2700. The quality of a solution is represented by a makespan of processing all the orders, so the lower a number, the better solution is. Experimental results of all algorithms are shown in the **Table 1**. Best known solution (BKS) is equal to 3969,0413839999997.

From the chart we can see that the SA showed itself as the least effective algorithm: an average fitness of the best-found fitnesses is 4245,646 and that is 7% worse than the BKS.

EPSO algorithm performed as the most stable algorithm: all the tests found values that are close to 4175. An average fitness of all the test is 4175.412 and that is 5.2% worse than the BKS.

GA sometimes showed itself better than EPSO, but it was not so stable as the EPSO was. An average fitness of the GA is 4186.442 and that is 5.5% worse than the BKS.

MPSO with a standard approach (MPSO1) had better results than the EPSO. It was not so stable as EPSO, but variance of results was less than GA had. An average fitness is equal to 4150.43 and that is 4.6% worse than the BKS. MPSO with our local search approach (MPSO2) performed better than all the aforementioned methods. Stability was the same as standard MPSO had. An average fitness is equal to 4099.538 and that is 3.3% worse than the BKS.

5 Conclusions

The problem of minimizing time for picking all the orders by workers at the conveyor system scheduling the order list appeared to be equivalent to the Job Shop Scheduling problem. However, there were some modifications in the interpretation.

The JSS problem is usually solved by algorithms from the metaheuristics field and GA and PSO are the most popular approaches for solving the problem. However, many works showed that ‘pure’ algorithms performed not so effectively comparing to hybrid algorithms or approaches with local-search modification in itself.

Prototypes of the Simulated Annealing, Genetical Algorithm, Electrical Particle Swarm Optimization and Particle Swarm Optimization with Multitype Individual Enhancement Scheme algorithms were implemented, and experimental tests were carried out. EPSO was the most stable, MPSO with our local search approach found a solution with minimal value from all the tests, the SA showed the worst results.

From the results we can concluded that a local search element of an algorithm definitely increased quality of solutions, but alone this element appeared not to be appropriate for solving the problem. MPSO showed the best results by the virtue of high-quality integration of the SA algorithm into PSO. And with our extra local search technique we could achieve even better results. Hence, the final proposed method is Particle Swarm Optimization with Multitype Individual Enhancement Scheme.

In the future work it is advised to experiment with the different configurations (population size, constants, etc.) of the implemented algorithms in order to find an optimal one for our problem.

Table 1. Results of the implemented algorithms

Algorithm	Test 1	Test 2	Test 3	Test 4	Test 5	Best	Deviation*	Avg.	Avg. deviation
SA	4215.95	4273.29	4274.91	4247.08	4217	4215.95	6.2%	4245,646	7%
GA	4199.37	4146.38	4162.08	4214.22	4210.16	4146.38	4.5%	4186.442	5.5%
EPSO	4173	4176.16	4167.8	4182.45	4177.65	4167.8	5%	4175.412	5.2%
MPSO1	4157.13	4159.96	4116.14	4149.33	4169.54	4116.14	3.7%	4150.43	4.6%
MPSO2	4105.01	4087.15	4123.95	4107.86	4073.72	**4073.72**	**2.6%**	**4099.538**	**3.3%**

*Deviation = (BKS – Best) / BKS

References

1. Kleeman M.P., Lamont G.B.: Scheduling of flow-shop, job-shop, and combined scheduling problems using MOEAs with fixed and variable length chromosomes. In: Evolutionary Scheduling, pp. 49-99). Springer, Berlin, Heidelberg (2007).
2. Ge H., Du W., Qian F.: A hybrid algorithm based on particle swarm optimization and simulated annealing for job shop scheduling. In: Third International Conference on Natural Computation (ICNC 2007), vol. 3, pp. 715-719. IEEE (2007).
3. Lin T.L., Horng S.J., Kao T.W., Chen Y.H., Run R.S., Chen R.J., Lai J.L., Kuo I.H. An efficient job-shop scheduling algorithm based on particle swarm optimization. Expert Systems with Applications 37(3), 2629-2636 (2010).
4. Ge H.W., Sun L., Liang Y.C., Qian F. An effective PSO and AIS-based hybrid intelligent algorithm for job-shop scheduling. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 38(2), 358-368 (2008).
5. Ge H., Du W., Qian F.: A hybrid algorithm based on particle swarm optimization and simulated annealing for job shop scheduling. In: Third International Conference on Natural Computation (ICNC 2007), vol. 3, pp. 715-719. IEEE (2007).
6. Lei D.M. Minimizing makespan for scheduling stochastic job shop with random breakdown. Applied Mathematics and Computation 218(24), 11851-11858 (2012).
7. Holland J.H., 1992. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press.
8. Barrera J., Coello C.A.: A particle swarm optimization method for multimodal optimization based on electrostatic interaction. In: Mexican International Conference on Artificial Intelligence, pp. 622-632. Springer, Berlin, Heidelberg (2009).
9. Lian Z., Jiao B., Gu X. A similar particle swarm optimization algorithm for job-shop scheduling to minimize makespan. Applied mathematics and computation 183(2), 1008-1017 (2006).
10. Wang Y. A new hybrid genetic algorithm for job shop scheduling problem. *Computers & Operations Research 39*(10), pp.2291-2299 (2012).
11. Kirkpatrick S., Gelatt C.D., Vecchi M.P. Optimization by simulated annealing. *Science 220*(4598), pp.671-680 (1983).

Driver’s drowsiness detection

Jakub Marinčič[*]

STU Faculty of Informatics and Information Technologies, Ilkovičova 2
842 16 Bratislava 4, Slovakia
jakub.marincic@gmail.com

Abstract. Traveling is an indispensable part of people's life. Everyone uses public transport or roads on an everyday basis. From walkers and bicycles, to buses and big trucks, all these participants need to arrive to their destination quickly and most of all, safely. In the daily news, we can see more and more accidents on the roads caused by a driver with lack of sleep or a person driving under the influence of alcohol. Our goal is to avoid situations like these, or better to say, help drivers to realize, that their condition is not suitable to drive. Many car makers are developing their own systems, that are available only for their concern. We are focusing on a group of drivers in a role with much more responsibility - they are the bus drivers. They handle big and heavy vehicles and carry mostly a large amount of people. Add to this a long journey and serious problem can happen. Based on this, we decided to implement our system for Funtoro devices. A platform, that can be installed into buses from more than one brand.

Keywords: OpenCV, Object detection, Funtoro, Python, Android

1 Platform

We are developing this project for Funtoro devices. An entertaining system that can be installed in transport vehicles. It’s easy to find in long-distance buses as a tablet, that are usually docked in the seat in front of the passenger. It can be used to watch movies, read books, play games or order drinks and snacks (if available).
System consists of FreeBSD Server and several Android tablets (also called monitors), where one is in “Driver mode” – this opens new abilities, such as freezing all monitors and displaying a message, but declines the other ones, like watching movies, reading books. This gives us an option, to contact only the driver when malicious behavior is detected.

[*] Bachelor study programme in field: Informatics Supervisor: Vladimír Kunštár, Institute of Computer Engineering and Applied Informatics Institute of Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

2 Solution design

Our project is currently in development. This section contains our planned approach on individual steps.

See Fig. 1 For better understanding of the interaction between entities.

2.1 Image capture

Images are captured on IP camera with good night vision capabilities thanks to IR technology (Milesight C2682-P). The camera provides two streams (primary and secondary), where different resolution and codec can be used. We are listening to primary stream with resolution 1280x720 and h.264 codec. For this an RTSP stream is opened on server.

Server is running FreeBSD operating system, where our application for driver's behavior detection (hereinafter referred to as application) is running. The application is written in Java and uses OpenCV library.

2.2 Image processing

Every image is evaluated. We need to find person, face, eyes and any objects, that driver could, but shouldn't interact with (like a mobile phone).

First a face needs to be found. For this we use the Viola-Jones algorithm.[1] It is much simpler than CNNs, so we don't need to spend so much processing power. For example, in [2] they use much faster CPU to achieve at least 21 FPS using Lite version of YOLO.

When the face is located, we are looking for eyes. There is no need to search whole image, so now we are looking only in the rectangle, where the face was detected. Using the same algorithm, but this time with different classifier (trained for eyes).

To detect drowsiness, the UKF algorithm is used, this algorithm has great positive detection rate of 99% [3]. The idea is to measure how much is eyelid covering the pupil. This level is measured and evaluated. Into consideration comes level of coverage and time.

We are trying to find a smartphone in an image. If it is located, we are measuring time, that it spends there. Since camera should be in location, where it cannot see any store compartments, the only case the smartphone is visible, is when driver operates it. After 3 seconds the notification is sent to alert the driver. Searching is done with the same Viola-Jones algorithm, but a different classifier.

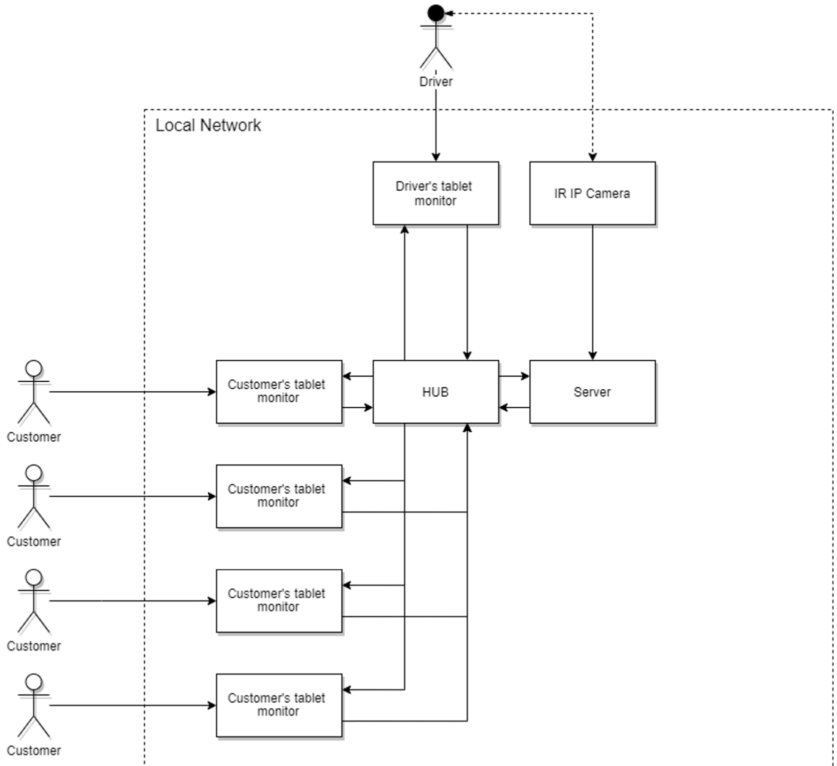
Every detected object is put into blue rectangle to provide visual feedback. This is streamed into network.

2.3 Displaying live status

Available is option to display the current status of the driver. On the one hand, there are notifications, that are displayed, when needed (e.g. when drowsiness is detected).

Another way is a build-in app. It can show history of notifications and live video stream. This stream is provided by server and already edited.

Fig. 1. Diagram shows the kind of interactions between each entity. Number of customer monitors is individual for each vehicle.


Driver
Local Network
Driver's tablet monitor
IR IP Camera
Customer's tablet monitor
HUB
Server
Customer
Customer's tablet monitor
Customer
Customer's tablet monitor
Customer
Customer's tablet monitor
Customer

3 Future work

Application has a lot of room to expand. We will focus on efficiency of individual detections, with time consumption and accuracy in mind. Also detecting more objects would be beneficiary.

Next, we want to add progress bar, that gives the driver feedback about his performance. This would be based on the number of notifications he received. In other words, the more concentrated to driving he is, the more points he gets. These points will be displayed in progress-bar and transformed to percentage. The color of the progress-bar will be dependent on percentage.

In further future, next sensors could be added. Information we could benefit from would be steering angle, heartrate, alcohol in blood. These could help us even more to detect any unwanted behavior like driving under influence of alcohol[4] or help with drowsiness detection[3].

4 Conclusion

Safety on roads is an important topic, that every car maker is trying to improve, but it is still not perfect. Systems like these can help to implement many features and help drivers and so forth increase safety. It alerts driver, when his sleepiness starts to interfere with his performance in situations, when he may think it's not that bad. Also interaction with smartphone is monitored and in case of positive monitoring, an alert is sent. These alerts are stored, with the ability to display them through application and edited record from camera is streamed and also viewable in application. Since these are only early stages of development there is a lot of place for optimization and new or improved features.

References

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, no. July 2014, 2001.

[2] R. Huang, J. Pedoeem, and C. Chen, "YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers," *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, pp. 2503–2510, 2019.

[3] A. Sahayadhas, K. Sundaraj, and M. Murugappan, "Detecting driver drowsiness based on sensors: A review," *Sensors (Switzerland)*, vol. 12, no. 12, pp. 16937–16953, 2012.

[4] S. Al-Sultan, A. H. Al-Bayatti, and H. Zedan, "Context-aware driver behavior detection system in intelligent transportation systems," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 9, pp. 4264–4275, 2013.

Prediction and Optimization of Electricity Usage

Bc. Kitti Nagyová[*]

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
kitty.nagyova@gmail.com

Abstract. Nowadays, there is a huge boom in the utilization of smart meters in homes. These meters measure and send electricity consumption measurements to data centers. By analyzing this data, it is possible to predict and optimize future electricity consumption. We can also calculate possible savings from the data, for example when installing new elements into the network, such as photovoltaic panels, high-capacity batteries and more. In this paper, we analyzed the forecasts of electricity consumption for 30 different households in one locality. Predicted power consumption is optimized using connected battery. Optimization is understood to minimize electricity consumption from the distribution network. Finally, the original and the optimized consumption are compared.

Keywords: prediction, optimization, neural networks, PSO.

1 Introduction

Electricity is a part of the daily lives of a large part of the population. Electricity consumption by humans in 2018 exceeded 23,000 TWh and is expected to increase further in the upcoming years [3]. On the other hand, the European Union is increasingly regulating the use of fossil fuels and it is the responsibility of countries to use clean electricity [10]. The future of energy are undoubtedly the microgrids, which are small local energy systems capable of autonomous supply of electricity to a particular location and self-sufficiency. For this purpose, they use coordinated renewable energy sources such as photovoltaic panels, wind farms and electricity produced from biomass. The main advantages of these grids include system resilience, economic and energy efficiency, and environmental protection.

Scientific and research work in power engineering field has already made a significant progress. Several new methods and procedures have been proposed to optimally manage electricity. However, there is still a place for improvement of the electricity

[*] Master study programme in field: Intelligent Software Systems
Supervisor: Dr. Petra Vrablecová, Institute of Informatics, Information Systems and Software Engineering, Faculty of Informatics and Information Technologies STU in Bratislava

production and consumption planning using standard mathematical calculations, but also statistical methods, artificial intelligence and machine learning. This work is aimed at comparison of the power demand prediction by several types of neural networks and minimization of electricity consumption from the grid after connecting the battery using particle-swarm optimization (PSO) algorithm. By comparing the consumption before and after optimizing the microgrid we can tell the impact of the newly added element, i.e. the high-capacity battery.

2 Related Work

Currently, there are several researches in the field of electricity consumption prediction and optimization. A common approach to prediction is the use of statistical methods, artificial intelligence methods, or their combination. These approaches can achieve high precision [1, 2]. C. Heghedus et al. [5] compared three types of neural networks – fully recurrent neural network, LSTM and GRU. They implemented their work in Python using TensorFlow framework. They used data provided by a local electricity supplier in Stavanger, Norway. They divided the experiment into two different case studies: either only Mondays are selected from dataset and predictions are made just for Mondays, or prediction for the following month when the entire dataset is considered. In their work, future consumption value is predicted based on its previous two values. Their mean absolute error (MAE) rate for daily forecast is 0.09 using GRU and the best MAE for the entire month is 0.059 using LSTM. Various principles are used in the field of optimization, among others evolutionary algorithms. Compared to other optimization methods (linear programming and genetic algorithm), PSO achieves better results [6]. In the field of power engineering, PSO is a good choice for problems of electricity price prediction [13] and also for electricity usage optimization [4], or maximization of the consumption of electricity produced by photovoltaics [8].

3 Optimal battery utilization schedule

The goal of our method is to obtain a battery charging schedule for the next day to maximize the usage of electricity from photovoltaics. To achieve this, we need to know the future electricity consumption and production. Finally, we optimize battery charging based on the predictions. Our method consists of 3 steps:

1. *power demand prediction* – We decided to use neural networks (see section 5).
2. *power production prediction* – We did not deal with power production prediction as it is a complex research field and it was out of scope of this research work. Therefore, we simulated the forecasts by adding Gaussian noise to actual production data.
3. *battery charging optimization* – We used PSO algorithm to calculate the optimal charging and discharging of the battery for the next day based on the predictions (see section 6). With this optimized battery schedule, we could determine the new

electricity consumption from the distribution network and evaluate the saved costs for electricity from the distribution network. We could also evaluate the return time of investment into high-capacity battery.

4 Dataset

In this paper, we use smart meter data collected in Pecan Street – Texas from 133 different homes. The datasets contain information about the production and consumption of electricity in addition to the date and time of measurement. The data are from January 1, 2014 to December 31, 2015.

Because installing a high-capacity battery is not cost-effective for one household, we decided to implement our solution for 30 households that have similar electricity consumption habits. We selected the households based on the clustering results, since clustering also improves the power demand prediction [7]. We compared datasets using the Dynamic Time Warping (DTW) algorithm and created clusters. The DTW algorithm uses a dynamic programming approach and aligns the time series to minimize the measured distance between them. The final dataset was calculated using eq. 1, where F_t is the sum of the values of 30 datasets at time t, n is the count of datasets and R_{xt} is value from dataset x at time t.

$$F_t = \sum_{x=1}^{n} R_{xt} \tag{1}$$

In the final time series, we can identify two periods for the electricity consumption (see Fig. 1.): peak period – 10:00 – 18:00, and flat period – 18:00 – 10:00.

The electricity production is the highest from 9:00 to 13:00. At this time, the production is higher than consumption and the difference between production and consumption is electricity that can be stored in the battery and used later.

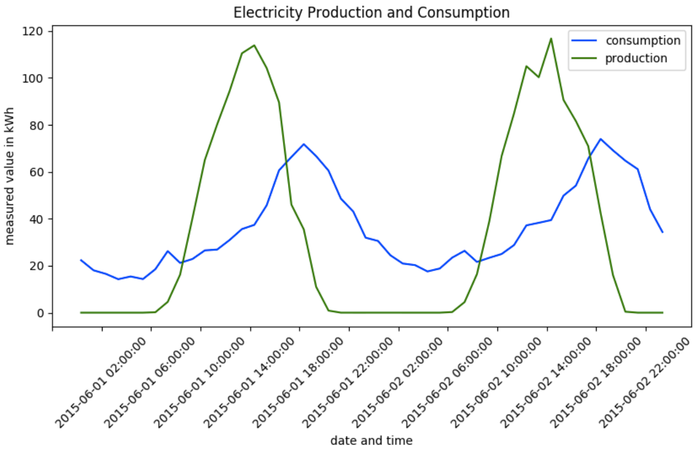


Fig. 1. Example of electricity production and consumption.

For prediction purposes, we added historical measurements at the same time from past days. Based on the correlation matrix shown in Fig. 2, we selected 5 attributes of historical measurements as an input of the prediction.

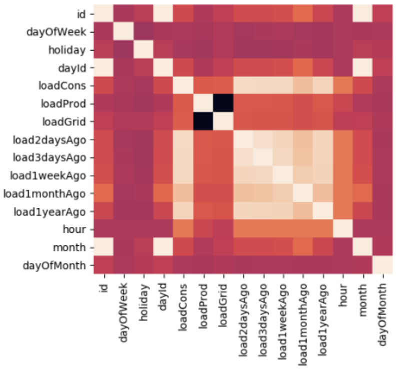


Fig. 2. Correlation matrix of dataset.

5 Prediction

Prediction using neural networks performs mostly better than prediction using statistical methods [12]. However, different neural networks have different accuracy. One part of this work is to compare the prediction of power consumption using 4 different neural networks: feed-forward neural network, LSTM, GRU and Echo-state neural network.

- feed-forward neural network,
- LSTM,
- GRU
- Echo-state neural network.

5.1 Training and evaluating

The input values for each neural network are the mentioned historical loads: load from 2 days ago, load from 3 days ago, load from a week ago, load from a month ago, and load from a year ago. The output value is the predicted consumption. We predict data for 28 days in summer from July 15, 2015 to August 15, 2015 and winter data from January 15, 2015 to February 15, 2015. To calculate the power consumption forecast, a Python program was written with keras and scikit-learn libraries containing the implementations of the forecasting algorithms. The prediction model was evaluated by computing the mean squared error (MSE), where R is the vector of observed values of the variable being predicted and Y is the vector of predicted values (see eq. 2).

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(R_t - Y_t)^2 \tag{2}$$

The predicted values were evaluated by computing mean absolute percentage error (MAPE):

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|R_t - Y_t|}{R_t} \tag{3}$$

6 Optimization model

In this paper, we minimized the electricity consumption from the distribution network after connecting a new microgrid element – high-capacity battery. For calculating the return time of investment, we work with the price €0.11 for 1 kWh of energy [11]. A suitable battery technology for implementation is a technology called Powerwall from Tesla, which contains of 10 batteries specified in Table 1 [9].

Table 1. Parameters of the selected battery.

Battery name	Tesla Powerwall 2
Capacity	13.5 kWh
Technology	Lithium-ion
Guaranteed cycles	10,000
Effectivity	90 %
Operating temperature	-20 °C - 50 °C
Dimensions	115 x 75.3 x 14.7 cm
Weight	114 kg
Price	5,500 €

We can store in the battery at time t the electricity b_t that equals to the difference of production p_t and consumption of electricity c_t at that time t (see eq. 4). Electricity consumption c_t at time t is equal to the sum of production p_t at time t and electricity from distribution network d_t at time t (see eq. 5).

$$b_t = p_t - c_t \tag{4}$$

$$c_t = p_t + d_t \tag{5}$$

If we connect a battery, the electricity from the distribution network d_t is equal to:

$$d_t = c_t + (b_{t-1} - b_t) - p_t \tag{6}$$

Optimization is minimizing the electricity consumption from distribution network; our final minimizing function is:

$$\min X = \sum_{t=1}^{N} d_t \tag{7}$$

7 Results

7.1 Calculation of return time of investment

In 2015, all 30 households together consumed 226,110.37 kWh of electricity. After connecting the battery, we calculated that the consumption is 204,261.33 kWh from the distribution network, which means reduction of 21,849.04 kWh, in total €2,403.39. The price of the battery technology is €55,000 and the return time of investment is 22 years and 11 months.

7.2 Prediction

The following Table 2. shows the prediction results and Fig. 4. shows the result consumptions of the first predicted summer week as curves. Daily consumptions during the whole summer are similar, these consumptions were predicted with better accuracy. The most accurate model is LSTM network with rate 7.17 %. Fig. 5 shows the result consumptions of the first predicted winter week as curves. Daily electricity consumptions in the winter are erratic. It could be caused by unpredictable weather and therefore incalculable usage of heating systems. In this case also the LSTM network has the lowest error rate 16.73 %.

Table 2. MAPE of four prediction models.

	Summer data	Winter data
Feed-forward network	7.73 %	16.86 %
LSTM	7.17 %	16.73 %
GRU	7.20 %	16.80 %
Echo-state network	7.87 %	21.50 %

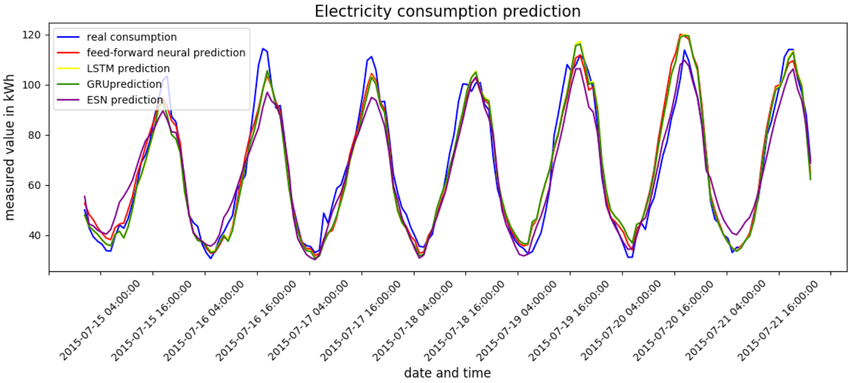


Fig. 3. Prediction of consumption in summer.

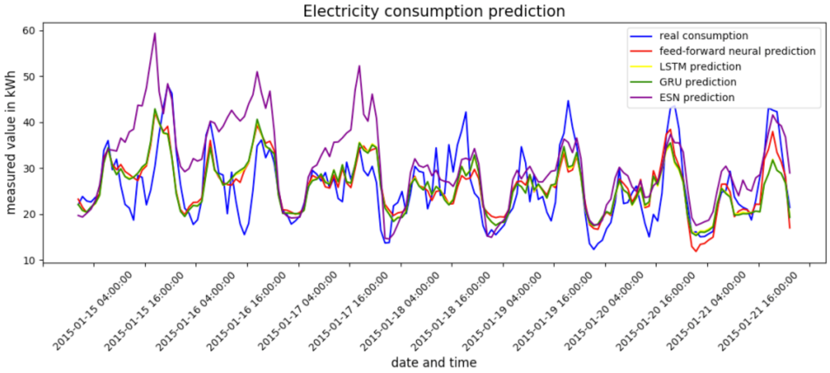


Fig. 4. Prediction of consumption in winter.

7.3 Optimization

In section 6.1 we determined the return time of investments. Using predicted consumption with LSTM model, noisy production and algorithm PSO described above, we calculated the optimal charging and discharging of the battery. In summer the original consumption was 29,753.79 kWh, after connecting the battery the consumption is 28,024.32 kWh, which means in the summer 5.81 % less electricity from distribution network is consumed. In the winter the original consumption was 13,418.68 kWh, after connecting the battery the consumption is 11,046.07 kWh, which means reduction 17.68 %. This higher percentage of reduction is caused by lower consumption in the winter, but almost the same high production. Fig. 6 shows the original and the new consumptions from distribution network in the first week of the summer, Fig. 7 in the winter.

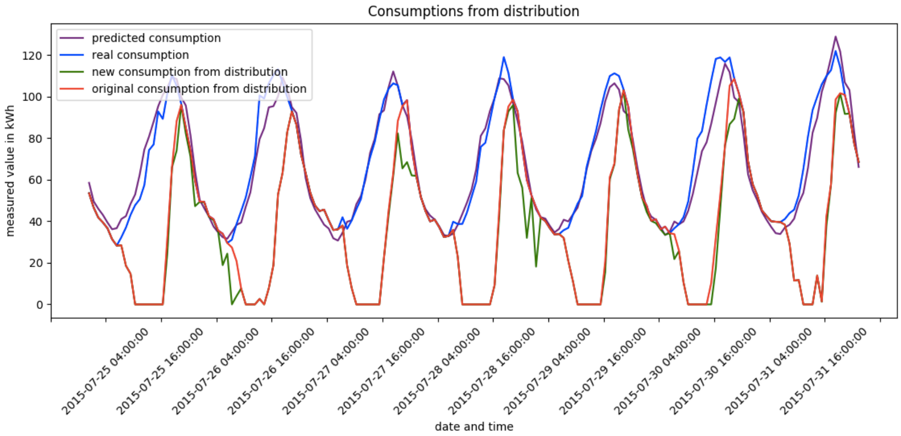


Fig. 5. Consumption from distribution network in summer before and after optimization.

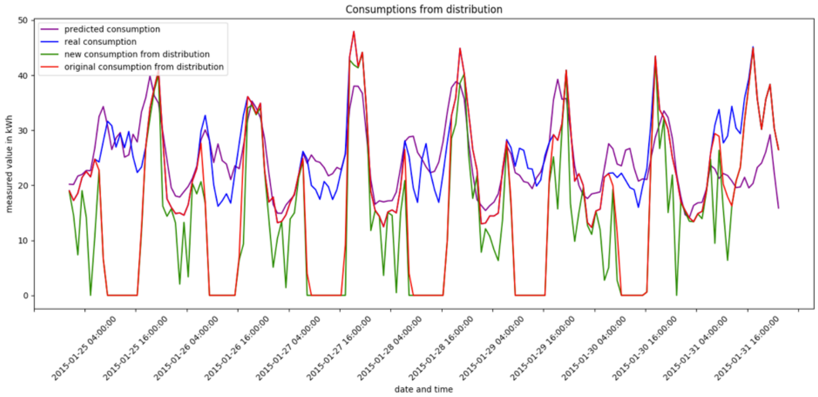


Fig. 6. Consumption from distribution network in winter before and after optimization.

8 Conclusion

In this paper, power demand predictions using different types of neural networks were compared and an optimization of power consumption by utilization of high-capacity battery was modelled. Neural networks were trained using past measurements of electricity consumption. Using the predicted consumption data with the best accuracy and production data with added Gaussian noise we calculated the optimal battery charge management in the future time period. For optimization we introduced our minimizing function and we used PSO algorithm. There are more ways to improve this work. A possible improvement of the work would be an optimization that considers the cost of electricity and minimizes it, even if the final consumption from the distribution network is higher. Another possible improvement could be an optimization that consider the number of battery charges and discharges. The third possible improvement is to model another optimization function with more attributes, such as weather, population, etc. However, these data are not available for now. Such an extension could find a better model with better battery charging and discharging scheduling, which would ultimately mean more battery power consumption and less electricity consumption from the distribution network.

References

1. Atienza, N.A.C. et al.: Prediction and Visualization of Electricity Consumption in the Philippines Using Artificial Neural Networks, Particle Swarm Optimization, and Autoregressive Integrated Moving Average. In: 2018 3rd International Conference on Computer and Communication Systems (ICCCS). pp. 135–138 (2018). https://doi.org/10.1109/CCOMS.2018.8463351.
2. Barić, I. et al.: Short-Term Forecasting of Electricity Consumption Using Artificial Neural Networks - an Overview. In: 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). pp. 1076–1081 (2019). https://doi.org/10.23919/MIPRO.2019.8757012.
3. Enerdata: Global Energy Statistical Yearbook 2019: Electricity domestic consumption, https://yearbook.enerdata.net/electricity/electricity-domestic-consumption-data.html, last accessed 2020/02/24.
4. Eseye, A.T. et al.: Optimal energy management strategy for an isolated industrial microgrid using a Modified Particle Swarm Optimization. In: 2016 IEEE International Conference on Power and Renewable Energy (ICPRE). pp. 494–498 (2016). https://doi.org/10.1109/ICPRE.2016.7871126.
5. Heghedus, C. et al.: Neural Network Architectures for Electricity Consumption Forecasting. In: 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). pp. 776–783 (2019). https://doi.org/10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00143.
6. Kamal, M. et al.: Energy Resource Planning for a Rural Microgrid: Comparison of Results using Different Optimization Algorithms. In: 2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP). pp. 1–6 (2019). https://doi.org/10.1109/ICACCP.2019.8882901.
7. Laurinec, P. et al.: Adaptive Time Series Forecasting of Energy Consumption Using Optimized Cluster Analysis. In: 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW). pp. 398–405 (2016). https://doi.org/10.1109/ICDMW.2016.0063.
8. Sivakumar, L.P. et al.: Implementation of Particle Swarm Optimization for Maximum Power Absorption From Photovoltaic System Using Energy Extraction Circuit. In: 2019 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS). pp. 1–4 (2019). https://doi.org/10.1109/INCOS45849.2019.8951378.
9. Tesla: Meet Powerwall, your home battery, https://www.tesla.com/powerwall , last accessed 2020/04/17.

10. The European Parliament and the Council of the European Union: Regulation (EU) 2019/943 of the European Parliament and of the Council of 5 June 2019 on the internal market for electricity. Off. J. Eur. Union. 62, L158, 54–191 (2019).
11. The U.S. Energy Information Administration: ???, https://www.eia.gov/electricity/, last accessed 2020/02/25.
12. Valipour, M. et al.: Comparison of the ARMA, ARIMA, and the autoregressive artificial neural network models in forecasting the monthly inflow of Dez dam reservoir. J. Hydrol. 476, 433–441 (2013).
13. Zhang, C., Ma, P.: Short-term electricity price forecasting based on PSO algorithm and RBF neural network algorithm. In: 2010 International Conference on Measuring Technology and Mechatronics Automation, ICMTMA 2010. pp. 334–337 IEEE (2010). https://doi.org/10.1109/ICMTMA.2010.22.

Identification and recognition of the driver's behavior

Matej DRHA*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
matejdrha@gmail.com

Abstract. The subject of my work is to create a system for the behavior recognition of the driver. The system will be designed for the Funtoro platform in the form of client – server solution. An Android application will be implemented for the client, which will communicate with the server using the HTTP protocol. The system will cooperate with other devices such as a camera, GPS and speakers, to send or receive data. The driver will be monitored by the camera and the video will be sent to the server where it will be processed. The server will use computer vision and artificial intelligence to recognize objects in the driver's cabin and evaluates the driver's behavior using a safety scale. The level of attention will be constantly displayed on the driver's monitor and the warning message will be shown, if the level of attention exceeds a certain value. The system will be based on similar existing solutions and will be expanded with additional features.

Keywords: driver's attention, Funtoro, computer vision, artificial intelligence.

1 Introduction

One quarter to one half of all road accidents are caused by the fact that the driver is not focused on driving. Just as it is important to keep your eyes on the road, you need to maintain the necessary level of focus while driving. People overestimate their ability to do more things at once and underestimate the likelihood of an accident happening. The problem is that the situation on the road can change in a second, and a driver busy with other activities responds too late.

Activities in the car that are not directly related to driving impair the quality of driving by diverting attention from driving. Distractions can be both physical and mental. Mental states such as fatigue, distraction and aggression lead to an increase of the possibility of an accident happening.

Use of mobile phone in cars is one of the most dangerous driver's distractors that contribute to road accidents and injuries inside and outside the vehicle. While hands-free phones and other methods such as speed dialing or voice activation reduce physical distraction, the most negative factor associated with using a mobile phone while

[*] Bachelor study programme in field: Informatics
Supervisor: Vladimír Kunštár, Institute of Computer Engineering and Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

driving, whether hands-free or hand-held, is diversion from driving. Driver's response time can be 30% to 50% slower when calling with a mobile phone while driving. In practice, this means that the braking reaction time is slowed by approximately 0.3 to 0.7 seconds.[1]

Fortunately, there are many technological devices for detecting driver's distraction currently. A sleepy person shows several characteristic facial movements, including fast and continuous blinking, head nodding and frequent yawning. Video is usually obtained using a simple CCD sensor or a webcam and then image processing algorithms are used to detect the face, eyes or mouth. Using element extraction algorithms, PERCLOS, yawning frequency, or head angle is extracted from the image. Driver's behavior is then analyzed and classified using classification methods. A similar procedure to the detection of driver fatigue can be used to detect interaction with a mobile phone, GPS, radio, food or drinks. By analyzing the video using object recognition algorithms, we can determine what object does the driver have in his hand or what equipment is he operating.

2 Related work

Almost every car company has already integrated artificial intelligence into its latest vehicles, whether it is the driver's attention monitoring, vehicle surroundings monitoring or autonomous vehicle driving systems. Most of the companies develop their own technologies and do not share them with others. Below, we will describe two systems that are not directly linked to any brand and one that is used in Cadillac vehicles. The common feature of all solutions is that they use a camera to monitor the driver's behavior.

Driver Monitoring System (DMS) from Veoneer. Veoneer is the world's largest company producing safety electronic devices, Advanced Driving Assistance Systems (ADAS), collaborative and automated driving systems. Veoneer uses system components developed by Seeing Machines, which is a world leader in Human-Machine Interface (HMI) and Artificial Intelligence (AI) technologies.[2][3]

The DMS consists of a small infrared camera and other dashboard-mounted infrared light modules. The camera focuses on the driver's face and monitors their attention by continuously searching for and tracking the features of the driver's face - simultaneously tracking multiple facial features, eyelids and pupils, large and sudden head rotations, and eye angles. The video from the camera is processed in real time using the FOVIO Driving Monitor Engine (DME) module. DMS works robustly in all foreseeable driving conditions, including direct sunlight and total darkness, regardless of race or ethnicity, even when the driver wears sunglasses.[4][5]

Optalert. Startup Optalert is focused on monitoring the driver's attention by tracking his face. The only product on the market, that has scientific confirmation from Harvard Medical School. It uses attention monitoring system based on Eyes on Task

(EoT) which also takes into account other driver visual tasks and can be specified for any vehicle interior. The Johns Drowsiness Scale (JDS) continuously evaluates the driver's inattention (from 0 to 10), so the system can warn the driver in time and avoid dangerous situations. The JDS driver status scale can be used as an input to other vehicle systems. This system can be also easily used in ADAS or Vehicle to everything (V2x) systems.[6]

DRIVE & CloudDETECT from EDGE3 Technologies. EDGE3 develops solutions for monitoring drivers, passengers and various personal belongings with a focus on complex on-board monitoring systems based on artificial intelligence for all vehicle segments, including autonomous vehicles.[7]

DRIVE is a comprehensive approach to eliminating driver distraction by monitoring the driver using multiple sensors in real time. It can also identify mobile phone use, eating, drinking, smoking, or monitor hand placement.[8]

CloudDETECT is the first cloud-based solution on the market to extract and identify unsafe driver behavior. The recorded video (with identified driver's mistakes) is stored in the cloud, then it can be analyzed using CloudDETECT to evaluate the driver's behavior. CloudDETECT can identify the same aspects as DRIVE.[9]

Use of components in my project

From each existing system, we take something we want to implement into our solution, try to improve some of the shortcomings and develop a competitive product to analyze the driver’s behavior.

Functionalities of existing systems that we would like to implement:

- eye, face and head monitoring (eyelids, head tilt, yawning, talking ...)
- hand monitoring (EoT definition, steering wheel use, object handling)
- monitoring under all foreseeable conditions (direct sun, night, glasses, different races and ethnicities)
- own scale to assess drowsiness and inattention
- using data from other system components (GPS)
- real-time image analysis
- disturbance-free interaction with the driver

Shortcomings of existing solutions that we will try to improve:

- camera location (most systems have a camera directly in front of the driver) - in our solution we find the ideal camera location experimentally, as the bus driver's cabin is different from ordinary cars

3 Solution design

3.1 Implementation platform - Funtoro

We will implement our system on Funtoro devices. These devices are most commonly used in buses, trains and airplanes and provide passengers with multimedia entertainment. The system consists of a server and interactive monitors that are placed on every seat, in front of the passengers and can be supplemented with additional features such as GPS navigation or a camera.

Funtoro server FMS5711-SHD-256GB
The server can handle up to 72 monitors at the same time - sending a different movie video stream, application data, and TV broadcast to each monitor. The server is equipped with Intel Atom processor with 4GB of DDR3 memory, 32GB of internal eMMC memory and 32GB of mSATA storage. The server runs the FreeBSD 12.0 operating system, which is tailored to the needs of the Funtoro Media On Demand (MOD) multimedia system. An external SSD with a capacity of 256 GB is connected to the server and is primarily used for storing massive (multimedia) content.

Funtoro monitor MCT-005-04GB
The monitor acts as a standalone device with its own Android 4.4 operating system and can operate in two modes - driver mode and passenger mode. The mode can be set for each monitor separately after accessing the configuration mode. The 10.1" touch screen with a resolution of 1024x600 pixels offers a high-quality viewing experience for movies or TV. The monitor offers wireless connection with Bluetooth and Wifi devices.

Camera Milesight MS-C2682-P
Wide-view IP camera records video at a resolution of 1280x960 pixels at 30 frames per second. An important component is the infrared LED that allows video capture at night. The camera is integrated into the system using a hub.

3.2 Top-level system architecture design

The proposed system will represent the client - server architectural model. The server will be a Funtoro server and the client will be a Funtoro monitor. In addition, other components such as a camera and GPS device will be connected to the system to be used by the server. An audible notification speaker will be connected to the monitor. The following figure (Fig. 1) shows the mentioned architecture design.

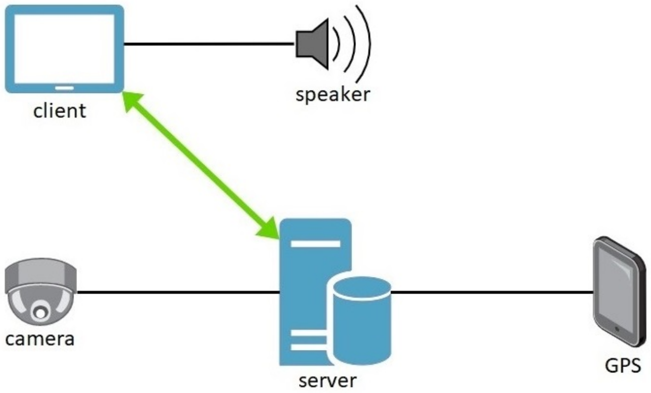


Fig. 1. Top-level system architecture design

3.3 Server part design

Server functions can be divided into three main parts: image processing, attention classification, communication and client request processing. Server receives a video stream from the IP camera via the HTTP protocol. The video is then processed and sent to the client as needed (more details are provided in the video processing subchapter). The server will also use GPS data - especially speed to classify attention. Communication with the client is realized via the HTTP API and the functionality of the connection is controlled by response messages from the client.

Video processing. The video is recorded at 30 fps. We will try experimentally to determine if we can analyze all captured images without any significant delay. Video processing will be divided into several steps. Video will be processed in parallel - in one stream the detection of fatigue and micro-sleep is processed and in the other the distraction by other objects is processed.

To identify fatigue, the face and its features must be detected first. We use the Viola Jones algorithm, which uses the Haar cascade classifier, from the OpenCV library, to detect the face. Level of eye openness is obtained using the SVM classifier. By experimentally combining several measured factors, we will try to determine the degree of drowsiness as accurately as possible.

In the second parallel flow we will focus on the driver's distraction, especially using the phone, eating, drinking, or having hands off the wheel. We will apply various deep-learning algorithms for object recognition from the OpenCV library, which uses pre-trained neural networks. We will try to optimize algorithms experimentally.

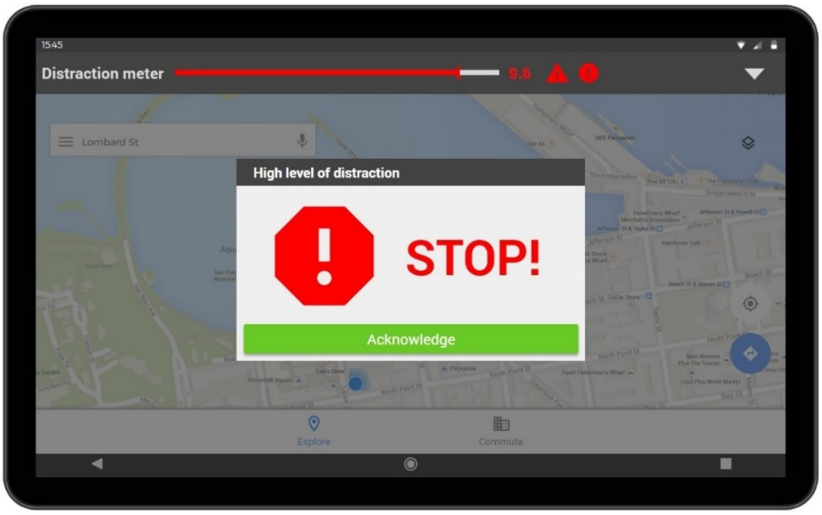
Each part will be classified separately to distinguish whether the driver is tired or distracted, and then the rate of distraction (fatigue) is calculated and sent to the client. The ideal frequency of sending messages to the client is calculated experimentally.

3.4 Client part design

The app will be designed for the Funtoro monitor running Android 4.4. Since the image processing and most calculations are performed by the server, the primary task of the application will be to display the data to the driver and to warn him in case of excessive distraction.

The application will work in two modes. The first mode is standard - the application is displayed fullscreen and the driver can interact with the application. The second mode is when the application is running in the background - it performs all standard calculations and is displayed in the foreground through a narrow bar with the distraction meter. In case of danger, a pop-up window appears over other running applications (Fig. 2).

Upon receiving data from the server, the client calculates the degree of drowsiness or distraction, determines the objective hazard (from 1 to 10) and displays it on a scale. The application will record the individual driver mistakes (total drowsiness level, phone usage time, handling other objects) and the driver will be able to view statistics at the end of the journey. The application will be able to stream video from the camera - when setting the ideal camera position or to verify the functionality of the entire system (driver mistakes will be highlighted directly in the video stream).



High level of distraction

STOP!

Fig.2. Distraction meter is placed in the top of the screen when the driver uses another app. Pop-up window is shown because the level of distraction has exceeded the limit.

References

1. Jeanne Breen Consulting, Car telephone use and road safety, (2009).
2. Who we are, https://www.veoneer.com/en/who-we-are, last accessed 2019/12/2.
3. Driver Monitoring Systems, https://www.veoneer.com/en/driver-monitoring-systems, last accessed 2019/12/2.
4. Technology, https://www.seeingmachines.com/technology/, last accessed 2019/12/2.
5. Automotive, https://www.seeingmachines.com/industry-applications/automotive/, last accessed 2019/12/2.
6. Integration into ADAS systems, https://www.optalert.com/industries/human-centric-automotive/integration-into-adas-systems/, last accessed 2019/12/2.
7. EDGE3 Technologies, https://edge3technologies.com/#About, last accessed 2019/12/2.
8. DRIVE, https://edge3technologies.com/drive/, last accessed 2019/12/2.
9. CloudDETECT, https://www.edge3technologies.com/clouddetect/, last accessed 2019/12/2.

Automatic Drum Transcription using Partially Fixed Non-negative Matrix Factor Deconvolution

Martin ŽÁK[*]

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technology
Ilkovičova 2, 842 16 Bratislava, Slovakia
zakyz.martin@gmail.com

Abstract. Automatic Drum Transcription (ADT) is a sub-task of Automatic Music Transcription (AMT) problem in Music Information Retrieval (MIR), which is constrained to transcribing percussive instruments. Previous works have demonstrated accurate transcriptions in simpler scenarios with decrease in performance as problem formulation gets more complex. Our approach combines strengths of 2 different NMF-based solutions. Partially Fixed Non-negative Matrix Factorization (PF-NMF) fixes drum components in the dictionary during factorization to get clean detection of drum events while leaving the rest flexible to explain the remaining musical content. Non-negative Matrix Factor Deconvolution (NMFD) increases fidelity of reconstruction and produces sharper activations for onset detection in general, by modelling temporal dimension of the components in the dictionary explicitly. We evaluate our method on two problem formulations: Drum Transcription of Drum-only recordings (DTD) and Drum Transcription in the presence of Melodic instruments (DTM). In DTD evaluation, our algorithm outperforms NMF-based state-of-the-art solutions by reaching 0.92 F-measure.

Keywords: Automatic drum transcription · non-negative matrix factorization · digital signal processing · spectrogram decomposition.

1 Introduction

Automatic Music Transcription (AMT) is a long-standing challenge in Music Information Retrieval (MIR) research. The goal is to analyze a musical audio recording and produce symbolic notation that carries musical semantics about given audio piece. The problem of AMT is considered a "holy grail" of MIR, as it produces a representation that is very fruitful when approaching many other tasks. For feasibility reasons, large body of previous work has considered

[*] Master study programme in field: Intelligent Software Systems
Supervisor: Lukáš Marták, Institute of Computational Perception, Johannes Kepler University Linz

the problem with constraining assumptions. In its full complexity, considering multiple instruments with all the distinctive musical artifacts they can produce, the grand challenge remains posed.

Popular music usually includes several musical instruments that sounds at the same time (polyphony) and create harmony. Therefore AMT has to deal with many non-trivial sub-tasks as musical instrument detection, onset and offset detection, multiple pitch estimation, musical dynamics estimation, etc. Typically, problem is considered for a particular instrument or group of instruments.

Automatic Drum Transcription (ADT) is a sub-task of AMT. For harmonic and melodic instruments, as considered in AMT, each note is mainly described by it's pitch, onset timing and duration. Each note is also played with some loudness, which is quantified by a property called ”velocity” in the MIDI standard, but it started to be considered only recently [1]. In a set of percussive instruments such as drums, each drum instrument has typically a fixed pitch and elicits ”note” - a particular recognizable sound - when played. Hence, pitch detection of AMT translates into source attribution in ADT. As note duration is almost never controlled by the musician playing drums, it is not considered in the score notation either. Also majority of systems focuses on transcribing basic strikes and don't take into account playing technique Therefore, problem is reduced to detection of note onsets [2].

There are several applications where automatic drum transcription system would be really useful. During drum lectures or practicing drums at home it could help to teach drummer to play drums on time and correctly using gamification techniques . Also to separate tracks for particular drums during studio drum recording can be helpful either for professional music production where several microphones capture all drums, or for amateur music producers who might only have a single microphone. Further automatic detection of rhythm could improve searching capabilities for music by groove [2].

Previous research on ADT systems has addressed above mentioned problems within several specific problem formulations. Namely, *Drum Sound Classification* (DSC), *Drum Technique Classification* (DTC), *Drum Transcription of Drum-only recordings* (DTD) that with the presence of additional percussion instruments is called *Drum Transcription in the presence of additional Percussion* (DTP) or with the presence of melodic instruments is called *Drum Transcription in the presence of Melodic instruments* (DTM) [2]. We decided to target DTM and DTD formulations, as those are very common cases in popular music and there is still room for improvement.

The remainder of this paper is structured as follows. First, related work is outlined in Sec. 2. After a review we describe our approach in detail in Sec. 4. Following Sec. 5 describes our evaluation methodology, used dataset, configuration of our algorithms and critical assessment of results. Finally, Sec. 6 summarizes this contribution and presents ideas for future elaboration.

2 Related Work

In this section, state-of-the-art systems in DTM and DTD are presented. It should be noted that in literature authors usually target these research problems on 3 main drum instruments: Hi-Hat (HH), Snare Drum (SD) and Kick Drum (KD). Therefore we present our solution for the same drum instruments in this work.

The transcription process has been previously done by different machine learning methods. Existing solutions can be mainly discerned into two categories: Deep Neural Networks (DNN) and Non-negative Matrix Factorization (NMF).

Within DNN approaches, Bi-Directional Recurrent Neural Network (BDRNN) was used. It was shown that this approach worked very well on audio recordings where drummers played the snare drum with different techniques [3]. Other BDRNN-based approaches used soft attention mechanisms and peripheral connections to model more complex temporal relationships in their latent features. Although they achieved high accuracy, robustness against data distribution shift didn't stand out among these models [4]. Another approach based on two-layer RNN architecture with time-shifted labels (tsRNN) exploited idea of slightly shifting original annotations forwards in time for training, and treat detected onsets during prediction time the same way. It has been shown to perform as well as BDRNN with significantly lower complexity [5]. In contrast, Convolutional Neural Networks (CNN) based approaches have shown much better robustness [4], but in combination with RNN or NMFD (Non-Negative Matrix Factor Deconvolution) overall performance didn't increase significantly [4, 6, 7].

Among NMF-based approaches, different variants were applied. Partially Fixed Non-Negative Matrix Factorization (PFNMF) keeps part of the dictionary matrix (e.g. percussive sources only) fixed during the factorization, while the rest is being updated. This formulation was first applied without any improvements [8], followed by an improved formulation that adds weight matrix for different components and additional template adaptation mechanisms [9]. One of the other approaches was NMF decomposition with Semi-Adaptive bases (SANMF). The semi-adaptiveness refers to the behaviour of dictionary matrix over the progress of decomposition [10]. It is held almost fixed at the beginning, so focus is on identifying activations of drums as captured by default templates, and as the iteration limit approaches, the dictionary is progressively released for updates, to adapt to the particular recording. Another extension of NMF, called Non-negative Matrix Factor Deconvolution (NMFD), incorporates temporal dimension in its dictionary of components. This is especially helpful for drum sounds, some of which exhibit highly non-linear spectro-temporal evolution [11].

In conclusion, DNN-based approaches seem to be more successful than NMF-based ones, as they have better capability of modelling complex relationships and finding latent features in music in general. On the other hand, they are prone to overfitting, harder to interpret and less robust to data distribution shift. NMF-based approaches are less prone to overfitting, easy to interpret and more robust. On the other hand they generally show worse results than DNNs (although only on average, the performance gap is not huge) and worse modelling capacity.

In this work, we investigate the avenue of augmenting existing NMF-based approaches with a goal of increasing performance without losing the valuable properties like interpretability of components and robustness that we have in contrast with the DNN-based methods.

3 Non-negative Matrix Factorization

The NMF algorithm takes a non-negative input matrix $\mathbf{X}$ and decomposes it into 2 matrices. In audio decomposition $\mathbf{X} \in \mathbb{R}_+^{m \times n}$ is a magnitude spectrogram with m spectral components and n time frames. NMF approximates the input matrix by linear combination of the factor matrices [12]

$$\mathbf{X} \approx \mathbf{W} \cdot \mathbf{H} \tag{1}$$

Matrix $\mathbf{W} \in \mathbb{R}_+^{m \times r}$, dictionary matrix, represents spectral features of sources present in the input matrix. Matrix $\mathbf{H} \in \mathbb{R}_+^{r \times n}$, activation matrix, dispose of time-varying gains representing amount of presence of these sources in the input. Number of sources to model (denoted by r) is picked as an assumption a priori.

This decomposition is an iterative process where success of reconstruction is measured using a cost function such as KL divergence or Euclidean distance and matrices are updated using multiplicative update rules [13] or projected coordinate gradient descent [12].

3.1 Partially Fixed Non-negative Matrix Factorization

The PFNMF algorithm is a derivative of NMF. It splits matrices $\mathbf{W}$ and $\mathbf{H}$ into two parts - components and activations for harmonic instruments and drum instruments separately. For better understanding we give an illustration of algorithm in Fig. 1 reprinted from [8]. Reconstruction of input is as follows [8]:

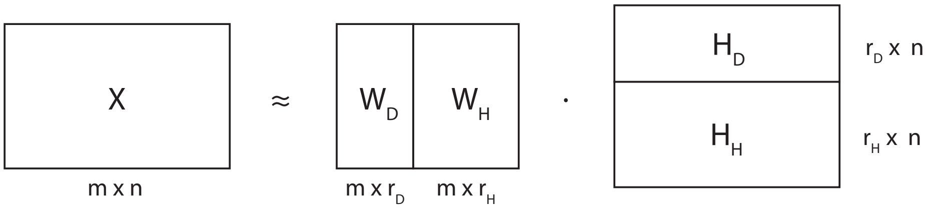


Fig. 1: Illustration of PFNMF algorithm, reprinted from [8]. Drum and Harmonic instruments are denoted by subscripts $(\cdot)_\mathrm{D}$ and $(\cdot)_\mathrm{H}$ respectively.

$$\mathbf{X} \approx \mathbf{W}_D \cdot \mathbf{H}_D + \mathbf{W}_H \cdot \mathbf{H}_H \tag{2}$$

During factorization process it fixes drum components $\mathbf{W}_D \in \mathbb{R}_+^{m \times r_D}$ in the dictionary matrix to get clean detection of drum events in activation matrix $\mathbf{H}_D \in \mathbb{R}_+^{r_D \times n}$ while leaving the rest ($\mathbf{W}_H \in \mathbb{R}_+^{m \times r_H}$ and $\mathbf{H}_H \in \mathbb{R}_+^{r_H \times n}$) flexible to explain the remaining musical content. r_D denotes number of drum sources while r_H denotes number of harmonic sources.

This method assumes initialization of $\mathbf{W}_D$ dictionary to spectral characteristics that are statistically representative of the considered drum instruments, according to some data sample (more in subsection 4.4).

3.2 Non-negative Matrix Factor Deconvolution

The NMFD algorithm is yet another derivative of NMF. It increases fidelity of reconstruction and produces sharper activations for onset detection by modelling temporal dimension of the components in the dictionary matrix explicitly. This effectively trades off time and space complexity of the method for fidelity of reconstruction.

The formulation of NMFD algorithm builds upon the $\overset{i\rightarrow}{(\cdot)}$ operator, which shifts the columns of its argument by i spots to the right, padding by zeros from the left [14]. The reconstruction is then given by

$$\mathbf{X} \approx \sum_{t=0}^{T-1} \mathbf{W}^{[t]} \cdot \overset{t\rightarrow}{\mathbf{H}} \tag{3}$$

where T is the size of temporal axis in dictionary entries (number of time frames with which each spectrogram-like entry in the musical instrument dictionary matrix is stored), and $\mathbf{W}^{[t]} \in \mathbb{R}_+^{m \times r}$ is a slice of t-th time frame that form the full 3-dimensional dictionary matrix $\mathbf{W} \in \mathbb{R}_+^{m \times r \times T}$.

Detailed formulation of the NMFD algorithm is available in [14]. We attach an illustration of this algorithm applied to the DTD problem of ADT in Fig. 2 for the reader's convenience.

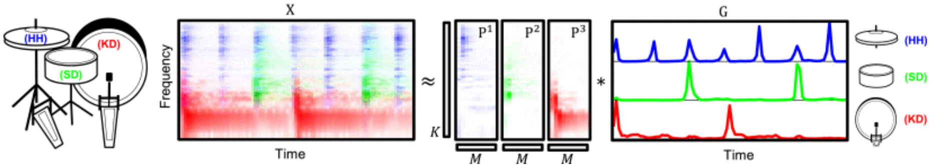


Fig. 2: Illustration of NMFD algorithm. Reproduced from [2].

4 Our approach

We propose to combine the strengths of two existing NMF-based approaches - Partially Fixed Non-negative Matrix Factorization (PFNMF) and Non-negative

Matrix Factor Deconvolution (NMFD). Hence, we call this method Partially Fixed Non-negative Matrix Factor Deconvolution (PFNMFD).

4.1 Partially Fixed Non-negative Matrix Factor Deconvolution

We combine improved PFNMF [9] with NMFD [14] by fixing drum components in the dictionary matrix during factorization process while modelling temporal dimension of the components in the dictionary matrix and altering the decomposition by sum over temporal axis according to the NMFD rule. Figure 3 illustrates the decomposition along with dimensions of its parameters.

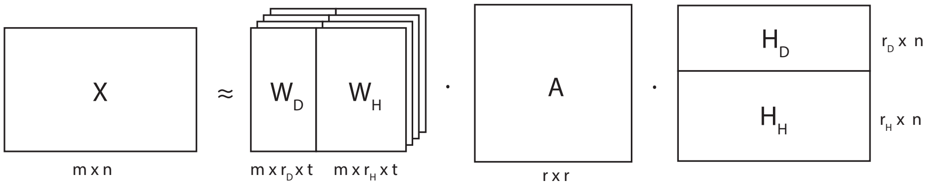


Fig. 3: Illustration of the PFNMFD algorithm.

The input spectrogram is now factorized by

$$\mathbf{X} \approx \sum_{t=0}^{T-1} (\alpha \cdot \mathbf{W}_D^{[t]} \cdot \overset{t\rightarrow}{\mathbf{H}}_D + \beta \cdot \mathbf{W}_H^{[t]} \cdot \overset{t\rightarrow}{\mathbf{H}}_H) \tag{4}$$

where α and β come from the weight matrix $\mathbf{A} \in \mathbb{R}_+^{r \times r}$ such that $r = r_D + r_H$, defined in the improved PFNMF [9]. In words, $\mathbf{A}$ is a diagonal matrix that contains weighting coefficients for individual sources to balance drum and harmonic dictionaries in the NMF cost function.

More precisely, $\mathbf{A} = diag([\alpha \vec{\mathbf{1}}_{r_D}; \beta \vec{\mathbf{1}}_{r_H}])$ where $\vec{\mathbf{1}}_c$ is an all-one vector of length c. This matrix is thus defined by (α, β) which are set to increase weights of drum instruments $\alpha = (r_D + r_H)/r_H$ at the expense of weights of harmonic instruments $\beta = r_H/(r_D + r_H)$. If $r_H = 0$, the algorithm doesn't balance instruments [9].

Multiplicative Update Rules Since we fuse together PFNMF and NMFD algorithms, the multiplicative update rules for our parameters $\mathbf{W}_H$, $\mathbf{H}_D$, $\mathbf{H}_H$ are different as well, but still specific to the cost function in use.

Euclidean distance cost function yields following updates:

$$\mathbf{W}_H^{[t]} \longleftarrow \mathbf{W}_H^{[t]} \otimes \frac{\mathbf{X}(\overset{t\rightarrow}{\mathbf{H}}_H)^\top}{(\alpha \mathbf{W}_D \mathbf{H}_D + \beta \mathbf{W}_H \mathbf{H}_H)(\overset{t\rightarrow}{\mathbf{H}}_H)^\top} \tag{5}$$

$$\mathbf{H}_D \longleftarrow \mathbf{H}_D \otimes \frac{{\mathbf{W}_D^{[t]}}^\top \overset{t\leftarrow}{[\mathbf{X}]}}{{\mathbf{W}_D^{[t]}}^\top (\alpha \mathbf{W}_D \mathbf{H}_D + \beta \mathbf{W}_H \mathbf{H}_H)^{\leftarrow t}} \tag{6}$$

$$\mathbf{H}_H \longleftarrow \mathbf{H}_H \otimes \frac{{\mathbf{W}_H^{[t]}}^\top \overset{t\leftarrow}{[\mathbf{X}]}}{{\mathbf{W}_H^{[t]}}^\top (\alpha \mathbf{W}_D \mathbf{H}_D + \beta \mathbf{W}_H \mathbf{H}_H)^{\leftarrow t}} \tag{7}$$

$$\forall t \in [0 ... T-1]$$

In case of KL divergence cost function, multiplicative update rules were following:

$$\mathbf{W}_H^{[t]} \longleftarrow \mathbf{W}_H^{[t]} \otimes \frac{\frac{\mathbf{X}}{\alpha \mathbf{W}_D \mathbf{H}_D + \beta \mathbf{W}_H \mathbf{H}_H} (\overset{t\rightarrow}{\mathbf{H}_H})^\top}{\mathbf{1} (\overset{t\rightarrow}{\mathbf{H}_H})^\top} \tag{8}$$

$$\mathbf{H}_D \longleftarrow \mathbf{H}_D \otimes \frac{{\mathbf{W}_D^{[t]}}^\top [\frac{\mathbf{X}}{\alpha \mathbf{W}_D \mathbf{H}_D + \beta \mathbf{W}_H \mathbf{H}_H}]^{\leftarrow t}}{{\mathbf{W}_D^{[t]}}^\top \mathbf{1}} \tag{9}$$

$$\mathbf{H}_H \longleftarrow \mathbf{H}_H \otimes \frac{{\mathbf{W}_H^{[t]}}^\top [\frac{\mathbf{X}}{\alpha \mathbf{W}_D \mathbf{H}_D + \beta \mathbf{W}_H \mathbf{H}_H}]^{\leftarrow t}}{{\mathbf{W}_H^{[t]}}^\top \mathbf{1}} \tag{10}$$

$$\forall t \in [0 ... T-1]$$

4.2 System Overview

The transcription system pipeline can be broken down into 3 main stages: dictionary initializaiton, decomposition, and onset detection.

First, we initialize the drum components of the dictionary $\mathbf{W}_D$ with the ”templates”, that we construct form recordings of isolated hits on drum instruments. There are multiple ways to go about this, but we choose to construct our templates by averaging time-aligned magnitude spectrograms of isolated samples, for the sake of simplicity. This would be the ”learning” bit in the machine learning context.

Remaining stages are relevant to inference. First, magnitude spectrogram is computed from the input audio snippet and subsequently decomposed using our PFNMFD method. Activations of the drum components $\mathbf{H}_D$ are then filtered for onset detection to generate predicted onset-level transcription. Drum events are recognized from potential ones based on several rules (more in 4.5).

Figure 4 illustrates the high-level architecture of our ADT system. We further detail these processing stages in subsections 4.3, 4.4, and 4.5.

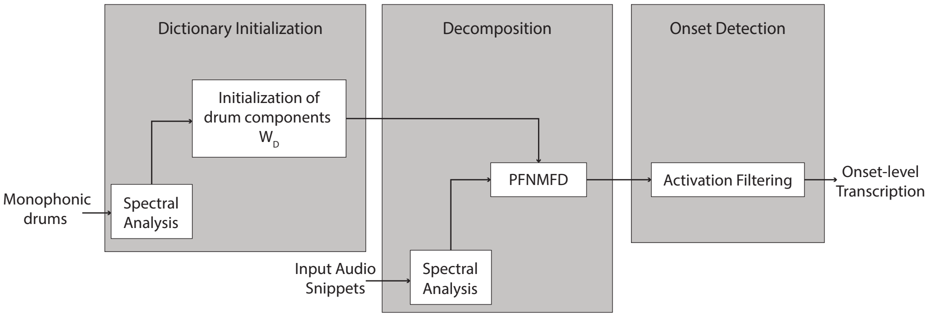


Fig. 4: Pipeline of our ADT system.

4.3 Spectral Analysis

In this subsection, we detail the choice of parameters for spectral analysis used to compute magnitude spectrograms from input audio snippets. This is relevant for both initialization and decomposition stages, and (needless to say that) we use the exact same parameters in both stages for obvious reasons.

We follow the choice of previous state-of-the-art algorithms which transformed input audio snippets into magnitude spectrograms by taking absolute values of the complex Short-Time Fourier Transform (STFT) spectrograms. This permits us to compare our results with majority of previous works and state-of-the-art systems. We used the unnormalized, linearly scaled magnitude spectrograms.

One can also find other kinds of Digital Signal Processing (DSP) pipelines used in the relevant ADT literature. These produce various inputs representations, such as Mel-Cepstrum Modulation Sepctrum (MCMS). We leave the exploration of influence of different input representations on our method to future work.

4.4 Initialization

To initialize our method, audio recordings with monophonic events of drum instruments are needed. We use IDMT-SMT-Drums [15] dataset, in which each of the drum instruments (KD, SD, and HH) has 95 audio recordings with isolated hits. For instance, one audio recording includes ≈ 10 hits on the drum instrument interlaced with short breaks of silence.

We initialize each drum instrument by following three steps procedure:

1. sample segmentation by onset detection (in case onset labels are not available) followed by extraction of fixed-length snippets starting at designated onset times,
2. calculating average time-frequency representation for each audio recording over its segmented samples,

3. calculating average time-frequency representation for each instrument over set of averages of given instrument computed in the previous step.

In first phase we summed up magnitudes per frame that resulted in vector of total energy for concrete magnitude spectrogram. It was followed by applying onset detection algorithm on this vector that filtered times of drum events according to rules described in 4.5.

In the second stage, these time information were used for picking spectral frames from magnitude spectrogram that were averaged. After this step average time-frequency representation of instrument for particular recording was created.

In third phase, we repeated previous two steps for many recordings and at the end we calculated average time-frequency representation of instrument over set of recordings by averaging all average representations per instrument.

4.5 Onset Detection

After factorization process, matrix $\mathbf{H}_D$ with potential drum events is processed by onset detection algorithm. Potential activation is considered as real activation if these terms are fulfilled [5]:

1. $\mathrm{F(n) = max(F(n\text{-}m), ..., F(n+m))}$
2. $\mathrm{F(n) >= mean(F(n\text{-}a), ..., F(n+m))} + \delta$
3. $\mathrm{n - n_{lp} > w}$

δ is a variable threshold. Peak must be the maximum value within a window size *2m + 1*. Also it has to exceed mean value plus threshold within a window size *2a + 1*. In addition it must have at least a distance *w + 1* to the last detected peak ($\mathrm{n_{lp}}$) [5].

5 Evaluation

We evaluate algorithms on DTD and DTM datasets. In addition to evaluating our method, we re-implement and evaluate the improved PFNMF algorithm with additional template adaptation that is described in [9], to enable for as fair comparison to the baseline as possible. We also experiment with different configurations that for DTD and DTM results are not reported in the ADT literature.

In this section we first detail the datasets we use along with configurations of experiments with the evaluated methods. Subsequently, we present and discuss the results of our evaluation.

5.1 Datasets

Two standard testing datasets are used for evaluation. The IDMT-SMT-Drums [15] dataset is used for DTD evaluation. It contains sampled, synthesized and recorded drum tracks. It comprises around 550 files with approximate length 15 seconds out of which 95 are simple polyphonic drum tracks. We use 14 audio recordings that represent 3 mentioned categories of drums.

Second dataset the ENST-Drums [16] is used for DTM evaluation. Recordings are available as solo drum and harmonic instrument tracks or as mixtures. Average length of one recording is approximetly 20 seconds. We use 12 audio recordings as mixtures.

5.2 Experiment Configuration

We have to set several important parameters that influence spectral analysis, PFNMFD or PFNMF methods and onset detection algorithm.

To compare our method with state-of-the-art systems we need to preprocess input signal in the same way. Therefore we use 44.1kHz sampling rate, Hanning window with size 2048 and hop length with the value 512 ($\frac{1}{4}$-th of the window size).

We also set parameters for improving the performance of algorithms. These parameters are:

- **opt** - optimization method. Values of this parameter are either *mu* (multiplicative updates) or *pg* (projected gradient)
- **lr** - learning rate. It contains values between (0;1) It is used only with optimization method *pg*
- **cf** - cost function. Values of this parameter are either *ed* (Euclidean distance) or *kl* (KL divergence)
- **sparse** - sparsity. It is applied on matrix $\mathbf{H}_D$ Theoretically, it can contain any real value. We use values within interval <0;1>. Higher values make matrix more sparse $\mathbf{H}_D$.
- **update** - additional update. It means whether matrix $\mathbf{H}_D$ should be additionally updated after factorization process.

Onset detection algorithm dispose of parameters that helps to filter out drum events from potential ones. Table 1 contains the set of parameters we empirically found to give good performance. This is by no means an optimal configuration, and more rigorous search (such as Bayesian optimization) might uncover any remaining potential for improvement.

Onset detection algorithm parameters					
pre_max	post_max	pre_avg	post_avg	threshold	wait
4	4	5	5	$0.1 \times \max(\mathbf{H}_{D[i]})$	4

Table 1: Parameters of onset detection algorithm with fixed values

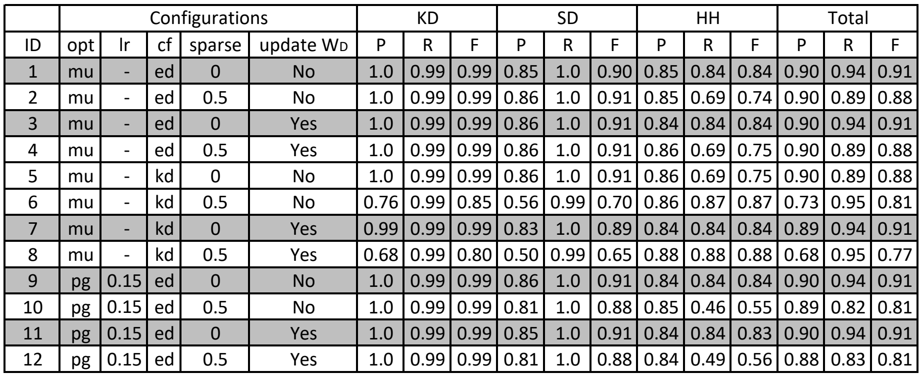
We set **pre_max** and **post_max** to 4 defining whether value of actual activation is greater than maximum value from previous 4 and following 4 activations.

pre_avg and **post_avg** were set to 5 that defined whether value of actual activation is greater than average value from previous 5 and following 5 activations. Parameter **wait** set to 4 denotes the minimal number of activations between 2 consecutive activations. Parameter **threshold** denotes the minimal value real activation needs to have. The value needs to be at least 0.1 times the strongest hit of all activations for particular drum instrument.

The standard information retrieval metrics Precision (P), Recall (R) and F-measure (F) are used to evaluate the algorithms. True positive, false positive and false negative onsets are identified by using $50ms$ tolerance window. We use the same tolerance as state-of-the-art systems.

5.3 Results

In this section, we present results of both PFNMNF baseline and our PFNMFD algorithm on both DTD and DTM problems. In all tables that follow, grey rows denote the best results in particular table.



	Configurations					KD			SD			HH			Total		
ID	opt	lr	cf	sparse	update W_D	P	R	F	P	R	F	P	R	F	P	R	F
1	mu	-	ed	0	No	1.0	0.99	0.99	0.85	1.0	0.90	0.85	0.84	0.84	0.90	0.94	0.91
2	mu	-	ed	0.5	No	1.0	0.99	0.99	0.86	1.0	0.91	0.85	0.69	0.74	0.90	0.89	0.88
3	mu	-	ed	0	Yes	1.0	0.99	0.99	0.86	1.0	0.91	0.84	0.84	0.84	0.90	0.94	0.91
4	mu	-	ed	0.5	Yes	1.0	0.99	0.99	0.86	1.0	0.91	0.86	0.69	0.75	0.90	0.89	0.88
5	mu	-	kd	0	No	1.0	0.99	0.99	0.86	1.0	0.91	0.86	0.69	0.75	0.90	0.89	0.88
6	mu	-	kd	0.5	No	0.76	0.99	0.85	0.56	0.99	0.70	0.86	0.87	0.87	0.73	0.95	0.81
7	mu	-	kd	0	Yes	0.99	0.99	0.99	0.83	1.0	0.89	0.84	0.84	0.84	0.89	0.94	0.91
8	mu	-	kd	0.5	Yes	0.68	0.99	0.80	0.50	0.99	0.65	0.88	0.88	0.88	0.68	0.95	0.77
9	pg	0.15	ed	0	No	1.0	0.99	0.99	0.86	1.0	0.91	0.84	0.84	0.84	0.90	0.94	0.91
10	pg	0.15	ed	0.5	No	1.0	0.99	0.99	0.81	1.0	0.88	0.85	0.46	0.55	0.89	0.82	0.81
11	pg	0.15	ed	0	Yes	1.0	0.99	0.99	0.85	1.0	0.91	0.84	0.84	0.83	0.90	0.94	0.91
12	pg	0.15	ed	0.5	Yes	1.0	0.99	0.99	0.81	1.0	0.88	0.84	0.49	0.56	0.88	0.83	0.81

Table 2: DTD evaluation of PFNMF algorithm with 1000 iterations

In Table. 2 we can see that many models reach the best result - 0.91 total *F-measure*. None of them use sparsity (sparsity value equals to 0). It could mean value 0.5 is too high. We can also notice that if sparsity is applied, mostly metric *Recall* for hi-hat drops significantly. The reason could be frequent occurrence of hi-hat in music. If this value is high, it could remove some hits.

Next, we can see that high sparsity value deteriorates results of algorithms using projected gradient optimization more than multiplicative update rules.

We further notice that hi-hat results are in general the worst. It makes sense since sound of hi-hat is the most diverse one (open hi-hat, closed hi-hat, etc.).

In Table. 3 PFNMFD algorithms using projected gradient for optimization clearly dominates. Moreover, the best PFNMFD algorithms get slightly better results - 0.92 total *F-measure* - than best PFNMF algorithms in DTD evaluation (previous figure).

	Configurations					KD			SD			HH			Total		
ID	opt	lr	cf	sparse	update W_D	P	R	F	P	R	F	P	R	F	P	R	F
1	mu	-	ed	0	No	0.88	0.98	0.91	0.77	1.0	0.86	0.96	0.88	0.91	0.87	0.95	0.90
2	mu	-	ed	0.5	No	0.88	0.97	0.91	0.76	0.99	0.85	0.96	0.88	0.92	0.87	0.95	0.89
3	mu	-	ed	0	Yes	0.87	0.97	0.91	0.76	1.0	0.85	0.96	0.89	0.92	0.87	0.95	0.89
4	mu	-	ed	0.5	Yes	0.88	0.97	0.91	0.74	1.0	0.84	0.96	0.89	0.92	0.86	0.95	0.89
5	mu	-	kd	0	No	0.88	0.97	0.91	0.68	0.98	0.78	0.95	0.93	0.94	0.84	0.96	0.88
6	mu	-	kd	0.5	No	0.89	0.98	0.92	0.68	0.98	0.78	0.95	0.93	0.94	0.84	0.96	0.88
7	mu	-	kd	0	Yes	0.89	0.97	0.92	0.68	0.98	0.78	0.95	0.93	0.94	0.84	0.96	0.88
8	mu	-	kd	0.5	Yes	0.89	0.98	0.92	0.68	0.98	0.78	0.95	0.93	0.94	0.84	0.96	0.88
9	pg	0.15	ed	0	No	0.96	0.99	0.97	0.80	1.0	0.87	0.96	0.89	0.93	0.91	0.96	0.92
10	pg	0.15	ed	0.5	No	0.97	0.99	0.98	0.70	1.0	0.80	0.93	0.41	0.52	0.86	0.80	0.77
11	pg	0.15	ed	0	Yes	0.96	0.99	0.97	0.80	1.0	0.87	0.95	0.89	0.92	0.90	0.96	0.92
12	pg	0.15	ed	0.5	Yes	0.97	0.99	0.98	0.69	1.0	0.80	0.94	0.36	0.48	0.87	0.79	0.75

Table 3: DTD evaluation of PFNMFD algorithm with 500 iterations and number of frames modelling spectral bases of instruments equaled to 10 (t=10)

Our method outperforms the NMF algorithms within DTD evaluation mentioned in [2]. We can see comparison results in Table 4.

Methods	F1(%)
PFNMFD	**92**
NMFD [14]	90
SANMF [10]	85
PFNMF [8]	71.5
AM1 [9]	72.5
AM2 [9]	69.2

Table 4: Detailed results comparison of DTD evaluation. We can see that our approach PFNMFD outperforms NMF-based solutions. We have to mention that we test our method on subset of 20 songs of IDMT-SMT-Drums dataset

Metrics also suggest, that PFNMFD models hi-hat better than PFNMF. We further observe, that if sparsity is applied on algorithm using projected gradient, performance drops drastically more than in PFNMF.

DTM is more complex than DTD therefore metrics dropped significantly. We can notice in Table. 5 that best PFNMF models reach 0.67 total *F-measure*. Too high sparsity deteriorate performance of models. But this time algorithms using multiplicative update optimization suffer. We can also notice that algorithms with PG optimization use very low learning rate. We assume that the reason is by a large amount of parameters (100 harmonic instruments) that effect error rate. Therefore learning rate must be really small.

	Configurations					KD			SD			HH			Total		
ID	opt	lr	cf	sparse	update W_D	P	R	F	P	R	F	P	R	F	P	R	F
1	mu	-	ed	0	No	0.65	0.83	0.71	0.78	0.62	0.65	0.57	0.53	0.50	0.67	0.66	0.62
2	mu	-	ed	0.5	No	0.69	0.78	0.71	0.84	0.44	0.53	0.61	0.08	0.14	0.71	0.44	0.46
3	mu	-	ed	0	Yes	0.65	0.84	0.72	0.78	0.62	0.65	0.56	0.52	0.50	0.67	0.66	0.62
4	mu	-	ed	0.5	Yes	0.68	0.78	0.71	0.86	0.42	0.52	0.65	0.13	0.19	0.73	0.44	0.47
5	mu	-	kd	0	No	0.65	0.85	0.72	0.80	0.64	0.67	0.65	0.69	0.62	0.70	0.72	0.67
6	mu	-	kd	0.5	No	0.59	0.38	0.42	0.67	0.14	0.21	0.58	0.33	0.40	0.60	0.28	0.34
7	mu	-	kd	0	Yes	0.63	0.85	0.71	0.84	0.63	0.68	0.65	0.67	0.61	0.71	0.72	0.67
8	mu	-	kd	0.5	Yes	0.60	0.37	0.41	0.66	0.15	0.23	0.58	0.30	0.36	0.61	0.28	0.33
9	pg	0.0005	ed	0	No	0.67	0.72	0.68	0.49	0.72	0.56	0.62	0.54	0.50	0.59	0.66	0.58
10	pg	0.0005	ed	0.5	No	0.67	0.72	0.68	0.50	0.72	0.56	0.63	0.57	0.52	0.60	0.67	0.59
11	pg	0.0005	ed	0	Yes	0.65	0.71	0.67	0.48	0.72	0.55	0.61	0.53	0.50	0.58	0.65	0.57
12	pg	0.0005	ed	0.5	Yes	0.67	0.73	0.68	0.49	0.71	0.56	0.63	0.55	0.51	0.59	0.67	0.58

Table 5: DTM evaluation of PFNMFD algorithm with 1000 iterations and 100 number of templates modelling harmonic instruments (r_H=100)

	Configurations					KD			SD			HH			Total		
ID	opt	lr	cf	sparse	update W_D	P	R	F	P	R	F	P	R	F	P	R	F
1	mu	-	ed	0	No	0.41	0.25	0.28	0.68	0.14	0.22	0.68	0.15	0.23	0.59	0.18	0.24
2	mu	-	ed	0.5	No	0.41	0.23	0.27	0.65	0.13	0.20	0.63	0.14	0.21	0.56	0.17	0.23
3	mu	-	ed	0	Yes	0.38	0.22	0.25	0.70	0.14	0.22	0.63	0.15	0.21	0.57	0.17	0.22
4	mu	-	ed	0.5	Yes	0.41	0.22	0.26	0.71	0.14	0.22	0.63	0.13	0.20	0.58	0.17	0.23
5	mu	-	kd	0	No	0.44	0.33	0.35	0.88	0.23	0.35	0.61	0.24	0.31	0.64	0.27	0.34
6	mu	-	kd	0.5	No	0.46	0.32	0.35	0.90	0.20	0.31	0.68	0.26	0.35	0.68	0.26	0.34
7	mu	-	kd	0	Yes	0.43	0.32	0.35	0.80	0.24	0.35	0.69	0.26	0.35	0.64	0.27	0.35
8	mu	-	kd	0.5	Yes	0.43	0.32	0.33	0.84	0.23	0.34	0.70	0.27	0.36	0.65	0.27	0.39
9	pg	0.00001	ed	0	No	0.77	0.82	0.78	0.78	0.67	0.67	0.53	0.15	0.19	0.69	0.55	0.55
10	pg	0.00001	ed	0.5	No	0.77	0.82	0.78	0.79	0.66	0.68	0.56	0.11	0.14	0.70	0.53	0.53
11	pg	0.00001	ed	0	Yes	0.77	0.82	0.78	0.78	0.67	0.8	0.48	0.12	0.15	0.67	0.54	0.53
12	pg	0.00001	ed	0.5	Yes	0.77	0.82	0.78	0.79	0.67	0.68	0.55	0.11	0.13	0.70	0.53	0.53

Table 6: DTM evaluation of PFNMFD algorithm with 500 iterations, t = 10 and rh = 100

Usage of PFNMFD algorithms in DTM evaluation is a little bit worse. In Table. 6 we can see that the best model reaches 0.55 *F-measure*. In general models using PG optimization reach better results then algorithms using MU optimization. We don't outperform state-of-the-art NMF models in DM evaluation mentioned in [2].

Also, from all of the experiments, we notice that additional updating of $\mathbf{W}_D$ doesn't improve performance.

6 Conclusion

In this paper we analyzed DTD and DTM research topics. We proposed PFNMFD algorithm that is combination of PFNMF and NMFD. Our algorithm outperformed NMF-based solutions in DTD evaluation of state-of-the-art systems. We used different configurations of algorithm during evaluation process. We have acquired some knowledge that we can take into account in future work.

Sparsity values were too high. We can set it to smaller values and see the results - especially with algorithms using PG optimization. Also we could set sparsity separately for hi-hat since too high sparsity value had mostly bad impact. Other finding is that using PG optimization is powerful - it outperformed multiplicative update optimization several times. We could improve PG optimization with adaptive learning rate or momentum. Further we could try set higher t value that defines number of frames modelling templates of instruments in dictionary matrix. We could also try to set t per drum instrument since each of them has different decay length. Since specific additional $\mathrm{W_D}$ update didn't help, we could try decomposition with semi-adaptive NMF bases - SANMF [10]. In matter of fact updates used in our evaluation didn't have any constraints. It can be problem because without limit they are not pushed to model sounds of drum instruments but mixture of multiple instruments. In future we could also focus on modelling drum playing technique by incorporating neural networks that would model techniques for each drum instrument.

References

1. C. Hawthorne, E. Elsen, J. Song, A. Roberts, I. Simon, C. Raffel, J. Engel, S. Oore, and D. Eck, "Onsets and Frames: Dual-Objective Piano Transcription," *ISMIR*, 2018.
2. C. Wu, C. Dittmar, C. Southall, R. Vogl, G. Widmer, J. Hockman, M. Müller, and A. Lerch, "A review of automatic drum transcription," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, pp. 1457–1483, Sep. 2018.
3. C. Southall, R. Stables, and J. Hockman, "Automatic drum transcription using bi-directional recurrent neural networks," in *ISMIR*, 2016.
4. C. Southall, R. Stables, and J. Hockman, "Automatic drum transcription for polyphonic recordings using soft attention mechanisms and convolutional neural networks," in *ISMIR*, 2017.
5. R. Vogl, M. Dorfer, and P. Knees, "Drum transcription from polyphonic music with recurrent neural networks," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 201–205, March 2017.
6. R. Vogl, M. Dorfer, G. Widmer, and P. Knees, "Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks," in *ISMIR*, 2017.
7. A. R. C. Jacques, "Automatic drum transcription with convolutional neural networks," in *21th International Conference on Digital Audio Effects*, September 2018.
8. C.-W. Wu and A. Lerch, "Drum transcription using partially fixed non-negative matrix factorization," 09 2015.
9. C.-W. Wu and A. Lerch, "Drum transcription using partially fixed non-negative matrix factorization with template adaptation," 10 2015.
10. C. Dittmar and D. Gärtner, "Real-time transcription and separation of drum recordings based on nmf decomposition," in *DAFx*, 2014.
11. H. Lindsay-Smith, S. McDonald, and M. Sandler, "Drumkit transcription via convolutive nmf," *15th International Conference on Digital Audio Effects, DAFx 2012 Proceedings*, 01 2012.

12. J. J. Burred, “Detailed derivation of multiplicative update rules for nmf,” 2017.
13. H. Seung and D. Lee, “Algorithms for Non-negative Matrix Factorization,” in *Neural Information Processing Systems (NIPS)*, pp. 1828–1832, 2000.
14. P. Smaragdis, “Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs,” in *Independent Component Analysis and Blind Signal Separation* (C. G. Puntonet and A. Prieto, eds.), (Berlin, Heidelberg), pp. 494–499, Springer Berlin Heidelberg, 2004.
15. C. Dittmar and D. Gärtner, “Real-time transcription and separation of drum recordings based on nmf decomposition,” in *DAFx*, 2014.
16. O. Gillet and G. Richard, “Enst-drums: an extensive audio-visual database for drum signals processing.,” pp. 156–159, 01 2006.

Effective Enumeration of Selected Graph Characteristics

Roman Hroš[*]

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
xhros@stuba.sk

Abstract. In this paper we show how to calculate the number of cycle subgraphs with a certain domination number. Specifically, they are spanning subgraphs, which are exponentially numerous, and therefore we have created an efficient algorithm that calculates the domination number only on $\frac{1}{4}$ of them and reveals it on the other subgraphs based on the similarity between them. Using this method, we are able to calculate the number of subgraphs with a certain domination number on a cycle of order 40 in about 4 hours. If we have these numbers, we can calculate the mean value of domination number on a random graph, which we can model, for example, a communication network.
These numbers can be arranged in two dimensional integer table in which we discovered the relationships from which we derived the formulas to calculate domination numbers of cycle subgraphs. The first is recurrent, where we need to know the number of subgraphs with a certain domination number on 3 smaller cycles in order to be able to calculate the following. The second relationship refers to the mean of the domination number of the cycle subgraphs if the probability of edges is specified.

Keywords: Domination number · Cycle graph · Spanning subgraphs · Integer sequence.

1 Introduction

The domination number and the dominating set [1, 6] are one of the essential characteristics of the graph and domination problem is NP-complete [2]. The dominating set Z for a graph G is $Z \subseteq V(G)$ such that every vertex not in Z must be connected by an edge to at least one vertex of Z. We want to find the smallest possible such set, also called minimum dominating set, because this set well models the optimal distribution of some objects in the graph, which is useful for various problems. The domination number [3, 6] is the cardinality of this minimum dominating set.

[*] Master study programme in field: Informatics
Supervisor: Dr. Dušan Bernát, Institute of Computer Engineering and Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

This work deals with calculating the domination number on each spanning subgraph of a cycle of order n. Since the number of subgraphs is 2^n, it is necessary to simplify this problem as much as possible in order to be able to calculate the domination number of the subgraphs of large cycle in a reasonable time. These subgraphs model very well different networks or other relationships, because in the real world, lines in the network or relationships between entities can go wrong, and that is exactly what the missing edges in the graph do. If we can calculate the domination number of each subgraph of the cycle, then we can calculate, for example, the mean value of domination number on a random graph if we know the probability of the edges appearing in the graph. This is a more theoretical problem, because the cycle graph does not occur much in the real world. But if we find a solution to this problem in a simple way, it can help us with more complicated graphs that arise from the cycle, for example, a Cartesian product with another graph or in another way.

We will show how we simplified this problem four times due to the similarity of subgraphs. Then we will show the patterns that have been discovered to solve this problem even more efficiently. We were inspired by [4] to search for patterns. In this article authors show recurrence formula for domination polynomial of cycles.

2 Naive approach

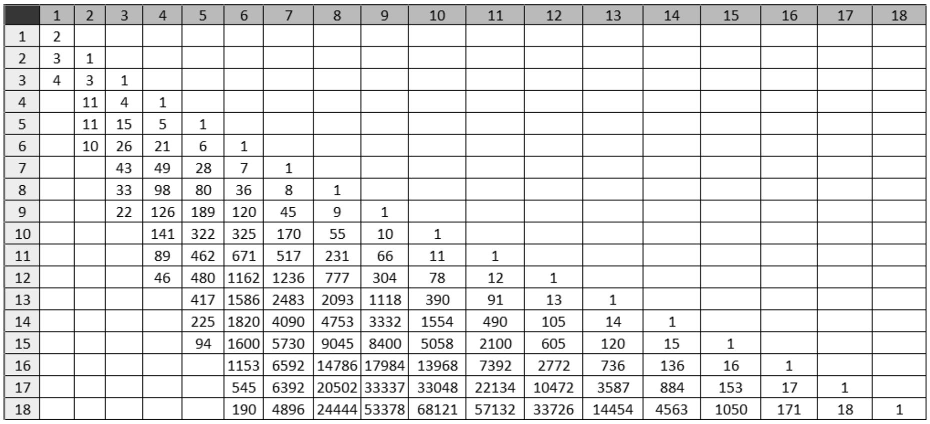
2.1 Representation of subgraphs

We use bit encoding for the cycle subgraphs. For cycle of size n we need n bits because we have 2^n subgraphs. The set bit represents the edge in the subgraph and the zero bit represents missing edge. In the decimal system, the subgraphs will be represented by the numbers 0 to $2^n - 1$.

2.2 Algorithm

We can see that all subgraphs are made up only of paths or isolated vertices. The isolated vertex must be in the minimum dominating set because it is not connected with any other vertex. The domination number of path can be calculated by the formula $\lfloor \frac{n+2}{3} \rfloor$ [3], where n is number of vertices.

The naive algorithm goes through all the subgraphs and go through all the bits in each subgraph. We know that 2 zeros in a row form an isolated vertex and the ones in a row form a path. For each path we calculate the domination number and sum it together with isolated vertices. In addition, we have to deal with subgraphs where their first and last bit contain simultaneously zeros or ones. So we get a domination number for the whole subgraph. We will write this result to the Table 1. The rows represent the cycle order and the domination number represents columns. According to the domination number of the subgraph, we increment the corresponding cell in this table.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	2																	
2	3	1																
3	4	3	1															
4		11	4	1														
5		11	15	5	1													
6		10	26	21	6	1												
7			43	49	28	7	1											
8			33	98	80	36	8	1										
9			22	126	189	120	45	9	1									
10				141	322	325	170	55	10	1								
11				89	462	671	517	231	66	11	1							
12				46	480	1162	1236	777	304	78	12	1						
13					417	1586	2483	2093	1118	390	91	13	1					
14					225	1820	4090	4753	3332	1554	490	105	14	1				
15					94	1600	5730	9045	8400	5058	2100	605	120	15	1			
16						1153	6592	14786	17984	13968	7392	2772	736	136	16	1		
17						545	6392	20502	33337	33048	22134	10472	3587	884	153	17	1	
18						190	4896	24444	53378	68121	57132	33726	14454	4563	1050	171	18	1

Table 1. Number of cycle spanning subgraphs with domination number by column and the order of cycle is per row. Cycle graph order is from 1 to 18.

3 Isomorphic subgraphs

We may notice some similarities in the subgraphs and their binary representations. For example, subgraphs that have one edge logically all have the same domination number. Also, subgraphs that have 2 missing edges equidistant from each other have the same domination number. We can detect this similarity by bit shifts. When shifting bits we get groups of isomorphic subgraphs that have the same domination number, so we can calculate this domination number on one subgraph from this group and we know that all subgraphs in this group will have the same domination number. We do not need to consider a subgraph with all zero bits or subgraph in which all bits are ones, because domination number is trivial but at the end we add it to result.

```
 1,   2,   4,   8, 16                          30, 29, 27, 23, 15
 3,   6,  12,  24                              28, 25, 19,  7
 5,  10,  20                                   26, 21, 11
 7,  14,  28                                   24, 17,  3
 9,  18                                        22, 13
11,  22                                        20,  9
13,  26                                        18,  5
15,  30                                        16,  1
```

The example shows bit shifts for C_5 cycle. We can see subgraphs represented as decimal numbers. In binary representation we shift ones in the left side of example and rotate them in the right side of example. The rows on the left begin with odd numbers 1 to 15. The rows on the right end with the same numbers, but in a different specific order. All subgraphs that appear in a line that either ends or begins with the same number have the same domination number. On the left we can calculate the number of subgraphs based on the number of shifts and on the right we can do it similarly.

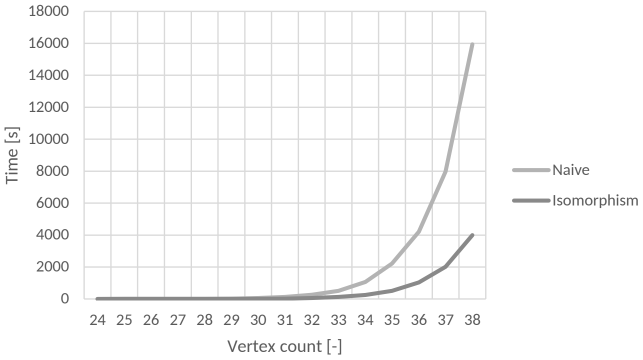


Fig. 1. The runtime of the naive and improved algorithm with isomorphism.

Algorithm 1: Effective enumeration of number of cycle subgraphs split according to their domination number

```
Data: $n$ - order of cycle
Result: Number of subgraphs of cycle of order $n$ split according to
        their domination number
$right \leftarrow 0,\ jump \leftarrow 4,\ start \leftarrow 2$;
while $start \leq half$ do // half = $2^{n-1}$
    $i \leftarrow start - 1$;
    while $i < half$ do
        $left = \lceil log_2(\frac{\#subgraphs}{i}) \rceil$;          // #subgraphs = $2^n$
        $domNumber \leftarrow CalculateDomNumber(i, n)$;
        $table[n, domNumber] \leftarrow table[n, domNumber] + left + right$;
        $i \leftarrow i + jump$;
    $right \leftarrow right + 1,\ jump \leftarrow 2 \cdot jump,\ start \leftarrow 2 \cdot start$;
```

We will go through the subgraphs in the algorithm according to the pattern on the right side of the example in sequence 1, 5, 9, 13 → 3, 11 → 7 → 15. It is $\frac{1}{4}$ of all subgraphs of C_5 cycle because it is $(4+2+1+1)/32$. This holds for each cycle because we can arrange these subgraphs as decimal numbers in the same way as in the example for any cycle order. For example for C_6 we get $(8+4+2+1+1)/64$ and in the same way for others. Figure 1 shows comparison and exponential time complexity of runtime between naive and improved algorithm.

4 Expected domination number of random cycle spanning subgraph

If the row in the Table 1 is filled, we can calculate the mean value of domination numbers of all subgraphs as expected domination number. We do this with an arithmetic mean, because we know how many subgraphs have what a domination number. We can denote number of spanning subgraphs of cycle of order n with domination number m as $f(n, m)$.

$$E(\gamma(S_n)) = \frac{1}{2^n} \sum_{m=\lceil n/3 \rceil}^{n} mf(n, m) \tag{1}$$

S_n is set of all spanning subgraphs of cycle C_n and $\gamma(S_n)$ is set of domination numbers of all these subgraphs. In this case, each subgraph has same probability, or each edge in the subgraph has a probability of occurrence 0.5. Probability of subgraph with k edges, where occurrence of edge has a probability p is $p^k(1-p)^{n-k}$. We can see for $p = 0.5$ it is always 0.5^n, what we can also see in Equation 1. If we wanted to change this probability and calculate the mean value of the domination number, then we will have to split the subgraphs according to the number of edges, which is not a problem, because it is enough to calculate the ones in its binary representation. Based on the number of edges we determine the probability of each subgraph. By this probability we weigh its domination number and if we do this over all subgraphs, we get the mean value of the domination number for the random cycle subgraph with the specified probability of edge occurrence.

$$E(\gamma(S_n), p) = \sum_{k=0}^{n} [\ p^k(1-p)^{n-k} \cdot \sum_{m=\lceil n/3 \rceil}^{n} mf(n, m, k)\ ] \tag{2}$$

Here $f(n, m, k)$ is number of subgraphs of cycle C_n with k edges and domination number m.

5 Recurrence formula

In the Table 1, the sum of the row is 2^n. The sum of the column is $9 \cdot 4^{m-1}$, where m is the column index. Other relationships between the number of subgraphs on diagonals can also be found in this table. We express each row as a generating function for example for C_5 we get $D(C_5, x) = x^5 + 5x^4 + 15x^3 + 11x^2$. Similarly it was done in [4] for domination polynomial. Exponents are domination numbers and coefficients are numbers of subgraphs. We discovered a relationship between these polynomials.

$$g(n) = \begin{cases} 2(x^{\frac{n}{3}} - x^{\frac{n}{3}+1}) = 2(1-x)x^{\frac{n}{3}} & n \ \% \ 3 = 0 \\ 0 & n \ \% \ 3 = 1 \\ x^{\frac{n+1}{3}} - x^{\frac{n+1}{3}+1} = (1-x)x^{\frac{n+1}{3}} & n \ \% \ 3 = 2 \end{cases} \tag{3}$$

$$D(C_n, x) = x[D(C_{n-1}, x) + D(C_{n-2}, x) + 2D(C_{n-3}, x)] + g(n) \tag{4}$$

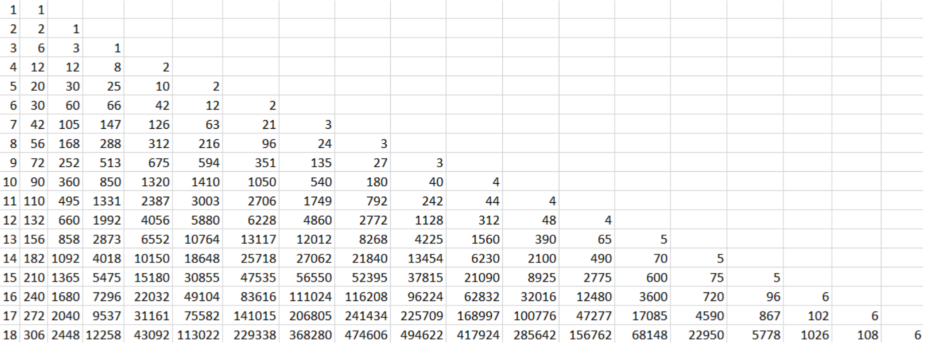
This formula is easy to apply directly in the table. Each row can be calculated by combining the previous three. For each entry, just look at the three values above it one column to the left.

$$f(n, m) = 2f(n-3, m-1) + f(n-2, m-1) + f(n-1, m-1) \tag{5}$$

It holds for every entry in row, which has index%3 = 1. For row index%3 = 2, for first entry we need add 1 and for second entry subtract 1. Other entries are calculated in the same way. For index%3 = 0, for first entry we add 2 and for second we subtract 2.

6 Formula for mean value of the domination number with probability of edge presence

In Equation 2 we need three dimensional table for numbers of subgraphs. We can modify this table by precalculating the second sum in this equation. We get the two dimensional Table 2. The rows are order of cycle and the columns represent the number of edges in subgraphs. The value in a given cell is a multiplied number of subgraphs with a domination number. From Table 2 we can calculate the mean value of the domination number of random cycle of order n with probability of edge presence p according to the following formula, where $h(n,k)$ is the cell value in Table 2.



1	1																	
2	2	1																
3	6	3	1															
4	12	12	8	2														
5	20	30	25	10	2													
6	30	60	66	42	12	2												
7	42	105	147	126	63	21	3											
8	56	168	288	312	216	96	24	3										
9	72	252	513	675	594	351	135	27	3									
10	90	360	850	1320	1410	1050	540	180	40	4								
11	110	495	1331	2387	3003	2706	1749	792	242	44	4							
12	132	660	1992	4056	5880	6228	4860	2772	1128	312	48	4						
13	156	858	2873	6552	10764	13117	12012	8268	4225	1560	390	65	5					
14	182	1092	4018	10150	18648	25718	27062	21840	13454	6230	2100	490	70	5				
15	210	1365	5475	15180	30855	47535	56550	52395	37815	21090	8925	2775	600	75	5			
16	240	1680	7296	22032	49104	83616	111024	116208	96224	62832	32016	12480	3600	720	96	6		
17	272	2040	9537	31161	75582	141015	206805	241434	225709	168997	100776	47277	17085	4590	867	102	6	
18	306	2448	12258	43092	113022	229338	368280	474606	494622	417924	285642	156762	68148	22950	5778	1026	108	6

Table 2. Adjusted table from Equation 2, that second sum is precalculated. Cycle graph order is from 1 to 18.

$$E(\gamma(S_n),p) = \sum_{k=0}^{n} p^k (1-p)^{n-k} h(n,k) \tag{6}$$

We have discovered the following relationships in this table. For each row, we just need to know the previous one.

$$\begin{aligned}
h(n,k) &= n \cdot \frac{h(n-1,k)+h(n-1,k-1)}{n-1}; \quad && 0 < k < n-1 \\
h(n,k) &= n; && k = 0 \\
h(n,k) &= n \cdot \lceil \frac{n}{3} \rceil; && k = n-1 \\
h(n,k) &= \lceil \frac{n}{3} \rceil; && k = n
\end{aligned} \tag{7}$$

When we have these relationships, we can calculate the mean value of the domination number on large cycles. For each cycle, we express mean of the domination number generally with the variable p. After simplifying the resulting polynomials, we found a relationship between them. This relationship after simplification can be expressed as following equation.

$$E(\gamma(S_n), p) = n\frac{1 - p^{3\lceil n/3\rceil}}{p^2 + p + 1} + \lceil n/3\rceil p^n \tag{8}$$

If we divide Equation 8 by n we get relative mean domination number. We can solve a limit in infinity if we want know it on infinite cycle.

$$\lim_{n\to\infty} \frac{n\frac{1-p^{3\lceil n/3\rceil}}{p^2+p+1} + \lceil n/3\rceil p^n}{n} = \lim_{n\to\infty} \frac{1 - p^{3\lceil n/3\rceil}}{p^2 + p + 1} + \lim_{n\to\infty} \frac{\lceil n/3\rceil p^n}{n} = \frac{1}{p^2 + p + 1} \tag{9}$$

Equation 9 holds for $0 < p < 1$, but in result, we can extend it for $p = 0$ and $p = 1$. We can see, for example for $p = 0.5$, on random spanning subgraph of infinite cycle, in minimum dominating set must be $\frac{4}{7}$ of vertices.

7 Conclusion

In this paper we show algorithm for enumerating domination numbers of all cycle spanning subgraphs. Algorithm calculates domination number just on $\frac{1}{4}$ of subgraphs and therefore is effective. We think it can still be improved, but we do not need to because we found recurrence relation for it.

This relation is not so simple and challenge is to find explicit formula as authors did in [5] for recurrence relation from [4]. Secondly, we found formula for mean value of domination number of random cycle graph with probability of edge presence. We think similar formulas can be found for other simple graphs as for example path graph. This work has no direct application in real problems, but in more complex graphs which can arise from these simple graphs by for example Cartesian product can be useful, because than we can better estimate these described characteristics.

References

1. Palúch, S.: Algoritmická teória grafov. 1st edn. EDIS ŽU, Žilina (2008)
2. Karp, M. R.: Reducibility Among Combinatorial Problems. In: R. E. Miller and J. W. Thatcher (eds.) Complexity of Computer Computations. New York: Plenum, pp. 85–103. (1972)
3. Bray, Nicolas and Weisstein, Eric W. "Domination Number." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/DominationNumber.html
4. Alikhani, S., Peng, Y.: Dominating sets and domination polynomial of cycles. In: Global Journal of Pure and Applied Mathematics, vol. 4, no. 2, 2008, pp. 151–162.
5. Arocha, J. L., Llano, B.: The number of dominating k-sets of paths, cycles and wheels. In: arXiv:1601.01268 [math.CO] (2016)
6. Haynes, T. W., Hedetniemi, S. T., Slater, P. J.: Fundamentals of Domination in Graphs. 1st edn. Marcel Dekker, New York (1998)

Cyber Security – Penetration Testing with Kali Linux

Marián Hlavačka*

Secondary Vocational School
Rastislavova 332, 972 71 Nováky, Slovakia
marianhlavacka@gmail.com

Abstract. Cyber security is a necessity, nowadays. Everyone uses smart devices every day and so many people do not realize which threats can come with them. In this paper, we focused on cyber security and penetration testing to learn how attacks occur and how to protect ourselves against them, only for educational purposes. We worked withKali Linux where, shown how to use this operating system, which tools it containsand based on this, we performed well-known attacks and protected virtualmachines against them. We also installed Kali Linux on arm devices like Raspberry Pi andNexus5x.

Keywords: Penetration Testing, Cyber Security, Ethical Hacking, Kali Linux

1 Introduction

Privacy and data security are difficult to achieve, especially nowadays. We use many smart devices, that make our life easier, but they also make it easier for cyber attackers to gain access to our data. Cyber attackers are people who want to monitor or steal some sensitive data from a person or an organization. Their motivation is very often a financial one. [1].

Cyber security is a field of study focused on the threats that occur in a computer world, network, programs, systems and so on. It applies or should be applied both in big or small organizations or to individuals [1].

Penetration testing is a method for testing system in a computer, a computer network or a web application with goal of finding any vulnerabilities that attackers would exploit. Penetration testers use the same tools as attackers, also called black-hat hackers, because those tools are widely available. However, it also helps penetration testers to better understand how those tools can be used against organizations [2].

* Secondary Vocational School graduation class in field: Mechatronics
Supervisor: Alexander Valach Institute of Computer Engineering and Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

As described in section 3, we worked with Kali Linux where we realized several network attacks and protections against them. Those attacks were performed on operating system (OS) Windows 10 and vulnerable virtual machine called *Metasploitable*. All described scenarios were realized using virtual machines. The reason is simple, if something goes wrong in guest operating system (*Metasploitable*), it will not affect a host operating system.

We also made our own schemes to illustrate a basic scenario of every attack that we performed. We described how to perform the tests step-by-step. Only for educational purposes.

The rest of the paper is organized in the following way. Section 2 describes existing vulnerabilities or problems of security. Section 3 covers detailed description of common threats and exploitations. Finally, we introduce the way to perform tests on the systems. Sections 4 concludes the paper.

2 Related works

In the world where every device is somehow connected to the Internet, information is more vulnerable than ever before [2]. It is not surprising that in the last years, security and exploitation of web applications and network devices on the Internet is one of the most discussed topics [3].

Usage of web applications and Internet without any protection leads to security risks. This causes an increase of cyber threats for many companies or organizations and that is something they cannot afford, because they don't want to lose their data. And here comes security field called penetration testing [4].

Vulnerability assessment and penetration testing techniques may help organizations to find security loopholes [2]. The penetration testers reveal new vulnerabilities and can expose threats based on them. Also, knowledge of penetration testers should be permanently updated to reflect the latest trends.

The smart attackers can develop a thoughtful plan how to bypass the security of applications, the network devices, computer systems and that is the reason why there are new vulnerabilities added into the list, which should be addressed to the organization's security department or security operation center, so they can mitigate the risk of an attack [3]. Regular updates of systems or applications (usually containing latest security fixes) can decrease the chance of being exploited by attackers [3].

3 Kali Linux

Kali Linux (we were using version 2019.3) is a debian-based Linux distribution aimed at advanced penetration testing and security auditing. Kali contains several hundred tools which are geared towards various information security tasks, such as penetration testing, security research, computer forensics and reverse engineering [4].

Kali Linux is developed, funded and maintained by Offensive Security, a leading information security training company [4]. Among its well-known tools are

Wireshark, Aircrack-ng, John the ripper and Nmap. We also used Kali Linux with other devices such as Raspberry Pi 4 and Nexus 5x.

3.1 Virtualbox and Wi-Fi adapters

Virtualbox (we were using version 6.0.14) is a multi-platform software, that enables several guest OSs running on the same host at the same time. Since those operating systems are virtual, their performance depends on resources they are provided with. For every OS we can set various parameters such as: memory (RAM), number of processor cores, network adapters, USB ports and others [5]. We have been using Windows 10 as a guest machine, because this OS is widely used today. Our goal was to exploit some vulnerabilities in a commonly used device.

Virtualbox has an optional package (Extension pack) which is needed if we want to connect additional devices to a VM, e.g. USB keys, a web camera or a microphone. The advantage of penetration testing in virtual environment is that when something will break in guest OS, e.g. we download malware or we will be under attack, a host operating system will not be affected. To prevent data loss, we can regularly create snapshots, which save an actual state of OS for later restoration. This function allows us to return into a particular system state [5].

Another way to perform these attacks would be to use physical machine with KVM (Kernel Virtual Machine). In that case you can also use a sandbox environment where you can capture malicious files. This is more common way to detect malwares [6].

Very important for penetration testing is to choose a correct WLAN adapter. Suitable adapter should support a network monitor mode and a packet injection features. A network adapter could operate in 2 modes [7]:

- Managed – default mode, in which an adapter captures packets destined to its own MAC address. (e.g. a computer) [7].

- Monitor – capture all the packets within our range, regardless of the destination MAC address [7].

One of the most suitable adapters for penetration testing is Alfa Network AWUS036NHA (2.4 GHz). It does not require installing additional drivers. It is also the adapter we have been using because it supports monitor mode and packet injection features. This adapter works as plug and play because it contains Atheros chipset which has built-in drivers in Kali Linux. AWUS036NHA supports 2.4 GHz band which is used by several Wi-Fi standards (IEEE 802.11g or 802.11ac). The 5 dBi antenna can also be replaced with another one, e.g. 9 dBi antenna.

3.2 Threats and vulnerabilities

Attackers can use different techniques for different purposes. Among well-known techniques are traffic monitoring, traffic redirection and device exploitation which is possible due to device or protocol vulnerabilities. This is the reason why protection exists in form of antivirus software, virtual private networks (VPN) and program that can find out another entity is monitoring the device and so on. In this paper, we summarize several well-known attacks:

A. DoS Attack - denial of service is an attack that targets a single device or a service with purpose of disengaging it. In this paper, we present a deauthentication attack, targeting communication between user and wireless access point. We will focus on a certain MAC address of the router and disconnect that client, broadcasting large number of packets. The utilized tool is called *aireplay-ng* and we will also need it later to perform a brute force attack [8].
B. Brute-Force Attack + Crunch - well-known attack to crack WPA2 protection or to crack other passwords. The main idea is that an attacker is trying to find a correct password using a wordlist, which is a file that contains possible passwords. This attack is considered long-time and does not ensure a successful password match. Its success depends on the size of the wordlist and another factors such as language that password resembles or if the password has been exposed to the public before and added to the wordlist [9]. We can guess key derived from password using a *crunch method*, in which we create our own wordlist.
C. Man-In-The-Middle (MITM) attack - attack with purpose of capturing traffic between devices. An attacker gets to the middle of the connection and serve as a converter. It can be performed only when we are able to capture a communication between two devices. If it works out, an attacker is able to see whole communication. One of the methods to perform this attack is called ARP spoofing. An attacker gets between client and router and all requests and responses will go through attacker's device [10]. To accomplish this, we will use *bettercap framework*. By modifying file (caplets - *hstshijack*), we were able to downgrade secure protocol https and read the traffic from the secured sites. *Bettercap* has many functions. For example, you can attack using other types of MITM e.g. DNS spoofing in which you are monitoring a network traffic, web sites or you can redirect victim on your own web site that contains some malicious code or a backdoor.
D. DNS attack - with *bettercap framework*, we can perform other attacks too. One of them is DNS spoofing. It can be processed in the following way. When an attacker becomes MITM, he or she can monitor our whole traffic including information about sites we visit and can redirect us to a fake site which contains a backdoor or some malicious code [11]. Kali has built-in web server called *apache2* and we can use it as a web site where victims will be redirected.
E. Metasploit Framework - *metasploit* (msfconsole) is a tool to perform, test and develop exploits for different applications and operating systems. It is one of the most widely used tools for penetration testing in Kali Linux. By the term exploit, we mean a type of malware that takes advantage of certain bugs or vulnerabilities, which cybercriminals use to gain illicit access to a system [12].
F. Backdoor - malicious type of code, which can take control over the infected computer and execute commands in it. The goal is to take control of account with highest level of privileges to execute any type of command. It allows an attacker to access machine bypassing common authentication, which would normally prevent users

unauthorized command execution [13]. To create backdoor and test this attack there are many frameworks available, e.g. a *Veil* or *Empire*.

3.3 Exploiting vulnerabilities

In this paper, we performed a brute-force, man in the middle and backdoor attacks on Windows 10. We also mention simple countermeasures. The first attack that we focused on is called a brute-force which is offline attack on wireless networks with WPA2 protection. First, we needed to capture a handshake which is occurring when a wireless device is connecting to access point. We performed deauthentication attack presented on Figure 1 using *aireplay-ng* tool for a short time to disconnect a user from Wi-Fi and wait until this user connects back automatically. After that, we captured handshake that contains 4 packets which are sent between a user's device and a wireless access point. Then we needed a wordlist that contains possible passwords. We have selected *a crunch method* that is a tool which combine letters, symbols, numbers to derive a possible password. By this way, we created our own wordlist that contained combined words without correct password. We wrote a real password to this wordlist and then we used tool called *aircrack-ng* that was comparing handshake and wordlist to find a password. This is called a message integrity code (MIC), because we were comparing passwords in wordlist until we generated a same MIC as we captured in handshake (Figure 1). We figured out that the best way to avoid this is to choose a strong password (combination of lower and upper case letters and numbers along with special characters). However, you can use advanced security in your router and enable DoS protection against deauthentictation attack (Figure 1). Another problem arises when using WPS (Wi-Fi Protected Setup) for quick connection, because WPS uses 8-character PIN that can be brute-forced by tool Reaver. To avoid this, you should lock your WPS.

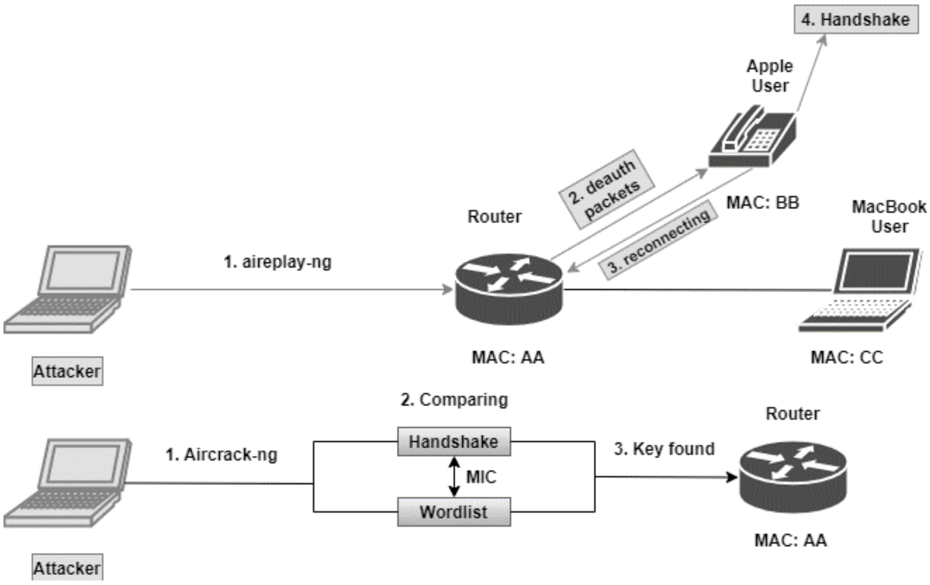


Figure 1 – The scheme of WPA2 cracking

Another attack that we performed is ARP spoofing (Figure 2) which is a form of man in the middle attack. We used framework called *bettercap* (v 2.23) where we were using *net.probe on* command to perform ARP spoofing. After that we chose Windows 10 IP´s by command *net.show* as the target and performed command *net.sniff* to spoof all network traffic. In Windows 10 we were using google.sk, de, ie, because this Google has HTTPS protocol which we downgraded by modifying caplet *hstshijack* that injects HTML and JS files within a payload. We used bettercap v2.23, because it has not any problems in comparison with pre-installed version which contains caplet hstshijack that did not work. In conclusion we had to download and replace hstshijack caplet from github because this one is modified to downgrade HTTPS requests (the path is /usr/share/bettercap/caplets). Then in bettercap we had to write *hstshijack/hstshijack* and press enter to downgrade HTTPS. After the command was executed the whole traffic was sent using HTTP in a plain text along with captured passwords.

One of the methods to protect against ARP spoofing is a program called *XArp* that draws attention on MAC addresses assigned to IP addresses. Also, we used *Wireshark* with protocol *ARP/RARP* that detects requests for certain time and if we are under ARP spoofing, we are notified that there is a duplicated IP address configured on network.

For *hstshijack* we found the protection plugin called *https everywhere* that enforces use of HTTPS, the same result as with HSTS. Another solution was to use a VPN which creates an encrypted tunnel between user and VPN server, thus an attacker is not able to downgrade HTTPS protocol or to read users' activities. To achieve better security, you can use VPN with https everywhere by which you make 2 layers of security, VPN and https everywhere that will protect you also when not connected to VPN.

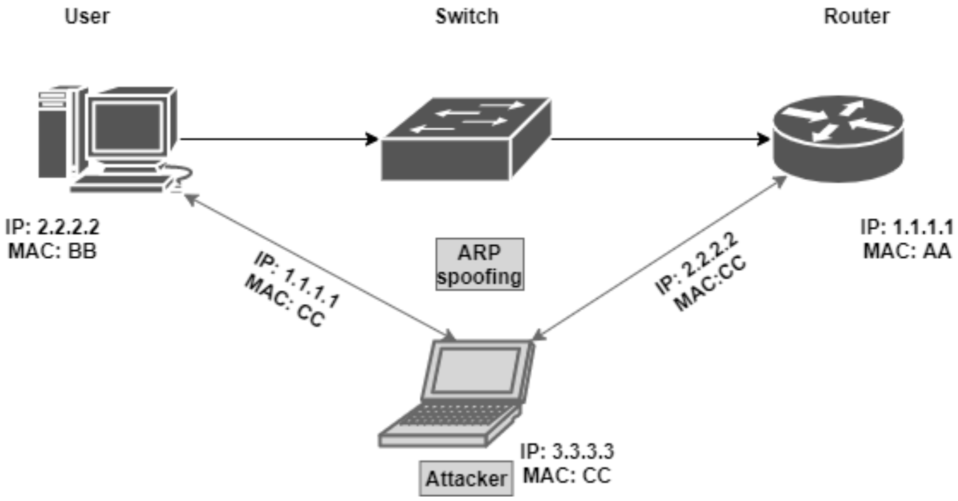


Figure 2 – The scheme of ARP spoofing attack (MITM)

Third attack that we performed is a fake update containing a backdoor (Figure 3). At first, we needed to download *Veil framework* that provides several methods for bypassing an antivirus software. After installation we launched *Veil* and it showed 2 categories: *Evasion* and *Ordnance*. Evasion serves for generating undetectable backdoors and Ordnance serves for generating undetectable payloads used by Evasion. We chose the first one, Evasion, and then we applied payload *14. go/meterpreter/rev_http.py*. Then we modified backdoor to be more unique to bypass antivirus. We had to setup *LHOST* to our IP, because this is the IP address of the device where victim will be connected. We were using *rev_http.py* and in this case we can bypass the firewall, because *rev* stands for reverse and it means that it opens a port in my computer and creates a connection between an attacker and a victim. Next, we setup *LPORT* to *8080* (this port is used by many web servers; it is not suspicious so it should bypass firewalls) and *LPORT* is the port that will be opened. Also, we changed values of *PROCESSORS* to *1* and *SLEEP* to *10*.

We had to download a program called *Evilgrade* running on the Kali machine which serves as a fake webserver from where a backdoor was later downloaded. In *Evilgrade* we chose program *DAP* and configured backdoor to its update. Firstly, we needed to use a MITM attack to intercept a connection. Then we setup DNS spoofing to website *update.speedbit.com* to see all requests. After that, we used a *metasploit framework (msfconsole)* to listen for incoming connections using command: *use exploit/multi/handler*. We had to use payload: *set payload windows/meterpreter/reverse_http* after that we just configured *LHOST* with *LPORT* same as in the backdoor and then typed *exploit*.

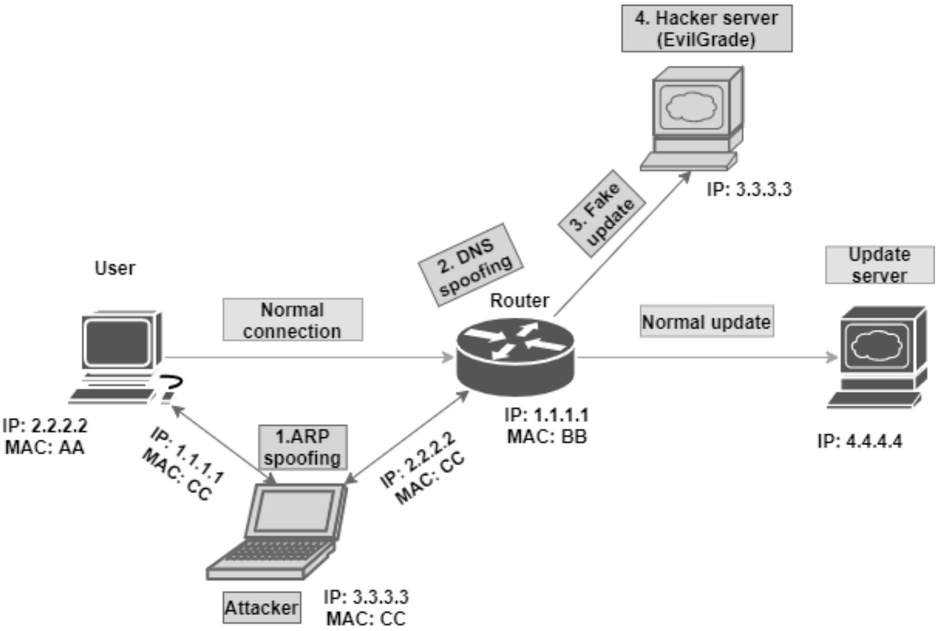


Figure 3 – The scheme of Backdoor as a fake update

We moved to Windows 10, the victim, and downloaded a program called *DAP*. Then we downloaded an update which was secretly a backdoor. In metasploitable framework we were connected to Windows 10 as displayed by the output of command *sysinfo* and we also launched a web camera using command *webcam_stream*.

Of course, we must point out that we switched off Windows Defender, because this backdoor is already present in its antivirus database. To prevent this attack, we should apply system updates frequently, use antivirus software from recommended providers, e.g. Eset, Malwarebytes, because they keep updating their database.

Integrity of file containing upgrade can be validated using hashes. Content of every file can produce a hash that serves as file's signature. This value should be unique and change when file changes. We used program *WinMD5* to check a file signature. If signature in WinMD5 matches the signature on website, files were not modified.

Another way to detect malicious files is sandbox (e.g. site hybrid-analysis) and it is a place where files will be executed and analyzed in sandbox environment. Program controls if there is any suspicious activity (e.g. opened ports) and based on it you get a report of that.

3.4 Security

Against every attack, there should exist a protection. Among typical security tools are antivirus software, a firewall or a VPN. However, we should use the most appropriate one to protect against different type of attacks. Deploying mentioned security solutions does not ensure data confidentiality, because attackers has always well thoughtful plan, how to bypass security.

A. Antivirus - Software that detects malware. It monitors data and behavior in real time and in the case, that malicious code is detected it blocks it and can remove infected files. Antivirus database is constantly being updated to enhance security. However, when malware signature is not present in the database, it is not marked as malicious by the software [14].

B. Internet security - expands antivirus possibilities. Its advantage is expanded to secure a communication on the Internet and during data transfer between a computer and another device, e.g. flash disk. It contains programs specifically focused on certain kind of unwanted software. It has a several functions like a Firewall, honeypot, Intrusion Detection System, security protocols, spam and so on.

C. Trap Honeypot - Honeypot is a security tool that every attacker is afraid of. Its task is to attract potential attackers and monitor their activities. It controls an activity of unauthenticated sources coming to a vulnerable and easily exploitable system. However, its task is to capture malwares and detect them. It included collected logs from the network. There are many kinds of malwares capturing but honeypot is probably the best idea [15].

D. https everywhere – a plugin for Google Chrome, Mozilla Firefox, Opera and Brave [16]. Automatically rewrites requests to sites to HTTPS protocol and does not allow attackers to downgrade https, to provide the same level of security as with HSTS. The disadvantage is that the plugin does not work with http-only websites and also attacker can still see your whole network traffic.

E. VPN - VPN creates secured and encrypted connections between networks or host and a network. It is used to hide an IP address of the user and can encrypt data. The attacker will not be able to see our activities or passwords despite usage of http, because the whole traffic will be encrypted. A recurring factor is to choose appropriate VPN, because Internet Service Provider will be able to see all our data. In combination with VPN we can use *https everywhere* plugin to enhance security. The communication between server and computer will be secured using two layers encryption consisting of VPN and TLS (Transport Layer Security) [17] used in *https everywhere*.

F. XArp, Wireshark ARP - *XArp* is a program which draws attention to a suspicious behavior based on IP and MAC addresses. In case an attacker chooses and ARP spoofing or another form of MITM attack, we are immediately notified about the MAC address change at the given IP address.

Wireshark, a packet analyzer, has many use cases. One of them is Wireshark ARP with which we can find suspicious activity, when someone wants to hack or monitor our device. When the attacker performs an attack e.g. MITM, we get warning in analysis: the configured duplicated IP address. We achieve this by setting a packet filter to *ARP/RARP*.

4 Conclusion

We performed all of the attacks for reason to act same as hackers. We found and applied proper protection methods that will keep our devices more secure. We recommend applying this protection on user's devices to help enhance their security. These attacks are easy to feasible and we wanted to exalt the tools which can help people to enhance their protection and test their system. However, described attacks can only be part of some possible scenario. Our main goal was to summarize these attacks to help understand how network or computer can be compromised. We demonstrated that we do not need any special hardware for Kali Linux.

In future, we plan to enhance a security of IoT devices, because the security should not only be considered in network infrastructure, but also with embedded devices connected to the network. They are often unable to run antivirus software or any another security tools.

Acknowledgement: This research was supported by the Ministry of Education, Science, Research and Sport of the Slovak Republic, Incentives for Research and Development, Grant No.: 2018/14427:1-26C0. This work was also partly sponsored by the Eset Research Centre. We also want to thank you Zaid Sabih from zSecurity for provided materials and for teaching penetration testing techniques.

References

1. M. Yevdokymenko, E. Mohamed and P. Onwuakpa, "Ethical hacking and penetration testing using raspberry PI," *2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, Kharkov, 2017, pp. 179-181. S. Patil, A. Jangra, M. Bhale, A. Raina and P. Kulkarni, "Ethical hacking: The need for cyber security," 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), Chennai, 2017, pp. 1602-1606.
2. H. M. Z. A. Shebli and B. D. Beheshti, "A study on penetration testing process and tools," 2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT), Farmingdale, NY, 2018, pp. 1-7
3. K. Patel, "A Survey on Vulnerability Assessment & Penetration Testing for Secure Communication," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2019, pp. 320-325
4. Kali By Offensive Security "What is Kali Linux," [Online] Available: https://www.kali.org/docs/introduction/what-is-kali-linux/
5. D. T. Vojnak, B. S. Đorđević, V. V. Timčenko and S. M. Štrbac, "Performance Comparison of the type-2 hypervisor VirtualBox and VMWare Workstation," 2019 27th Telecommunications Forum (TELFOR), Belgrade, Serbia, 2019, pp. 1-4.
6. K. Yoshioka, Y. Hosobuchi, T. Orii and T. Matsumoto, "Vulnerability in Public Malware Sandbox Analysis Systems," 2010 10th IEEE/IPSJ International Symposium on Applications and the Internet*, Seoul, 2010, pp. 265-268.*

7. K. W. Professor, W. Z. Student and W. Z. Teacher, "A study on the application of intrusion detection technology to WLAN," 2011 IEEE 3rd International Conference on Communication Software and Networks, Xi'an, 2011, pp. 344-346.
8. Y. Jiang, K. F. Zheng, Y. X. Yang, S. S. Luo and J. P. Zhao, "Evaluation Model for DoS Attack Effect in Softswitch Network," 2010 International Conference on Communications and Intelligence Information Security, Nanning, 2010, pp. 88-91.
9. C. Chen and T. Chang, "The Cryptanalysis of WPA & WPA2 in the Rule-Based Brute Force Attack, an Advanced and Efficient Method," 2015 10th Asia Joint Conference on Information Security, Kaohsiung, 2015, pp. 37-41.
10. M. Data, "The Defense Against ARP Spoofing Attack Using Semi-Static ARP Cache Table," 2018 International Conference on Sustainable Information Engineering and Technology (SIET), Malang, Indonesia, 2018, pp. 206-210.
11. M. Janbeglou, M. Zamani and S. Ibrahim, "Redirecting outgoing DNS requests toward a fake DNS server in a LAN," 2010 IEEE International Conference on Software Engineering and Service Sciences, Beijing, 2010, pp. 29-32.
12. H. Gupta and R. Kumar, "Protection against penetration attacks using Metasploit," 2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), Noida, 2015, pp. 1-4.
13. O. Nasser, S. AlThuhli, M. Mohammed, R. AlMamari and F. Hajamohideen, "An investigation of backdoors implication to avoid regional security impediment," 2015 Global Conference on Communication Technologies (GCCT), Thuckalay, 2015, pp. 409-412.
14. F. Hsu, M. Wu, C. Tso, C. Hsu and C. Chen, "Antivirus Software Shield Against Antivirus Terminators," in IEEE Transactions on Information Forensics and Security, vol. 7, no. 5, pp. 1439-1447, Oct. 2012.
15. P. D. Ali and T. G. Kumar, "Malware capturing and detection in dionaea honeypot," 2017 Innovations in Power and Advanced Computing Technologies (i-PACT), Vellore, 2017, pp. 1-5.
16. I. Dolnák and J. Litvik, "Introduction to HTTP security headers and implementation of HTTP strict transport security (HSTS) header for HTTPS enforcing," 2017 15th International Conference on Emerging eLearning Technologies and Applications (ICETA), Stary Smokovec, 2017, pp. 1-4.
17. W. B. Diab, S. Tohme and C. Bassil, "VPN Analysis and New Perspective for Securing Voice over VPN Networks," Fourth International Conference on Networking and Services (icns 2008), Gosier, 2008, pp. 73-78.

Interactive presentation of fake news detection

Jakub Perdek*$^{[0000-1111-2222-3333]}$

Slovak University of Technology, Bratislava, Slovakia 841 04
perdek.jakub@gmail.com

Abstrakt Basic principle of this work is to construct web solution where it is possible to test s specific message according to machine learning and deep learning algorithms using a standard web browser. This testing should be fast and must use at least one defined language to detect potential fake news. Messages sent for detection should have multiple sentences or be up to one page long. Description and evaluation with complexity of some methods should be part of it. Aim is to construct, describe and deploy some methods for detection of fake news mainly in social networks. Another aim is visualization and overall complexity of fake news evaluation related to its amount. Partial purpose is to compare some frameworks from deep learning related to results obtained using certain datasets on them. Main part is response of visitors on appearance and functionality of web solution. Web solution has been created and with some methods deployed so far. These methods can be tested. During realization nosql database Redis, cleaning of data and emphasis on similar conditions which were deployed while learning of certain model was used. Compared frameworks, Tensorflow and Theano resulted in approximately same accuracy results during testing. But during training comparison of loss was better for Theano which exceeds Tensorflow of more than ten percent. Asset of this work is analysis of concrete methods with an option of their use in one place.

Keywords: fake news detection · deep learning · Naive Bayes

1 Introduction

Interactive presentation of fake news detection should help people perceive methods and their goal as an opportunity to stop some affairs. In this work we focus on analysis of fake news, especially on social media. The solutions should acquire behavioral characteristics and reasons to prepare basis to their detection. Many of these attributes cannot be compared because of boundaries they have. The main idea is to describe them on one website (see 1). This will be an interactive presentation of algorithms and their use. People can use ones prepared for them. Another is analysis of complexity compared to the amount of data

* Bachelor study programme in field: InformaticsSupervisor: Igor Stupavsky, Institute of Informatics, Information Systems and Software Engineering, Faculty of Informatics and Information Technologies STU in Bratislava

that can be utilized. Deep learning and bigrams are analyzed in first. We tried to get some results from their normal use and then improve them as much as it is possible. Comparing some utilities can show as different same techniques are. Nevertheless their resulting with small deviations as same. Used information is varying from old work of graduate students from 1989 (see [5]), to work which returns back only ten years. We try to obtain some reasons, their background to get some concomitant attributes. These attributes should be observed in other situations with specific similarities. Their presence in propaganda, social media, charlatanry (see [5])and other areas of everyday life shows us their relationship to information sharing and social life of people. Influence of them is often misused for political cases and other changes of our lives. They give us much more information that begins with an unknown source of health, extends to tabloid information, behavior of stereotypes, influence of majorities without respect of another truth causing no discussion on some topics. All of these forms overload our minds. Reading is comfortable, but it does not emphasize our critical thinking and knowledge.

2 Used algorithms

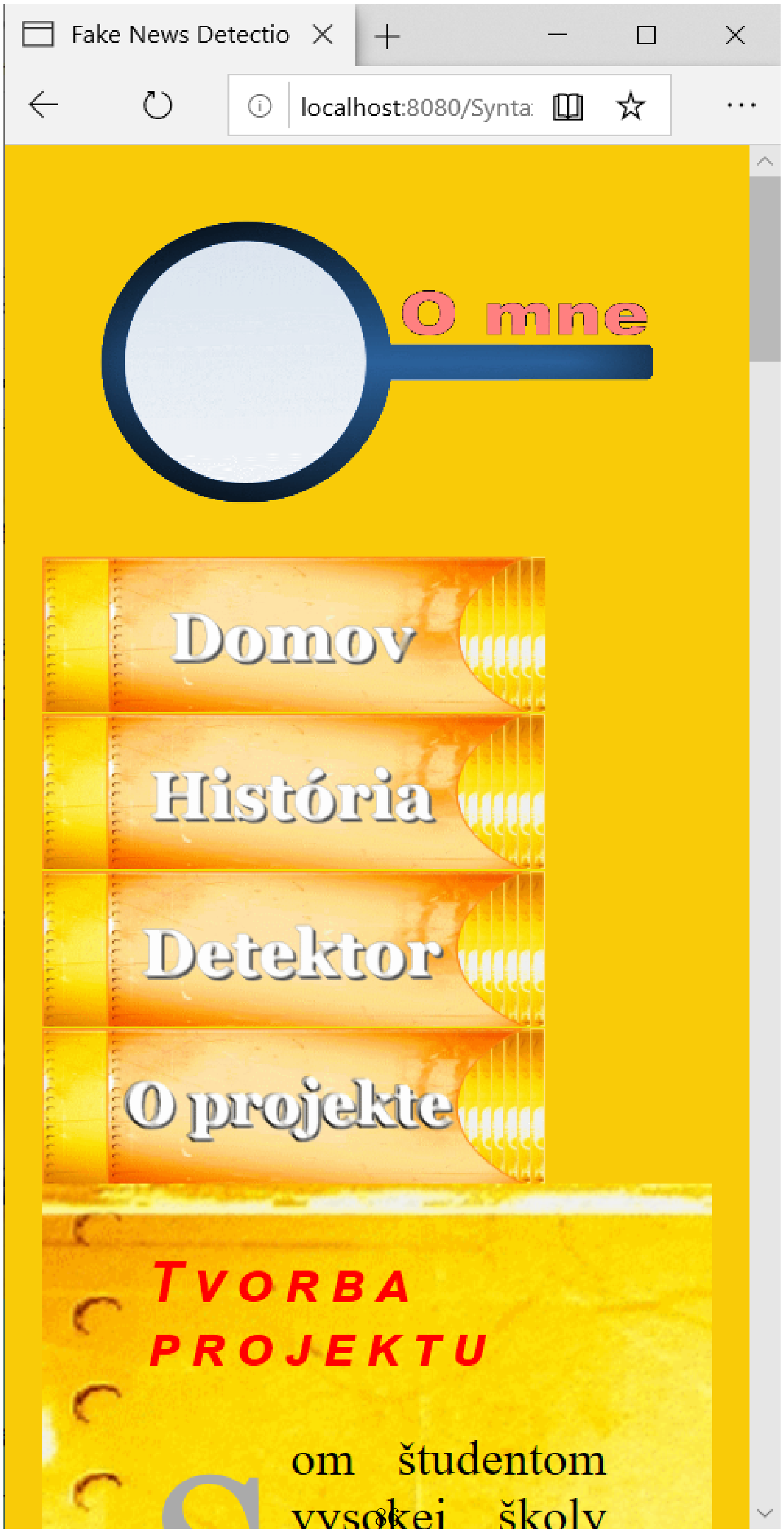
Fake news represents potential threat and we need to think about ways how to reduce it. Is spreads faster than normal information and usually appear as conscious activity. Plenty of people are working on some detection programs in an attempt to capture fake news and write about fake news makers. These names appear on list that can reduce this threat. Some visualization is also needed to help people think about information and its detection. This work tries to think about some forms of visualization of created solution, which can be used to detect fake news. The integration of these services is important topic of this work, because detection should be fast and accurate. We use two programming languages to construct this solution. One for object orientated representation of some algorithms and for its portability. Second for many libraries suitable for machine learning and frameworks for deep learning that have been developed so far. Problems are coused not only by languages, but also by versions of some libraries or other dependencies that require special treatment.

Firstly, we implemented solution using deep learning and some basic layers, for example dense or some convolutions. I focused on textual data using Keras[1] as framework. During my work I tried both Keras backends, Theano[2] and Tensorflow[3] to run the same code on them. One necessary change of Keras.json file should be needed. Finally, We tried to compare the results where Theano is slightly better.

[1] https://keras.io/

[2] http://deeplearning.net/software/theano/

[3] https://www.tensorflow.org/



om študentom
vysokej školy

Figure 1. Menu and responsive design of web page.

Not all detection algorithms are part of deep learning. Deep learning is only small part of machine learning. I have created some implementations based on statistics as Naive Bayes or bigrams in Java language so far. Another focus was implementations using Python libraries, for example supported vector machine algorithm and previously named ones. One of them is mlpy, nltk and known skilearn.

Many algorithms have been created for detection in our project so far. It differs from those using statistics from those that rely on gradient descent. All of them could be used in many ways. Abstracting some of the features from them can help create better and maybe faster solution to some problems. Known example is combination of convolution network to construct it faster and recursive network which helps determine flow. Recursive networks can be slow. Maybe using combination of bigrams in Naive Bayes can improve results too.

Another part is creation of some algorithms for better analysis and to find some aspects which can be visualized to help person think about reason of potential fake parts resulting in obtained final category and percentage.

Description of some known algorithms of machine learning should be provided here aiming on implementation parts.

2.1 Naive Bayes algorithm

Naive Bayes algorithm is used for text classification. Usually for spam detection. It uses conditional probability of categories and some words. Analysis of fake news can be compared to spam detection and treated as classification problem. We can finally obtain if it is fake or not.It depends on implementation whether additional categories should be added, for example, to identify some kind of fake news there.

I implemented following programs for classification using Naive Bayes:

1. At first, program was initially inspired by description given in Data Analysis from Hector Cuesta. [3] A good aspect is that design allows an extension to classify many categories, not only two. But is very simple at all. It is based on statistics and counts probabilities for each category, which is good indication whether the visualization can request this information to inform about other categories and their influence on the final result. Design is focused on words and their occurrences for each category. Additional attributes that word represents should be added and compared to the original result. This inspired me to rewrite program. New part is coded in Java to be object orientated and easily improved and portable based on own design of algorithm including all named parts. Later, using the same inputs and obtaining the same outputs from these implementation, focused on the core of program, helped create unit tests and test accuracy. Because it is simple algorithm, resulting percentage was very small, below a few per cent. It can be comparable with results using only n-grams approach.

2. Secondly, Naive Bayes constructed using nltk library. Consists of creation field of words that can be represented by n-grams, but for simplicity, unigrams or normal words are used. These fields with method which creates set from an array should be put to nltk.classify.`apply_features` method. Then training of Bayes classifier can begin using returned object from previous step. The last step before testing classification of text is an overview of given accuracy from classifier. It can be tested using accuracy method of classifier and the classifier object and test data.
3. Finally, the last approach use sklearn library. This library be used for Supported vector machine algorithm too. Steps include vectorization of data and their use in transformation of train and test text data. Categories are encoded for text and test data too. Object from sklearn library `naive_bayes` is then applied. Data are prepared and training can begin using training data and represented categories after last step. The same as in previous algorithm accuracy method of this specific `naive_bayes` object is used on test data and encoded categories to obtain accuracy. Result shows that accuracy reaches 85% on gossipcop dataset with 19933 records. Basic approach which has been used to create program can be found here [6]. We used multinomial naive bayes with counter vectorizer and alpha parameter with value 0.9 on liar dataset[4] too. After preprocessing of data final result reaches 25%.

2.2 N-grams

N-grams are machine learning method that uses n-long word or character sequence from sentence. They are like jettons, which points to lost of information about text structure that has consequences for its use. However, they are always a powerful tool for our review [2]. Similarly as Naive Bayes simple approach, n-grams were implemented in Java too. This combination should increase statistical information about trained data and can be similarly visualized as mentioned above. Some techniques of pre-processing data as stemming or stop words removal (see [1]) are applied to text before training and classification.

2.3 Dense networks

Fully connected networks have all neurons connected to the next layer. They are able to train quickly, but their behaviour is hard to visualize. They learn global patterns of input signs. As example it can be all pixels of picture. The designed solution uses sigmoid function after several dense layers to classify text as fake or real. Some drop layers may also be combined. The probability gained is 81 % in my experiments on gossipcop dataset with 19933 records or up to 90.65 % in misinfofinder dataset which contains only 250 records.

[4] `https://github.com/thiagorainmaker77/liar_dataset`

2.4 Convolution networks

Convolution networks can learn local patterns that are the main difference between dense networks. Pattern can be found anywhere in the text. They can be visualized using filters applied to some convolution layers. Probability for text tested was slightly better, but their training is slower.

2.5 Multiple input networks

If is opportunity to use more classification techniques because of extended information then multiple input network can be used. For example Liar dataset apart from statement have information about subject, speaker, speaker's job, content, party affiliation and state info. We build multiple input network inspired by Wang approach[7] consists of trained embedding layer, but not on words from Google news, but on one hundred dimensional Wikipedia words from Glove project[5] and three convolution layers with two max pooling layers between them. Other information have own input layer and some convolution layers. According to Wang each ended by bidirectional LSTM layer. All of these inputs are concatenated. The last layer is dense layer with 6 outputs and softmax function representing 6 classes from which only one should be chosen. We do previous steps for speakers job and statement and obtained test accuracy up to 21.5%. Without additional information test result will be only 20.5%. Wangs results in this case reaches 25.8%.

3 Visualization of given algorithms

Typical input and output for any fake news detector should consist of category name and the percentage obtained for given category. These two values can be displayed to desired person. A certain algorithm is based on different conditions that the text should met. Because of their differences, different visualization techniques should be used. For classifiers which classify text will be useful know some facts about it. For example, number of certain unigrams, number of these words for requested category or maybe sequence of these words anywhere in the text.

Naive Bayes algorithm needs number of all unique words, the number of unique words for the given category, the number of occurrences of the given word and the number of all words associated with the given category. These values can be visualized and packed with the result category name to demonstrate reason for such choice if the probability increases by specific value. Chances of all categories can also be compared and displayed in a graph.

N-grams use the probability of occurrence of words. Laplace smoothing can be used to determine number untrained words to ensure that in most cases final result should not be 0. The number of times a word is obtained from sentence in training set that the probability for category which represents training set will

[5] https://nlp.stanford.edu/projects/glove/

be higher. The number of words that should be ascertained is again very useful to display (see 3).

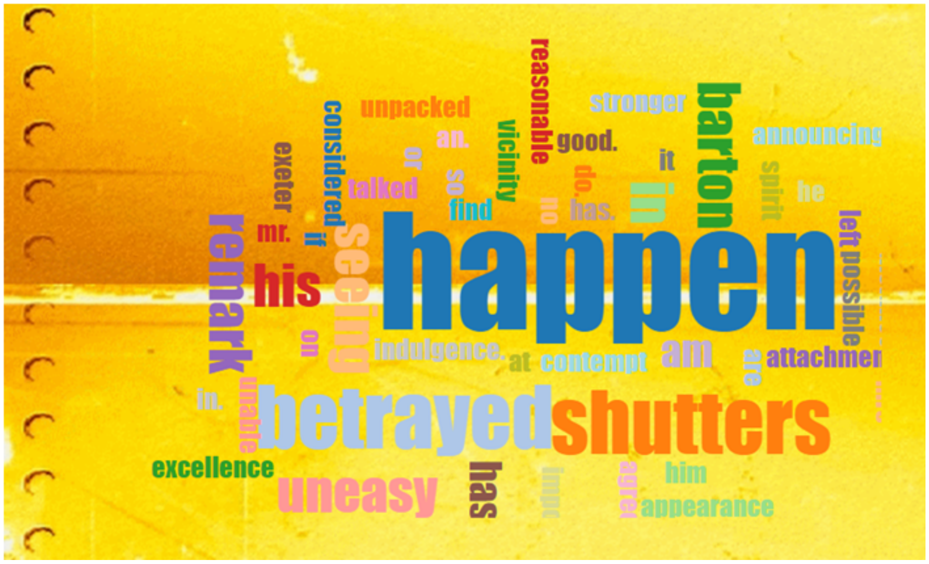


Figure 2. Word visualization after detection displayed on web page.

Displaying word count can be provided by graphs, but better is to use font size. Font size can show which words actually influence the end result.

Some methods cannot be easily visualized. Dense layers are difficult to visualize, but are often used to find sequences (maybe of words) on any place in the text. The convolution network can be visualized by using filters on specific layers. Through these filters some features can be displayed how as the convolution network sees the world and makes certain choice.[2]

It is possible to get certain words which were used in prediction. Keras function models.Model consisting of given model input and array of outputs from specified layers returns model which can be used for prediction. This prediction is called by predict function on this object. Then it is possible to iterate through all layers of this model to get activations. Using argmax function of numpy as encoder is possible obtain numbers. From them is possible to get previous tokenized words back. The same principle can be used on convolution layers. Results can be converted back after training of random sample on certain layers too. The same principle is used by Chollet for visualization of pictures to get patterns which are learned by certain filter. Then all patterns from each filter are presented together in one picture. [2]

Visualization of words can be fulfilled by creating json file with obtained words. One way is to organize file for graph representation. This type of file contains nodes and edges as arrays. Nodes should be one of two types. The first type represents layer and the second one word. Other information as count or unique identifier can be included. Edge contains source and target identifiers. For certain case information about possible connection with the same word from another layer can be included. All can be visualized by javascript, especially by D3. Interactive graph from json file can be generated. Styles are possible to used on many elements. Nodes which represents layers can make larger and edges to them visualize as green elements. Connections between words as visible pink objects. During our interaction font size of texts near out hover should be bigger an visible. After double clicking on certain node, contribution of chosen word on stop word list or another list will be done. This improves an interaction and can be usable for may cases. Improvement of visualization by displaying number of the same words is easy to reach. The graph view is displayed in 3 and distant whole view in 6.

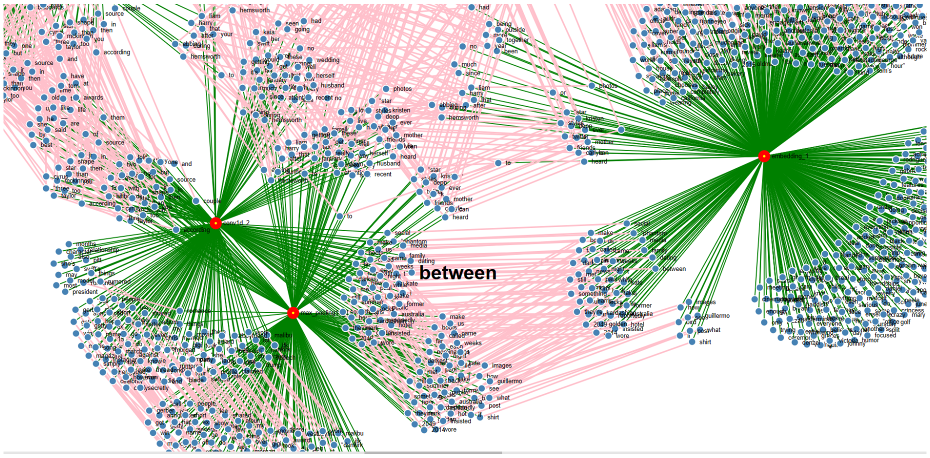


Figure 3. Graph structure of used words during prediction on 10 records.

Often there is connection only to certain layer where certain word has been used. This design is based on words used by layers, but should be improved to visualize words on filters too. But this type of graph will be enormous in size if more layers are combined as main point of visualization. This design can be reached by inserting all information to json file. It includes every word for every filter and for every layer. File can be large, for example 7MB with use of only 6 layers, each with 64 filters on 10 records.

4 Architecture and benefits of web services

Service orientated architecture is the best way to design this application. Many services can be reused from different application coded in various languages. Many good principles should be used to exchange data or calling some sort of web service and receive requested data, such as web services based on SOAP or REST. These services should also be used by others to provide larger services. Its platform independent. [4]

Services based on wsdl files can regenerate some classes in many languages with a certain plugin to get services fast. Its up to application how to use and display these data to end customers.

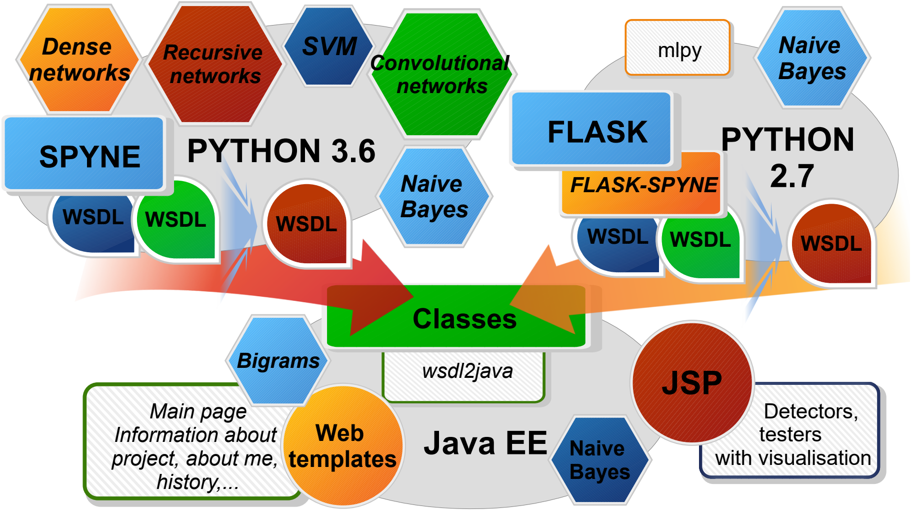


Figure 4. Components and communication.

4.1 Problems before improvement of services

During development, good design, including architecture, should be chosen. A system that needs a design change can depict as follows:

1. Lot of Python versions must be maintained on the same machine. Mlpy usually needs lower versions of Python. For Theano specific CUDA version is necessary because newer ones are rejected or Tensorflow required Python 3.6 when version 3.7 was released for GPU support.
2. Specific system directories and files are required for launching some applications. After change of directory all paths must be changed. Files for Windows should be .bat or .exe. For example, conda environment requires conda.bat file to activate the environment.

3. Various programs need fulfill their package dependencies which arise with requirements that allow use of GPU and older libraries namely mlpy, or flask-spyne.
4. Base application written in Java which provides services must call some procedures frequently for every request. Due to another language and complex programs, including deep learning frameworks, it is difficult to simulate code using Jython. These calls must be fulfilled by command line interpreter as the easiest solution. Some other implementations written in Java as Naive Bayes or Bigrams are provided here too.
5. File-based communication is widely spread through application programs for visualization.

More problems later arise and it comes necessary to search ways. The critical problem and reality of this system is that it is very slow. For users who try to test few sentences, it may be unsatisfactory to wait tens of minutes for a simple request. What about time to wait for complex ones on backend? Having observed some tips in the PHP book that criticize the command line approach of calling another programs, there were more and more disadvantages:

1. Activation of Conda environment must be ready for every request.
2. The packages used after loading the scripts must be loaded for every request. If large frameworks like Theano or Tensorflow are used it takes few minutes to launch such script. A large number of packages not make application faster.
3. Starting of command line interpreter requires start of more than one process for each request.
4. Using files for transfer which can be used by javascript needs lot of read/write operations. This includes creating a file and its name should be associated with unique value because the content of requests differ. Often they cannot be removed after its use, for example for visualization. Accessing and downloading files from multiple directories does not help in this situation too.

4.2 Improvement of system

After the implementation of web services (based on SOAP) many of problems described above mainly those caused by use of command line interpreter have disappeared. Enhanced parts can be enumerated as follows:

1. Activation of conda environment must be ready for every request.
2. Use of server guaranteed that application with all libraries is prepared and these files should not be repeatedly loaded. They are loaded only once when server starts. This includes some serialized files which can be used in application and should not be loaded for every request. It is suitable for all preparations.

3. No need for command line interpreter and possible activations of Conda environment for every request. All is managed by server which can run this activation less than twice.
4. All data should be transmitted via a single platform independent message between applications. If visualization is required, data can be converted to json format and can easily be used to display key information using javascript.

Orchestration is based on central application. It needs only wsdl files that end point Python servers offer. After change Java application can regenerate classes from this source code, if some changes of this files appear, which can be solved using protocols to inform eligible person. Final design is depict here [4].

Reality shows that using soap messages to communication is less time expensive than command line opportunity because of reasons mentioned above. It is more flexible and change of base application can be easily satisfied if necessary.

5 EXPERIMENTAL EVALUATION

Detectors were manually and automatically tested. Manual testing consists of writing short message into defined area and then clicking on button, which is placed below the area. While users are waiting few minutes, animation of clocks is displayed. After all processing is done they can see probability of resulting category, category name and one of prepared visualization because of difference of specific algorithm. This test was performed for two implemented methods, one for deep learning and second for Naive Bayes classifier.

Comparison of command line approach and web service orientated one require tool for endurance testing. JMeter can be used for this type of testing, especially for web services. Server, port number, method and path of application form in jpa page were set. Text for detection was added as parameter represented by text area identifier. Experiment consists of sending request to this form simultaneously and see duration and other aspects. Then server should be restarted with different value disabling web services approach and enabling command line one. These results can compared.

Four different tests were launched for every approach. Firstly, five requests were simultaneously send for fifty times. Secondly ten requests for twenty-five times. Thirdly twenty-five requests only ten times and for last fifty requests five times. Result shows that command line approach is up to twice slower in program that use only two serialized objects and nltk library and created two small files. Large frameworks should be more time consuming. All test ending times varying less than eighteen seconds. None request ends in error. Results are shown in table using n x m notation, where n means how many requests run in parallel and m how many cycles they were repeated (see 1 and 2). If visualization needs more data, then approach with REST web services using JSON format should fasten solution because of data preparation which needs converting sent data to this format during creation of jsp template on server and later used by javascript. If data is sent in required format, conversion is not needed.

Table 1. Data from endurance test for implementation using command line

parameters	**5 x 50**	**10 x 25**	**25 x 10**	**50 x 5**
Total time [minutes:seconds]	11:11	10:31	8:30	8:13
Average elapsed time [s]	88.0736	46.8667	24.4731	13.3138
Average connect time [ms]	3.162	1.128	1.58	1.28
Average latency [s]	88.0735	46.8667	24.4731	13.3138
Average iddle time [ms]	0.0	0.0	0.0	0.0

Table 2. Data from endurance test for implementation using web services

parameters	**5 x 50**	**10 x 25**	**25 x 10**	**50 x 5**
Total time [minutes:seconds]	4:18	4:21	4:19	4:25
Average elapsed time [s]	52.8989	25.8488	10.4397	5.1341
Average connect time [ms]	1.212	1.252	1.352	1.388
Average latency [s]	52.8988	25.8488	10.4397	5.1340
Average iddle time [ms]	0.0	0.0	0.0	0.0

6 Discussion and conclusions

Many algorithms were invented to detect fake information. Through deep learning local and global patterns can be found. Machine learning techniques as Naive Bayes uses statistical information to construct final probability. Substantial aspect of these techniques should be visualized and shown with result. Many of these techniques can be combined as using n-grams in Naive Bayes and maybe extend computation using emotional lexicon. Combination of recursive network with convolution network is very typical. Each of these algorithms have various implementations, but only few of them predict with high probability shown as deep learning ones.

Results obtained on dataset from politics and fashion reaches up to 85 per cent. This dataset can be found on the Kaggle[6]. Naive Bayes and SVM implemented by skilearn have surprisingly better accuracy than convolution or dense networks from deep learning.

Evidence must have its visualization and needs fast way how to prepare and bring data to the end user. Web services are up to twice faster than command line according to endurance tests on localhost and reasons mentioned above. Tests consist of few word messages which were send to Python application which predicts probability, then to web application to construct java servlet page and some information (for example words) as json object to javascript where visualization could be processed and displayed.

We created stylized graph structure to show which words are used on given layer. It interacts with user by displaying larger font size when hitting by hover and can be used as stop word list generator. Easily expansion for another infor-

[6] https://www.kaggle.com/

mation as word count and other details is opened by use of css and javascript on svg picture. Problem of high complexity remains (see 6).

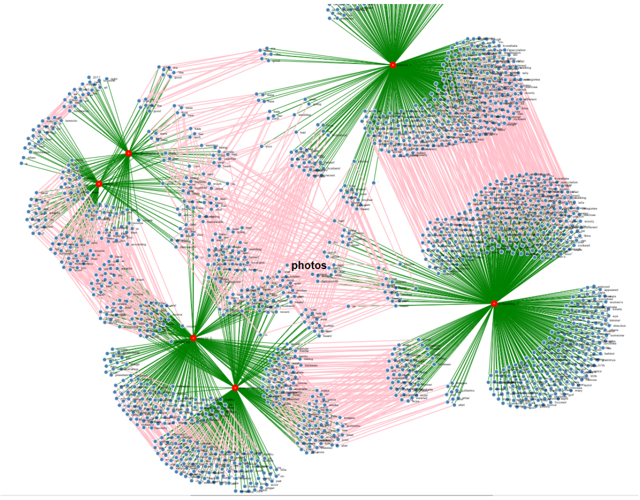


Figure 5. Distant view of graph consisting of 6 layers, each up to 100 filters obtained from 10 records.

ACKNOWLEDGMENTS

I would like to express thanks for Rebelion project, APVV-17-0267, aimed on automatic recognition of antisocial behaviour in online communities. Thanks to for all of their advices and help.

Reference

1. Ahmed, H., Traore, I., Saad, S.: Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. Springer International Publishing, 127–138 (2017)
2. Chollet, F.: Deep learning v jazyku Python. 1st edn. Grada, Praha (2019)
3. Cuesta, H.: Analýza dat v praxi. 1st edn. Computer Press, Brno (2015)
4. Earl, T.: SOA Servisně orientovaná architektura. 1st edn. Computer Press, Brno (2009)
5. Kettner, P.: Léky, léčitelství a šarlatáni. 1st edn. Horizont, Praha (1988)
6. SVM and Naive Bayes tutorial program, `https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34`. Last accessed 10 Jan 2020
7. Wang, W.: ”Liar, Liar Pants on Fire”: A New Benchmark Dataset for Fake News Detection, Association for Computational Linguistics, 422–426, Vancouver, Canada (2017)

Measuring data flows and controlling them with QFLA in Software Defined Networks

Viktor Lančarič*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
xlancaric@stuba.sk

Abstract. With the growing popularity of multimedia services, it is increasingly difficult to ensure their quality in traditional networks. The software-defined network (SDN) model provides a way to programmatically configure switches during operation and manage network resources in a more efficient way. Thanks to the properties of software-defined networks where the hardware part is separated from the software, we can access advanced data stream control in multipath networks. This paper deals with possibilities of using the OpenFlow communication protocol, which allows network drivers to determine the path of network packets through network switches. Such flow control can be described by a mathematical problem known as a knapsack problem, that solves the packing of a subset of items (data flows) into knapsacks (links) with limited capacity so that the sum of the prices of the inserted items is the maximum possible. QFLA algorithm was designed to effectively solve such a problem and can be used to create an SDN control unit capable of effectively managing data flow and ensuring the quality of service in multipath networks. This paper deals further with the implementation and testing of a solution over the network topology created in the SDN network simulator.

Keywords: SDN, QFLA, RYU, OVS, Knapsack problem, network monitoring

1 Introduction

In recent years, video telecommunication software like Cisco telepresence and live streaming services got popular between users. Users are expecting seamless experience using these services. Such services are working on top of the IP protocol, which does not ensure any Quality of Service.

Traditional networks have a static architecture that is decentralized, complex and hard to manage. To solve these limitations the model of Software-defined Networking has been created. It provides a way to programmatically configure switches during

* Master study programme in field: Internet Technologies Supervisor: Ing. Marek Galinski, Institute of Computer Engineering and Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

operation and manage network resources in a more efficient way. The software part must provide efficient data flow management capabilities. What opens space for creating a solution that would be able to control data stream routing in multipath networks. SDN control plane allows us to create an SDN application – network controller, that will allow to control data stream routing, to identify and monitor the state of active connections from one place. Common communication protocol associated with Software-defined Networking is OpenFlow, which will allow us to execute control commands in our topology. In a result, the application will be able to distribute data streams between available network paths to achieve maximal utilization of each path based on its bandwidth. The problem of data stream distribution is similar to the well-known Knapsack problem, in which we need to solve how to pack a subset of items (data streams) into knapsacks (links) with limited capacity so that the sum of the prices of the inserted items is the maximum possible.

This paper is organized as follows. Section 2 describes Knapsack problem and QFLA algorithm. Section 3 describes approach for getting data flow information from OVS switches. In section 4 we summarize our achieved results.

2 State of the art

Quick forward lookup algorithm is an optimization algorithm (QFLA) for solving value independent multiple knapsack problem. QFLA allows an exchange of items between multiple knapsacks and works with the version of the gradual addition of items, and thus the items cannot be arranged in advance.

2.1 QFLA

The algorithm works on the principle of shifting one or more items between knapsacks. The algorithm itself consists of two phases. In the first phase, the algorithm tries to reassign one item from a particular knapsack, so that after moving the item, it can accommodate the new request and store the new item in the knapsack. The second phase consists of a similar principle of shifting items and enters it until the first phase fails, so it cannot move one item so that the new item can be placed in the backpack. At this stage, the principle of relocating multiple items to other knapsacks is applied in order to release sufficient capacity to accommodate the current item placement requirement in that knapsack [1]. QFLA has been already improved**.** Improvements have increased its performance and reduced complexity. The improvement resides in arranging knapsacks and items. This enhancement ultimately reduces the average time needed to solve the problem. The complexity of embedding in the worst case is in the complexity of O (n). Therefore, in the worst-case scenarios for each item m, the algorithm will have to insert n items into the linked list and thus pass through. So, in the worst-case scenario, the complexity would be around 0 (n * m) [2]. Another improvement to QFLA is using AVL binary tree for storing knapsacks sorted by their remaining capacity. The complexity of adding and removing is in AVL binary trees in the worst case of O (log n), which is more efficient for a bigger number of knapsacks, than using linked lists [3].

2.2 Knapsack Problem

The Knapsack problem is one of the most studied problems of dynamic programming. There exist several variations of the problem and different algorithms to solve these problems. Since QFLA is trying to solve the Knapsack problem, we should describe the problem a little. The problem has the complexity of non-deterministic polynomial time (NP-complete). No efficient solution algorithms are known for this type of problem, i.e. algorithms whose computing time only expands as a polynomial of instances of examples. [4]

Our problem can be compared to the Knapsack problem, where the knapsacks are represented by network links and the item, we want to put into one of the available knapsacks is network communication between two clients. The problem arises whenever there is a change in network traffic leading to the creation of a new session or client addition. [1] Complete enumeration is an exact solution technique but requires a lot of time to find the best solution. Following techniques are trying to improve the solution:

- **Branch and bound** - complete enumeration, with the exception of nodes that cannot contribute to an enhanced solution
- **Dynamic programming** – basically enumeration improved by the addition of rules of dominance. This technique can be used to obtain interesting time limits for several problems in the Knapsack family.
- **Preprocessing** - several factors may be in advance set at their ideal values in order to exclude certain values of the factors of the solution [5]

Our problem is similar to Knapsack problem but rather a subset problem since the Knapsack problem expects items of different size and value, that when put together into a knapsack, exceeds its capacity. The Knapsack problem does not count on that all items will be put into the knapsack. In our case, all items have the same value and the capacity of the link is final and cannot be exceeded, because it would degrade the user experience [1].

3 Implementation

The environment we've created for testing our solution designs is as follows. For network simulation we've used GNS3 and GNS3 server. GNS3 server is running in VMware Fusion 11.5.1. VMware is fast and supports nested virtualization (the VMs inside the VM are accelerated by your CPU). This implementation allowed us to create an SDN network as will be mentioned later. We also use Docker containers for OVS switch and Ubuntu system, where is Ryu controller running. Ryu framework is written in python, so we had to install python version 3 in the container. An important part was to enable the persistence of docker containers in GNS3, so the installed software is present after reloading the GNS3 application. For generating and measuring network traffic, we used docker container ipterm.

3.1 Monitoring part

This part of the application is responsible for collecting statics of network flows and calculation of their bandwidth. From the received EventOFPFlowStatsReply reply we get information about flow byte_count, duration_sec and duration_nsec which can be used to calculate bandwidth of the flow with the following equation:

$$\text{bandwidth} = (\text{byte\_count2} - \text{byte\_count1}) / (t2 - t1) \quad (1)$$

Our goal is to create an SDN application that will communicate its controlling logic through the northbound interface of the SDN controller. To test the controller, we designed SDN network topology as well. Figure 1 shows simple SDN topology for the purpose of testing the SDN application, which monitors flow statistics. The topology was designed to include Open Virtual Switch (OVS), which is controlled by the Ryu controller. Next, there are two ipterm docker containers responsible for the generation of network flow, which are monitored and evaluated by the Ryu controller. This topology serves as testing topology for monitoring network flows.

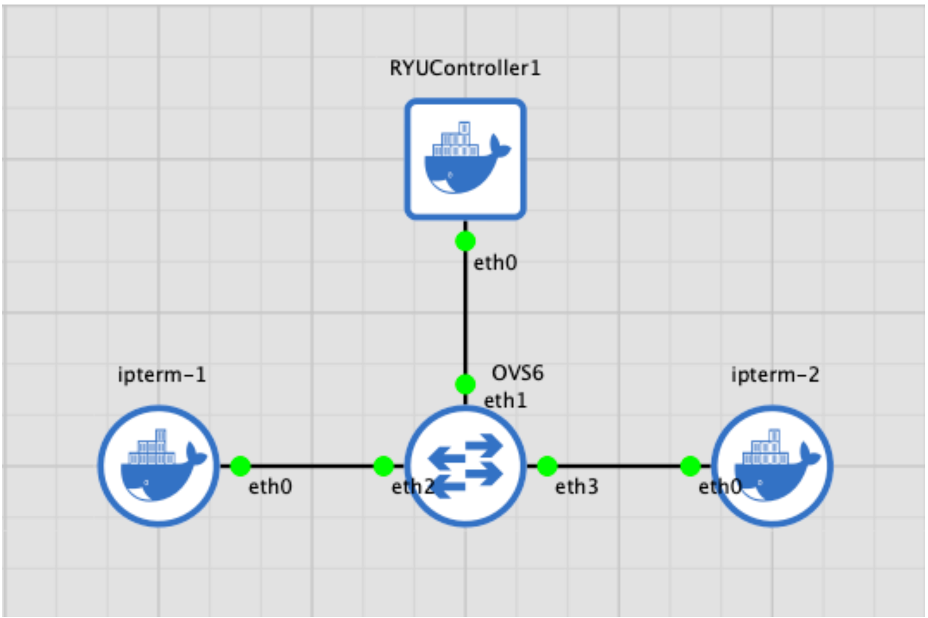


Fig. 1. Simple SDN topology

Figure 2 includes multiple network paths, multiple OVS switches, and one Ryu controller. On this topology, we test the final implementation of the SDN application responsible for controlling data stream routing in the network.

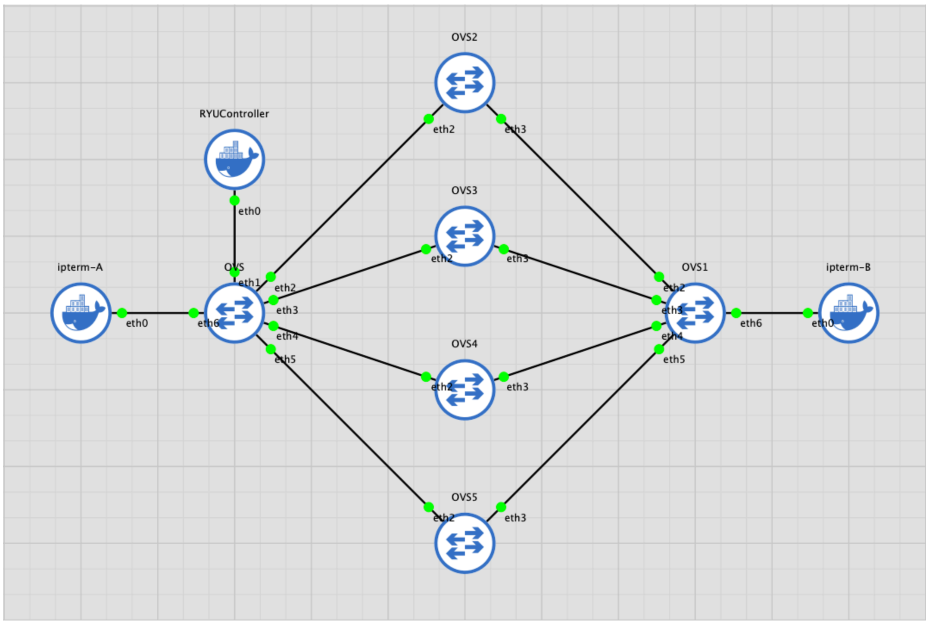


Fig. 2. Multipath SDN topology

Ryu application is the most important element of this work. Application for effective operation needs accurate and timely information about network status. Based on this information application will find the optimal network path for the newly added flow with the help of the QFLA algorithm.

Ryu framework allows to simply add monitoring function to our application. Monitoring function runs in parallel with our application, as a thread that periodically requests OpenFlow switch for statistical information. The monitoring function infinitely requests statistical information from the registered switch. The monitoring application will issue two requests: **OFPFlowStatsRequest** and **OFPPortStatsRequest**. OFPFlowStatsReply class's attribute body is the list of OFPFlowStats and stores the statistical information of each flow entry, which was part of FlowStatsRequest. The number of packets and bytes matched to the respective flow entry are output by being sorted by the received port and destination MAC address. OPFPortStatsReply class's attribute body is the list of OFPPortStats. OFPPortStats stores statistical information such as port numbers, send/receive packet count, respectively, byte count, drop count, error count, frame error count, overrun count, CRC error count, and collision count.

Based on the received switch statistics we can calculate the bandwidth of the network flow. Due to this limitation, we decided for another approach. We will use PACKET_IN OpenFlow message. Since this message contains a complete request in the data part of the response, we can inspect it.

4 Results

To test our solutions, we chose three data flow samples:

- A - 64 kbit/s – for example voice call
- B - 1500 kbit/s – HD video
- C - 25 mbit/s – group video chat

For testing we are using the Simple SDN topology mentioned in section 3. We prepared simple python script that creates data stream and sends it from one host to another.

4.1 Requesting switch statistics

As described in section 3 Ryu is using OFPFlowStatsRequest request to retrieve periodically flow statics. Table 1. shows results for time interval 100ms, when the Ryu application was requesting statistics from the switch periodically every 100ms. Statistics results are under column – stats. Under column measured we have results from our second solution for measuring bandwidth of the flows. We ran each test ten times and averaged them to get objective results.

Table 1. Bandwidth calculation deviation for 100ms interval

Flow	iperf [kbit/s]	measured [kbit/s]	deviation [%]	stats [kbit/s]	deviation [%]
A	82.4	79.36	3.830	101.608	18.904
B	1518.4	1544	1.658	670.304	126.524
C	24818	25351.6	2.105	32875.705	24.509

Table 2. Bandwidth calculation deviation for 500ms interval

Flow	iperf [kbit/s]	measured [kbit/s]	deviation [%]	stats [kbit/s]	deviation [%]
A	82.4	75.571	9.036	7.634	979.353
B	1518.4	1541.337	1.488	144.268	952.481
C	24818	24494.208	1.321	21809.342	13.795

Table 3. Bandwidth calculation deviation for 1000ms interval

Flow	iperf [kbit/s]	measured [kbit/s]	deviation [%]	stats [kbit/s]	deviation [%]
A	82.4	74.649	10.383	61.059	34.951
B	1518.4	1506.944	0.760	1288.0522	17.883
C	24818	24277.392	2.226	23619.751	5.073

Table 4. Bandwidth calculation deviation for 2000ms interval

Flow	iperf [kbit/s]	measured [kbit/s]	deviation [%]	stats [kbit/s]	deviation [%]
A	82.4	74.035	11.298	71.356	15.476
B	1518.4	1497.68	1.383	1331.84	14.007
C	24818	24319.784	2.048	23854.998	4.036

Tables 2 through 4 compare the results of our solution for different time intervals. Our goal is to get accurate results as soon as we can get. The time interval of 100 milliseconds and 500 milliseconds are overall the most accurate.

5 Conclusion

From the initial results mentioned in the section 4 of this paper we can see that periodic flow statistics collection and accurate bandwidth calculation can be done around 2 seconds after the flow is added to the network. This can be slow for some cases, so we would like to see better statistic updates on OVS devices. There is also second option we are currently evaluating, which provides better and faster results, as we can see from the tables above under column measured. This option is based on direct data flow measuring on the Ryu controller, until it can be effectively placed in the knapsack with the help of QFLA algorithm.

References

1. GALINSKI, M. Multimedia sessions optimization using SIP Single Port. Slovenská technická univ. v Bratislave FIIT UPAI: 2016.
2. HLAVATÝ P., GALINSKI M.: 2018. Problém batoha – Kanpsack problem.
3. VOLKO, J. Upravený problém batoha. Slovenská technická univ. v Bratislave FIIT UPAI: 2019.
4. Murawski, C. and Bossaerts, P. How Humans Solve Complex Problems: The Case of the Knapsack Problem. Sci. Rep. 6, 34851; doi: 10.1038/srep34851 (2016).
5. D. Pisinger. Algorithms for knapsack problems. PhD. Thesis, Dept. of Computer Science, University of Copenhagen, Dennmark, 2 1995.

Detection of Changes in Behavioral Biometric Characteristics in Mobile Devices

Mria Rajnkov*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkoviova 2, 842 16 Bratislava, Slovakia
xrajnikova@stuba.sk

Abstract. The theft detection system is very important for mobile users, to secure their private data. We propose a solution how to detect theft of mobile device, by monitoring data from accelerometer. In our experiments we collected data from nine participants, 262 positive and 258 negative samples. Our model can detect 97,78% occurrences of stealing from our data.

Keywords: behavioral biometrics · smartphone theft detection · inertial sensors.

1 Introduction

Mobile devices, like smartphone, contain a lot of sensitive information like photos or messages and it can be unpleasant for the user, when another users access them.

There are many ways how to secure your mobile device. The most used are PIN, password, pattern and fingerprint. These are used to authenticate owner and unlock phone. After unlocking these devices are vulnerable and thief can use them as they like.

While the device is unlocked sensitive information can be accessed quickly by reading what is on the screen. That is why it is important to detect the exact moment of theft, where it is possible to trigger another layers of protection. For example when someone steals a phone to read a message on screen, our model will detect it and turn off screen.

Every person has unique biometric characteristics. They can be divided to physical and behavioral. Physical describes characteristics of body like fingerprint, face, iris. Behavioral describes an activity, for example keystroke, gait recognition or the way of holding mobile devices. By using biometric characteristics it is possible to authenticate owner or recognize an activity[**?**][**?**]. They can also be used to detect theft [**?**][**?**][**?**]

[*] Bachelor study programme in field: Information Security
Supervisor: Assoc. Professor Daniela Chud, Institute of Informatics, Information Systems and Software Engineering,
Faculty of Informatics and Information Technologies STU in Bratislava

Biometric results are not certain like in iGuard[?]. In for example password result can only match or not. In biometrics like fingerprint recognition, there is a probability that it is a match. This is presented by TPR and FPR. TPR (true positive rate) is probability of positive result being true and FPR(false positive rate) is probability of positive result being false (we said the fingerprint match but it was someone elses).

Our method uses behavioral biometrics to detect theft of mobile device while an owner is using his phone. We detect changes in acceleration with accelerometer. These sensors are in almost all smartphone and other mobile devices. By analyzing collected data from experiments we propose a model which can detect phone theft.

2 Related work

There are multiple ways of protecting mobile device. The most common one is using PIN, password, pattern or fingerprint for unlocking the device. This approach can protect stolen device, but cannot prevent theft. Excluding fingerprint, this method is easy to copy, when thief sees user unlocking phone. This layer of protection is useless if the phone is stolen while unlocked.

When the phone is stolen we can locate it using GPS. Googles Find My Device[1] can locate phone by using only Gmail. It can also start ringing on the device or remove data. This can help retrieve the stolen device, but GPS location can be inaccurate[?]. Also, it can be disabled by thief before owner notice the theft.

Todays mobile devices contain accelerometer and gyroscope. These sensors allow recognizing an activity performed by user. There are multiple works for recognition of daily activities[?][?] like sitting, standing, walking, running and others. This can be used in healthcare for monitoring patients.

We can focus on one activity, an anomaly. For example iPrevention[?] can detect an anomaly in gait. It is used for older people to prevent falling while walking by detecting a possibility of fall. Warning user in this situation can prevent fall and injures caused by it.

Phone theft can be considered as an activity we want to detect. By similar ways we can detect theft from accelerometer and gyroscope data. It is fast and can warn owner. This action can prevent the theft and is useful in situations like pickpocket, where the victim would not notice it. This approach is used in many works and is most similar to ours.

In Detecting Phone Theft Using Machine Learning[?] authors focused on detecting pickpocket when the thief is running away. They compared logistic regression, random forest and linear SVM models for detection. Logistic regression had highest 100% TPR while random forest and linear SVM had lowest FPR 0,02%.

[1] https://www.google.com/android/find

SmartDog[?] is focused on picking up a device from pocket or bag. They detect if the action was performed by owner or not. By using DTW-based algorithm they got 94,5% TPR and 10,2% FPR.

iGuard[?] detect theft of mobile device while walking. Authors divided activity to walking(DTW-algorithm), take out motion(logistic regression) and change of speed(Cumulative Sum test). This approach helped with accuracy of detection. By using multiple algorithms they got 95% TPR and 4% FPR.

Our model is focused on recognizing theft from ordinary usage. We used Random Forest classifier inspired by results in [?] to improve results in theft detection. Our model detected more thefts then in this work and overall results all better than iGuard[?] and SmartDog[?].

3 Our approach to theft detection

In this paper we propose a model for theft detection. By collecting data from accelerometer sensor, this model can recognize if there was a theft. Data are divided to segments and from each one, we calculate vector of features. From these features model can decide if theft happened or not. This process is represented in Fig. ?? with our data.

3.1 Collecting data

For training a model we need to collect accelerometer data. We created a mobile application for collecting, with logger from Behametrics[2]. This logger creates about 50 logs per second. The application is installed on mobile phone, which is used for all experiments.

In first experiment we simulated theft to collect data samples from 9 volunteers. For negative samples (device was not stolen) we asked volunteer to play a simple game on phone. For positive samples (device was stolen) we had one volunteer for the role of victim and one for thief. While victim played the game, thief stole phone from his hand and pressed the button on main screen.

The button which is clicked after theft is used to divide samples as the end of positive sample. Currently, we collected 262 positive and 258 negative samples from these scenarios:

- Sitting
  - hands on the table
  - hands on the thighs
- Standing

At the beginning of experiment, volunteers were asked to log in. This way we collected personal data for analysis: age, height, gender and their main hand.

[2] https://gitlab.com/behametrics/behalearn

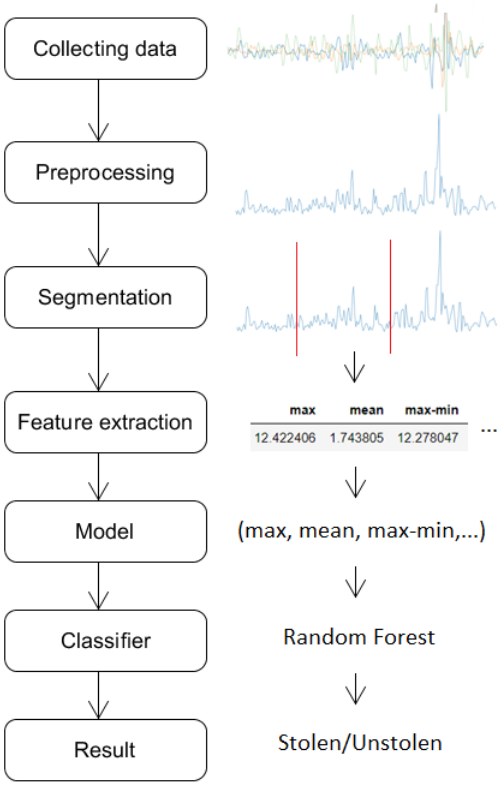


Fig. 1. Data processing

3.2 Preprocessing

Sensors data are logged in three axes: x, y, z. Our algorithm uses only total acceleration a. We calculate it by using following equation:

$$a = \sqrt{x^2 + y^2 + z^2} \tag{1}$$

3.3 Segmentation

From data collection we have files with sensors data. We need to separate each sample. We have logged times of two events: start of the game and button click after thief stole the phone. From these times we cut a sample. It has size of 220 logs (about 4.4 seconds).

For negative sample we cut the time when game was played. Their length is same as positive samples.

3.4 Feature extraction

After data are segmented, we calculate features from them. We recognize features: Maximum, mean, max-min (maximal change of acceleration), standard deviation and quantiles (25%,50%,75%).

3.5 Training model

We use random forest classifier ($sklearn.ensemble.RandomForestClassifier$), to recognize theft in our samples, represented as vectors Random forest was chosen for results in [**?**]. This work compared results from logistic regression, random forest and linear SVM. Random forest had best FPR (0.02%) but TPR (62%) needed improvement.

For validation, we divided samples into train and test in 2:1 ratio. Only train samples were used for training our model.

4 Results

Our model was tested in theft simulations and showed 97,78% TPR. It detected correctly more thefts than iGuard[**?**] or SmartDog[**?**]. It has also lower FPR of 3,66%, which means less false alarms during day.

5 Conclusion and Future work

Accelerometer is sensor which can detect mobile device theft and using it we created our model. We improved accuracy in theft detection models and raised security in this field.

In the future, we will try to further improve trained model by collecting more data and specifying or discovering more features. To show value of our model we will test it in different environments and compare our theft sample with more activities.

Interpretability of Neural Network decisions in objects and textures recognition

Peter Sakalk[*]

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkoviova 2, 842 16 Bratislava, Slovakia
sakalikp@gmail.com

Abstract. Recognition and description of visual properties of objects are intensively investigated areas of computer vision and deep learning with significant application in robotics, augmented reality, or industrial applications. However, having excellent results is not enough. In many cases, it is necessary to know the reasoning behind the decision of neural network. Deep learning cannot be applied in many domains as it is not trusted because of its black box nature. Recently, research in the field of deep learning has focused on improving the level of decisions explanation. This has led to the development of numerous explanatory methods. In this work, we propose a method based on two of the existing solutions. The first is Layer-wise Relevance Propagation which we use to calculate the relevance of the elements. The second is Occlusion sensitivity which we use to determine if an element affects a particular class. The goal of this work is to provide a tool for users to visualize decision-making information produced by a deep learning approach. We apply this method to models trained on CIFAR-10 and perform various experiments to prove its applicability on other more difficult datasets.

Keywords: Interpretability · Neural network · Computer vision.

1 Introduction

Interpretability of complex machine learning models, such as deep neural networks, presents a significant challenge. Even though, these complex models reach impressive results, they are not in a state that would be sufficiently intuitive and understandable for people who should benefit from these interpretations. This inability to explain their own behavior is a cause for deep neural networks to have difficulties breaking into areas such as medicine or the financial sector. In these areas, we cannot rely on decisions that we cannot explain adequately, because the possible prediction error can be responsible for misdiagnosis or loss of finances.

[*] Bachelor study programme in field: Informatics, Supervisor: Ing. Luk Hudec, Institute of Computer Engineering and Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

There are several methods of interpretability. Most of them, however, focus on explaining how individual pixels affect the overall predictions. However, they do not tell us which specific parts of the image affect the behavior of the neural network. Therefore, we propose a method that combines these two approaches.

In our work, we focus on increasing the interpretability of deep neural networks. We do not just want to find out the impact of individual pixels but to solve the focus of the network. We aim to analyse, if the network responds to the hierarchy of the local patches or searches for the whole objects or is looking for only some abstract representations.

The paper is organized as follows. In section 2, we present representative works related to the proposed approach. In section 3, we define our method, and in section 4, we present our early experiments. Section 5 concludes this paper.

2 Related work

This section presents various several works that deal with interpretability of neural networks. These methods can be divided into three categories according to the basis of their main approach.

1. **Backpropagation based** methods are based on computing a contribution of each pixel to the result by back-propagating the signal through the network.
2. **Perturbation based** methods are based on occlusion analysis, where only a part of an image is shown to a network to determine which local areas contribute the most to the decision making.
3. **Layer-wise relevance propagation** methods distribute the relevant score of neuron among lower layer neurons to compute the relevance of an each pixel.

2.1 Backpropagation-based

The process of this method is performed during one forward pass and related backpropagation. We assign a contribution to each pixel. It is faster but less accurate than other methods. One of such methods is Sensitivity analysis [**?**]. By this method, we examine what impact each feature has on the models prediction. In this method the attributions are computed by taking the absolute value of the partial derivative of the target output with respect to the inputs.

Another work applying back-propagation for interpretability is Guided backpropagation [**?**] method. It represents an improvement for Sensitivity analysis where we consider only gradient with a positive loss. In this method, neurons perform as detectors of particular image features. The method is focused on highlighting the most important image features, that neurons detect and are contributing to the classification. The other areas of the image are being ignored.

2.2 Perturbation-based

Also known as Occlusion Sensitivity methods [?, ?] systematically mask various parts of the input image and feed it to the network. Then, they compare the results and analyse how the network confidence changes. The masked areas are usually covered by zero or mean values that would not affect the network decision making. The methods consist of three fundamental steps.

- The change of the original image - overlaying a portion of it or disrupting the input pixels
- Feeding the network with the modified image to get a prediction
- Comparing the predicted output of modified image to the output received from the original image.

The strength of this approach is that we are very likely to localize the most critical areas of the image. On the other hand, the disadvantage is the high computational effort compared to other approaches and difficulty to select a sufficient size of the overlaying window.

2.3 Layer-wise relevance propagation (LRP)

LRP [?,?] is a method which is performed after the prediction during a backpropagation phase. Using this method, we can tell which pixels were most relevant to the resulting prediction. Its crucial feature is conservativeness. It means that the relevant score of neuron is redistributed among the lower layer neurons with which it is connected. Pixels can acquire either a positive or negative contribution. Contributions can be propagated across the network by multiple rules. It is not recommended to apply one rule to the entire neural network, because of irrelevant output. The advantage of this method is that it is fast. We obtain the output in the form of a heatmap after performing one forward and backward propagation.

3 Method

We applied our interpretation approach on a convolutional neural network with a simple architecture called "Simplenet" [?] by which we achieved almost state-of-the-art results on the used dataset CIFAR-10. It is a network with optimized number of parameters and robust enough for the most common classification problems. Its advantage is that it does not contain tens to hundreds of millions of parameters and does not require complicated regularizations during training. The lower number of parameters also contributes to fewer memory requirements. However, its disadvantage is lower performance compared to robust and complex network. Despite this deficit, it can exceed them on some problems. It is proven as a good compromise between low computational memory complexity and classification accuracy. To obtain satisfying results of interpretation, the model is trained to achieve 90% accuracy on the dataset CIFAR-10.

Step 1 - Applying LRP First step of the method is to feed forward the analysed image through the network and obtain the prediction. After the prediction, we apply the LRP method on one of the classes from the dataset. We use it to identify regions of interest that help us speed up the Occlusion sensitivity Fig.**??**. So we don't have to go through every single pixel, and we can only focus on the most important ones. We combine two rules of LRP interpretation approach. Specifically LRP-ϵ, LRP-γ [**?**]. LRP-ϵ is used in higher layers where there is a more dense interconnection. There are fake contributions that may be distracting. With this rule, we can remove these fake contributions so that only the best ones remain. The second rule LRP-γ we use for the input layer. This rule focuses on the whole image and not only on individual pixels, and then the output heatmap is understandable.

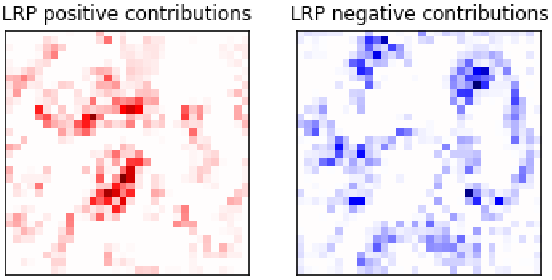


Fig. 1. Output after LRP application.

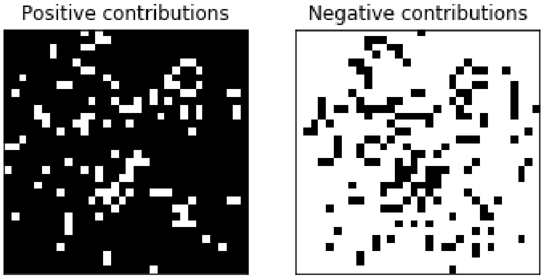


Fig. 2. Output after threshold function application.

Step 2 - Selecting regions of interest On this heatmap, we apply the threshold function Fig.**??**. Because, even after applying LRP, we still have too many pixels for Occlusion sensitivity and we need to reduce that number. We do it in two ways.

In the first one, we use adaptive thresholding before the global threshold because the global threshold leaves too large pixel spots in some places. That is why we use an adaptive threshold with a smaller block size to get only the most significant pixels out of them. Specifically, we use the gaussian-weighted sum of the neighbourhood values, because it is not as noisy as other threshold functions. The result is pixels on which we apply Occlusion sensitivity.

In the second one, we apply Gaussian blurring on the output from LRP. This way, we achieve a smoother pixel transition and bigger regions. Now we use the adaptive threshold function based on the mean of the neighborhood area. Because of the Gaussian adaptive threshold function divides individual regions into smaller ones. Then we calculate the mask size separately for each region. The minimum mask dimensions are 7x7 pixels.

Step 3 - Occlusions Finally, we apply the Occlusion sensitivity analysis over the pixels obtained from the threshold function. It tells us how much the prediction changes if we cover the most critical pixels and their surroundings. We perform a series of experiments on the altered images and the following prediction. We cover the images in two ways. In the first one, we use a single color mask. It is one of the most commonly used type of masks in this method. We are interested in the difference between this type of mask and the more sophisticated type of mask. In the second, we cover the pixels in the mask using the gradient inpaint. Its task is to replace the pixels in the mask with the pixels around it. The aim is to have the mask resemble the original image. It looks natural, so it is more confusing for the neural network. The size of the mask depends on the size of the image. We intend that the mask occupies at least 20% of the image. Otherwise, the mask covers only a negligible part, which has a minimal effect on the overall prediction.

4 Evaluation

We follow two experiments in this section. The first experiment is focused to verify the importance of regions marked by LRP as the most significant for the classification to the correct class. The second volume of experiments is focused on verifying the areas with the lowest and even opposite contribution to the correct classification. We performed a set of experiments on various images. We evaluate the results through empirical analysis, as there is currently no universal metric for interpretability evaluation. The experiments are focused on evaluating the final confidence value of the network with respect to the correct class.

We conducted occlusion analysis with 4 variations, the solid color white, black and mean (grayscale image = 127) and gradient inpaint to occlude the selected regions of interest. The first row in the figures (Fig. **??**, **??**, **??**) represents the original image and it‘s prediction, the output from the LRP, and the output from the threshold function. The second to fourth rows in the figures represent the application of different types of masks in the order of white, gray, black, gradient inpaint. These images represent regions that have an impact on prediction and their prediction. The number of images in these rows represents how many times the local minimum or maximum (according experiment) of the prediction is exceeded.

Occluding regions important for correct class We select the most critical regions from the LRP output and then perform the Occlusion sensitivity according to step 3. As we gradually cover the regions from the most influential region, we observe whether the prediction is falling. In this way, we find out what role the region played in making decisions. When we use solid color masks, the network focuses only on the strongest regions and no longer falling or is stuck in one region. When we use gradient inpaint mask, the predictions also fall in regions that contributed less to the overall prediction (Fig. **??**, **??**). We recorded similar results in several images from dataset CIFAR-10.

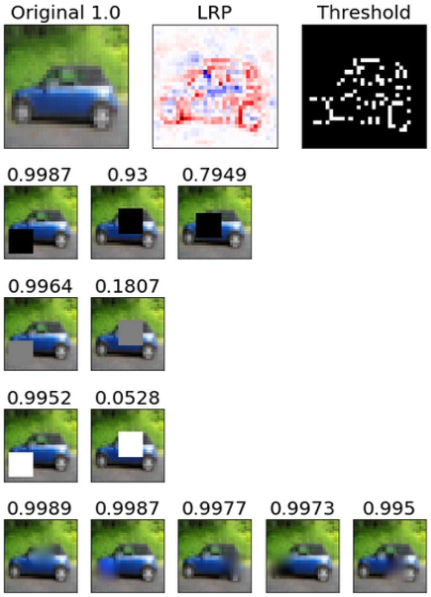


Fig. 3. The collection of images represents the application of the first experiment on the simple image of the car, that is one of the test images from CIFAR-10 dataset. We covered most parts by using gradient inpaint. Solid color masks covered less and similar parts. The numbers above pictures represent their prediction. The number of images in rows 2-4 represents how many times the local minimum of the prediction is exceeded.

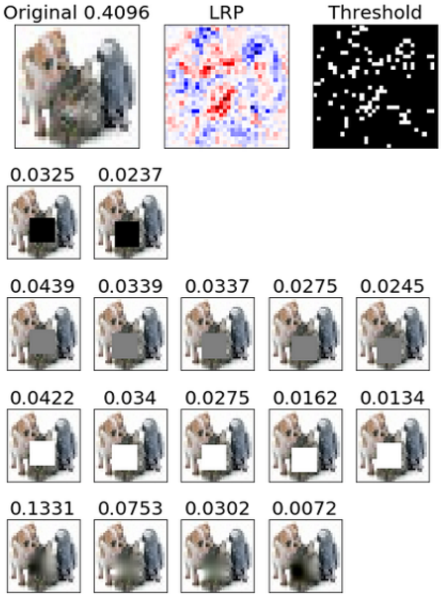


Fig. 4. The collection of images represents the application of the first experiment on the complex image of the animals (dog, cat and bird), that does not come from CIFAR-10. Focused on cat. All masks covered similar parts in surrounding of the cat face. The numbers above pictures represent their prediction. The number of images in rows 2-4 represents how many times the local minimum of the prediction is exceeded.

Occluding regions with opposite importance As in the previous experiment, we also focus on LRP regions. Now we select those that negatively influenced the overall prediction. By using the Occlusion sensitivity as described in step 3, we gradually cover the regions. We start from the region that was most negatively marked by LRP. We observe whether the prediction of the class is rising. In the images from the dataset CIFAR-10, we achieve a prediction of almost 100 percent and can be changed only minimally. For example with this experiment we did not find any ROI in the original image on Fig. **??**. Therefore, we are not doing this experiment with these images. For more complex images Fig. **??** where are multiple objects, the experiment works better. We see that the prediction increases when we cover parts of other objects.

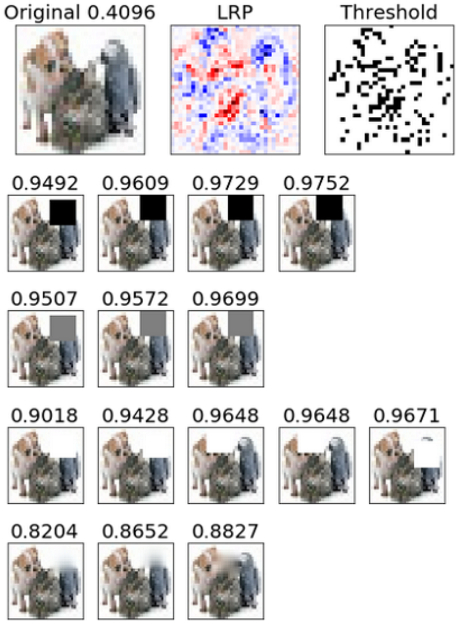


Fig. 5. The collection of images represents the application of the second experiment on the complex image of the animals (dog, cat and bird), that does not come from CIFAR-10 dataset. Focused on cat. White mask matches with the background, but does not reach the higher prediction. The numbers above pictures represent their prediction. The number of images in rows 2-4 represents how many times the local maximum of the prediction is exceeded.

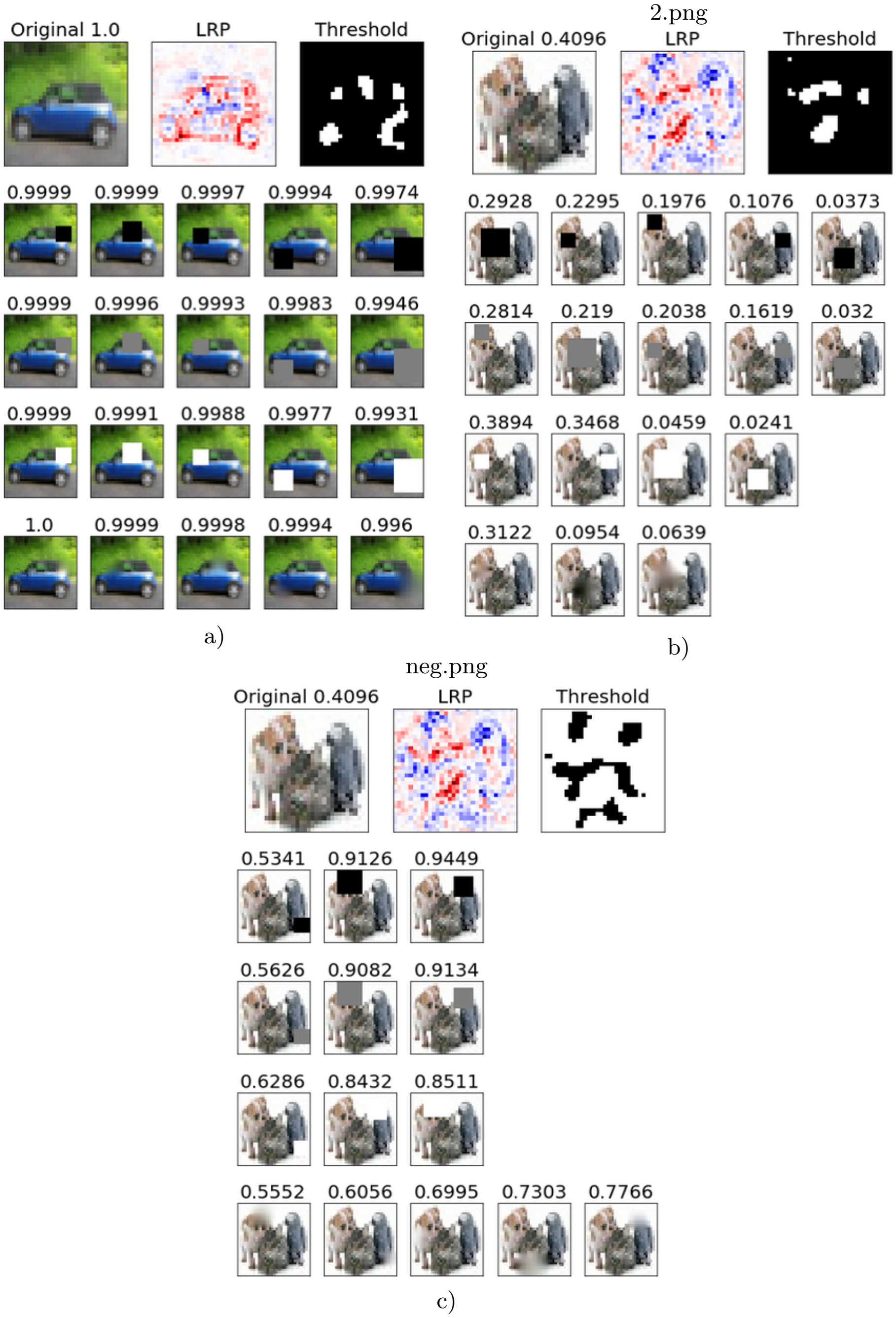


Fig. 6. The collection of images represents the application of choosing ROI by selecting blobs. The number above images represents reached prediction. The number of images in rows 2-4 depends on how many times prediction decreases under original prediction (Occluding regions important for correct class) or increase over (Occluding regions with opposite importance). a) image from dataset CIFAR-10. The prediction is changing with minimal effort. The prediction is changing only in a minimal way. b) and c) the change is more significant but not as much as when we select pixels. Both are focused on object cat.

5 Conclusion

In our work, we have experimented with two existing methods that deal with the interpretability of neural networks. Both methods have their pros and cons. We have experimented with and evaluated different approaches to improve the interpretability strength of the mentioned methods. We did not want our method to focus only at pixels, but to show the individual parts of the objects that contributed positively or negatively to the prediction.

Our method helps to understand how well the network is trained and what parts of images and objects it uses to make its decisions and predictions. The presented figures demonstrate by covering the regions of the most contributing pixels and object parts, that the network learns the object representations hierarchically as local features and attributes of an object. In Figure **??** it is visible how the network confidence is lowering, due to higher occlusion of the location of the classified cat. In Figure **??**, the most important regions are around the wheels and the car roof. Therefore, it is possible, that the network is focusing on hierarchical local features. If such trend occurs in all networks, it could be possible to analyse and understand its behavior in more difficult domains such as histology or finance markets, where understanding of the features used for decision is on more abstract level.

We choose ROI in two ways from the threshold function. In the first one, we select pixels. In the second one, we select blobs. When we select blobs, we observe a significant reduction in the number of regions. Therefore, in the first experiment, we select every region where the prediction decreased against the original (Fig. 6a, 6b). In the second experiment, we select every region where the prediction increased against the original Fig. 6c. The network is now focusing on more regions of the object. On the other hand, the prediction will not change as much as in the first way.

Wojciech Samek also deals with the topic of interpretability. He is the inventor of LRP. He works with both methods that we used in our work. His works are available at webpage heatmapping.org.

In the future, we would also like to conduct further experiments on dataset of much higher variability, like texture or histology data, to test our approach.

References

1. Alexander Binder, Sebastian Lapuschkin, Gregoire Montavon, Klaus-Robert Muller, and Wojciech Samek. Layer-Wise Relevance Propagation for Deep Neural Network Architectures. 01 2016.
2. Marco Ancona, Enea Ceolini, A. Cengiz Oztireli, and Markus H. Gross. A unified view of gradient-based attribution methods for deep neural networks.CoRR, abs/1711.06104, 2017.
3. Seyyed Hossein Hasanpour, Mohammad Rouhani, Mohsen Fayyaz, and Mohammad Sabokrou. Lets keep it simple, using simple architectures to outperform deeper and more complex architectures. arXiv preprintarXiv:1608.06037, 2016.

4. Wojciech Samek, Grgoire Montavon, Andrea Vedaldi, Lars Hansen, and Klaus-Robert Muller. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. 01 2019.
5. Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps.CoRR, abs/1312.6034, 2013.
6. Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net.CoRR,abs/1412.6806, 2014.
7. Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks.CoRR, abs/1311.2901, 2013.

IIT.SRC 2020 Conference
User identification and authentication enhanced with behavioral biometrics

Vladimír Jančok

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia

vladimir.jancok@stuba.sk

Abstract. In this paper we investigate the possibility of identifying users based on the mouse usage patterns and movement dynamics. This method can enhance the traditional login-and-password approach when attempting to gain access to a system. In our experimental setup, we use the standard mouse to gather the user data. When performing the experiments, we extract the appropriate behavioral features from the user actions. The resulting user model supported by machine learning modules can be used for later static or dynamic authentication and improve the security level of authentication system.

Keywords: User identification, Mouse movement dynamics, Biometrics.

1 Introduction and Related Work

1.1 Behavioral Biometry and Pointing Device Research

Biometric characteristics such as mouse movement dynamics or keyboard writing activities performed in computer system are characteristic for each user. Mouse dynamics such as mouse clicks or movements, can be reliably used as source for feature extraction and processing.

We can use these features when modeling user for identification, recognition of emotional status, detection of computer experiences [2]. In our work, we are considering multiple combinations of most prevailing user modeling techniques.

A number of studies have been devoted also to user authentication using standard input devices [4, 6, 8] where the efficiency of user verification by means of mouse movement dynamics was elaborated with promising results.

[1] Doctoral study program in field: Intelligent Information Systems / Supervisor: Doc. Daniela Chudá, PhD. / Faculty of Informatics and Information Technologies STU in Bratislava

1.2 Input device data processing

In this paper we are primarily focused on web-based environment where computer mouse or touchpad is used, however similar methods can be applied to mobile devices as another source of biometric input.

Data processing – recorded data are stored in log files that can be split to samples according to mouse action events. Samples are further on pre-processed, if missing values occur, filtered or calculated. Data can be grouped and linked with selected user actions that subsequently simplifies the feature extraction.

Computer mouse can be a valuable source of data, generating events such as mouse movement, mouse-down (button press), mouse-up (button release) and wheel scrolling. We collectively refer to events as raw data. Each event is associated with coordinates of the cursor and the time of occurrence of the event,

Various universal metrics can be calculated from the mouse actions, such as curvature or velocity. Each metric is further processed to compute a single value called feature. A vector of features computed from metrics is referred to as instance and is added to a biometric profile of the user [1].

Feature Extraction – basic features related to the mouse movement dynamics are based on direction, angle, distance and velocity [8].

Direction based features are calculated as the relative number of movements in a particular direction.

Distance based features are calculated as the ratio of full paths to shortest path. For each *n* pixel length movement pair the angle is calculated between them and average angle and standard deviation of angles is used as a feature. For each defined pixel length movement the velocity is calculated and the standard deviation of velocity is used as a feature.

Information gain – when considering the selection of appropriate features for building the biometric user profile, the following calculation can be used for the feature and class dependency measures.

$$IG(f,C) = H(f) - H(f \mid C) \tag{1}$$

$$H(f) = -\sum_{v \in V(f)} \left[p(v).log_2\big(p(v)\big)\right] \tag{2}$$

Where *H(f)* is feature entropy and *H(f | C)* is conditional feature entropy. Information Gain *IG* is defined by the probability *p(v)* that feature *f* gains value *v*.

Machine Learning –increased precision in the field of behavioral biometric is in most recent research works [4, 7] achieved by Machine Learning (ML) methods. We can evaluate the most commonly used ML methods with focus on the accuracy of features through a comparison of different algorithms, i.e., Naive Bayes classifier (NB), Decision Trees (DT), Random Forests (RF), Support Vector Machines (SVM) and Multi-Layer Perceptron (MLP). Several experiments confirmed that SVM offers a good trade-off between precision and recall [5]. Classification ML methods can also decrease the amount of mouse actions required for verification, if the user model is trained with sufficient training set.

2 User Model for Identification and Authentication

Our research focuses on the area of building profiles with computer system for identification and increased accuracy of user profiles for different types of users. Naturally, in a web-based environment, users work with the computer mouse much more than with the keyboard [3]. User modeling heavily depends on information provided by the user.

2.1 User model building techniques

The simplest way of building a user model is explicit user modeling, when the user is asked to provide relevant information directly. However, a big challenge of research is to minimize obtrusion and therefore implicit user modeling is trending. This approach relies on feedback from the user in the form of actions [3].

It is common for the same user to use different kinds of pointing devices from time to time [8]. For example, a laptop user may use the on-board touchpad for some time, but switch to a USB-connected mouse when a larger desk space is available. Therefore, it is desirable to detect the type of pointing device being used in a nonobtrusive way. In other words, the question we attempt to answer is, given a series of cursor movements: Is it possible to tell if they are from a touchpad or a computer mouse?

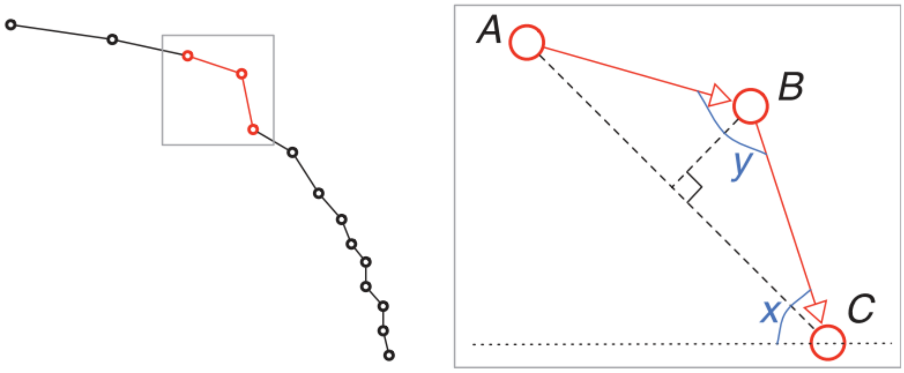


Fig. 1. Illustration of angle-based metrics and calculations (N. Zheng at al., 2016)

2.2 Mouse movement and actions evaluation

Like all biometric systems, mouse dynamics authentication systems involve an enrollment phase and a verification phase. There is still a reasonable potential in increasing the accuracy, speed and cost of biometric authentication in web environment that have become the primary focus of our research.

There were two approaches of user identity verification:

a) *Static* - collect and verify a user's mouse data at specific times (e.g. at login time). Features computed from the user's movement between each pair of dots comprise the enrollment signature. Authenticating involves the same series of dot-to-dot movements, which are compared against the enrollment signature. Mouse movements can be recorded through JavaScript embedded in the web page and sent to a server for processing.

b) *Continuous* - collect and verify the user's mouse data repeatedly throughout the entire session. Mouse events are aggregated as higher-level actions such as point-and-clicks or drag-and-drops, characterized by action type, distance, duration and direction. Consecutive actions over some time frame are grouped into sessions [7].

Table 1. Research works related to the pointing device usage biometrics.

Study	*Identity verification*	*FAR*	*FRR*	*Participants*
Shelton et al. (2013)	Static authentication	25%	n/a	8
Gamboa et al. (2007)	Static authentication	6,2%	6,2%	50
Mondal et. al. (2013)	Continuous authentication	n/a	n/a	49
Feher et al. (2012)	Continuous authentication	1%	1%	25
Zheng et al. (2016)	Continuous authentication	1,3%	1,3%	30

3 Evaluation of Proposed Model

In order to fully verify the updated user model, several experiments will need to be conducted. In our conditions up to 30 participants can be considered as a sufficient sample.

Once the samples are extracted from the data, we perform the classification (user identification) for each type of event separately. Since the number of samples can vary greatly across users, it can be temporarily reduced also up to 30 for each type.

Experimental dataset may contain selected mouse events for a particular session for each user performing the defined tasks on a website.

Machine learning module is responsible for data preprocessing, feature extraction and user authentication.

In relation to the Information Gain calculation defined in the section 1, the entropy of our dataset in terms of the probability distribution of observations in the dataset belonging to one class or another, sklearn library *(e.g. sklearn.feature_selection.mutual_info_classif(X, y, discrete_features='auto', n_neighbors=3, copy=True, random_state=None)* calculate the Information Gain for each selected attribute.

Our proposed solution is written in Python 3 language, using the following libraries: sklearn, pandas, numpy and bokeh. Machine learning module can also provide data visualization functionality.

Table 2. Sample of mouse movement logged data per user session.

input	session_id	timestamp	event_type	button	x	y
cursor	5e428659f6ba2b00ec9b1cbe	1581418078441	move	null	428	310
cursor	5e428659f6ba2b00ec9b1cbe	1581418078455	move	null	427	310
cursor	5e428659f6ba2b00ec9b1cbe	1581418078471	down	left	423	311
cursor	5e428659f6ba2b00ec9b1b70	1581418078486	up	left	416	314
cursor	5e428659f6ba2b00ec9b1b70	1581418078502	down	left	406	317

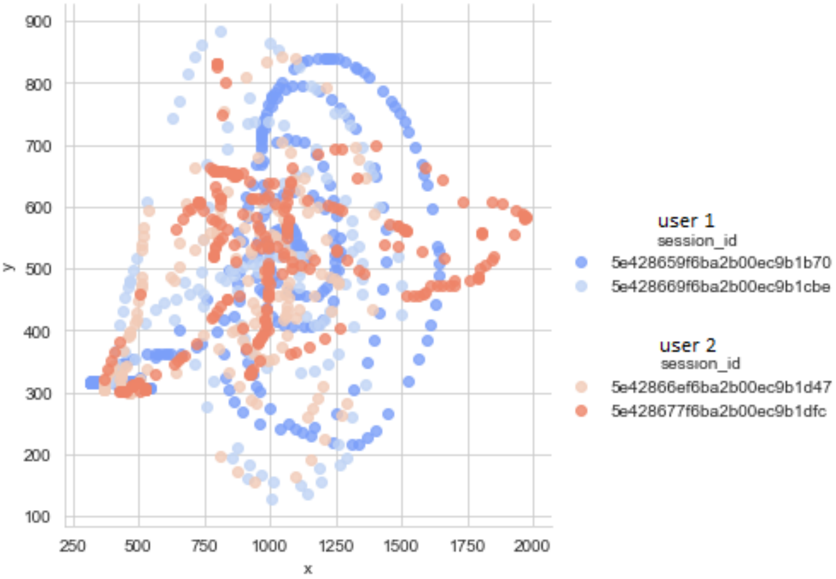


Fig. 2. Visualized sample of logged mouse cursor movement of *2 users* in 2 sessions selected from experimental dataset

The sample was selected from 28 users where each session contained several mouse event actions. Particular differences in the ange-based metrics can be spotted between users visually, where *user 1* and *user 2* show similar behavior during both sessions.

4 Conclusions and Future Work

In this paper we introduced and partially elaborated a user model for biometric identification based on standard input devices such as mouse and keyboard applicable in general use web environments. In comparison with existing user identification or authentication approaches the current state of our proposed approach to user identification should be scaled to more users in order to confirm the experimental results.

Future work will include improving the identification accuracy and speed by selecting the most relevant and least redundant individual features rather than considering entire groups of features or choosing their most commonly used subset. Dynamic user identification should be also included.

There are some function modifications needed in order to minimize the percentage of false accepts and more detailed elaboration of outliers when collecting data is planned as well.

Supervised machine learning algorithms capable of performing classification, regression and outlier detection such as Support Vector Machine (SVM) are going to be further elaborated. We will use our model to predict the classes of the samples in the testing set and increase the accuracy of the classifier.

The main challenges in the research area of user identification and authentication are precision of evaluation and the amount of data and subsequently the time required for a particular authentication decision.

Higher input device hardware independence of the proposed model is also an interesting motivation for the future research.

Acknowledgments: This work was partially supported by the Slovak Research and Development Agency under the contract No. APVV-17-0267 - Automated Recognition of Antisocial Behaviour in Online Communities and project Modelling, prediction and evaluation of user behavior based on the web interaction for adaptation and personalization the Scientific Grant Agency of the Slovak Republic, grant No. VG 1/0667/18.

References

1. D. Chudá, D. Krátky, K. Burda: Biometric Properties of Mouse Interaction Features on the Web Interacting with Computers, iwy015, https://doi.org/10.1093/iwc/iwy015, (2018)
2. P. Krátky and D. Chudá: "Estimating Gender and Age of Web Page Visitors from the Way They Use Their Mouse," in Proceedings of the 25th International Conference Companion on World Wide Web - WWW '16 Companion, pp. 61–62 (2016).
3. P. Krátky and D. Chudá, "Recognition of web users with the aid of biometric user model," J. Intell. Inf. Syst., vol. 51, no. 3, pp. 621–646, Dec. 2018.
4. S. Mondal and P. Bours, "A study on continuous authentication using a combination of keystroke and mouse biometrics," Neurocomputing, vol. 230, no. November, pp. 1–22, 2017.

5. P. Krátky, T. Repiský, and D. Chudá: "Is the Visitor Reading or Navigating?" in Pro-ceedings of the 18th International Conference on Computer Systems and Technologies - CompSysTech'17, pp. 80–87, (2017).
6. C. Shen, Z. Cai, X. Guan, and J. Wang, "On the effectiveness and applicability of mouse dynamics biometric for static authentication: A benchmark study," Proc. - 2012 5th IAPR Int. Conf. Biometrics, ICB 2012, pp. 378–383, (2012).
7. S. Mondal and P. Bours, "A study on continuous authentication using a combination of keystroke and mouse biometrics," Neurocomputing, vol. 230, no. November, pp. 1–22, (2017).
8. A. Pentel, "Predicting Age and Gender by Keystroke Dynamics and Mouse Patterns," (2017).
9. J. Shelton, J. Adams, D. Leflore, and G. Dozier, "Mouse tracking, behavioral biometrics, and GEFE," Conf. Proc. - IEEE SOUTHEASTCON, (2013).
10. N. Zheng, A. Paloski, and H. Wang, "An Efficient User Verification System Using Angle-Based Mouse Movement Biometrics," ACM Trans. Inf. Syst. Secur., vol. 18, no. 3, pp. 1–27, Apr. (2016).

Support for development of digital systems using modeling and visualization

Krištof Orlovský[*]

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
kristoforlovsky@gmail.com

Abstract. Hardware description languages have been widely used in domain of hardware design from their invention up to this day. They brought countless advantages in domain of hardware design, but can still suffer from the limitations of their purely textual form. We can mention low presentability, hard documentation and potential high understanding complexity. These shortcomings were the source of motivation for this paper. Our goal in this project is to eliminate this limitation in means of graphical visualization using block diagrams. We proposed essential steps, processes and concepts which were identified during the visualization of HDL designs. These steps include specification of the goal block diagram using proposed metamodel and introducing solutions how to represent and store diagram visual information using JSON data format from which we can later on generate the source code. The result of our work so far is a prototype application, which represents the realisation of our solutions. The prototype is a web integrated development environment implemented using microservices architecture. This IDE allows its users to create and edit their own diagrams representing HDL design. The solution can generate these diagrams from SystemVerilog source code and vice versa. We also enumerate further approaches in this domain.

Keywords: SystemVerilog, Hardware Decription Language, graphical visualization, ASIC design, digital system development, block diagram

1 Introduction

Hardware description languages, shortly HDL, were from their creation widely used in the domain of hardware design because of the higher level of abstraction they brought to complex integrated circuit (IC) description than other approaches. Furthermore, we can mention more advantages: earlier design functional verification, describing IC

[*] Master study programme in field: Software Engineering
Supervisor: Dr. Lukáš Kohútka, Institute of Computer Engineering and Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

designs is analogical to programming and many more, thoroughly identified in multiple works [1, 2].

The first HDLs has already appeared in 1960s and their concept lasts and stays popular to this day. We can observe how essential the usage of these languages is in other works, for example S. Palnitkar writes "With rapidly increasing complexities of digital circuits and increasingly sophisticated CAD tools, HDLs will be probably the only method for large digital designs. No digital circuit designer can afford to ignore HDL-based design." [1].

Despite the multiple advantages of this approach in this paper, we will focus on the limitations of its plain text-based usage. Digital designs can be more complex or larger in size if we describe them in plain text form without any form of graphical visualization. This can result in multiple drawbacks. Briefly, we mention a few: error vulnerability, worse readability (bigger complexity), poor presentation and documentation possibilities. As we mention later in this paper, (in section 3. Related work) there are already existing applications that solve this problem to a certain level. What these solutions lack is the hierarchical abstraction visualization. Under this term we mean that a whole block diagram in our solution can be reused, using abstraction, as one block (module) representing the whole functionality in another diagram. Some of the analysed solutions allowed this abstraction but could not visualize it. What we propose is a possibility to hierarchicaly visualize this abstraction. That means users can view the whole lower abstraction diagram from the parent diagram. Next these solutions do not allow continuous transition between code and diagram, which we try to implement.

In this paper, we propose our solution for upper-mentioned limitations using visualization to ease or eliminate them. Simply outlined, our solution is a creation of an online web integrated development environment (later web IDE), which enables and enhances the work with development of digital electronics designs for single user, but also for groups of cooperating users. The current key elements of the proposed IDE are: text editor with SystemVerilog language support and functionality, such as static text analysis, but also dynamic help for developers in the form of information delivery about existing defined variables, signals, ports, modules and other attributes. Another goal of our web IDE is integrating the elements needed to support collaboration of multiple authors and version-control functionality. Emerging from the previous objectives, the next one is that there is a need to design and implement data storage and archive the created designs. And finally, we are getting to our primary goal of visualizing SystemVerilog, which consists of creating and editing IC designs using block diagrams. The upper mentioned objectives represent a "big picture" idea. The resulting system will not be a work of one thesis, but of a collaborating multiple theses. Our specific primary goal is the **feature of graphical visualization using block diagrams**.

2 Visualization of SystemVerilog language

In this section, we dive into the used processes and options of SystemVerilog language visualization. Right from the start, it is important to define specific design of our proposed visualization. Next, we propose a serializing and archiving format for the created diagrams.

Our main goal from the user experience perspective is to achieve continuous two-way visualization. This means that the IDE users should be able to switch between the code and diagram and vice versa on demand at any time during their development.

2.1 Block diagrams

We chose to visualize SystemVerilog using block diagrams (See **Fig. 2.**). This decision was based on multiple existing papers (e.g. [3, 4]) and tools [5, 6], which used similar diagrams. This way we can capture and clarify the overall concepts, structure and relations of digital designs without concern for the details of implementation.

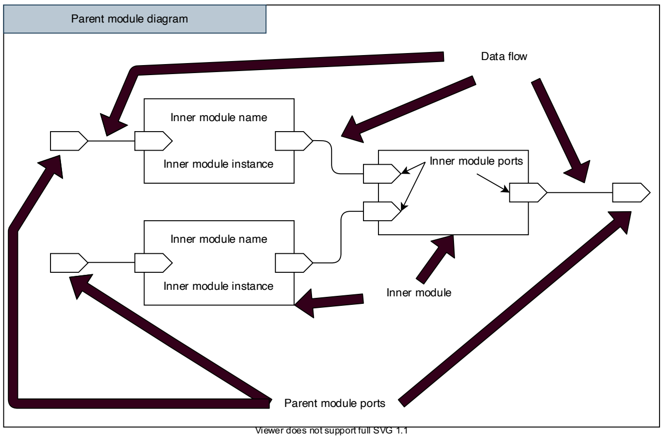


Fig. 2. A brief example of an explained generic block diagram used for our goal completion

In **Fig. 2.** we briefly explain the specific elements used in our proposed block diagram. In our case, the next advantage of block diagrams is the ambiguousness of the block. This means that with the right specification, we can propose our own type of diagram in which we specify, what the block does and its body mean. For this purpose, we designed a metamodel (**Fig. 3.**) describing and specifying our goal block diagram.

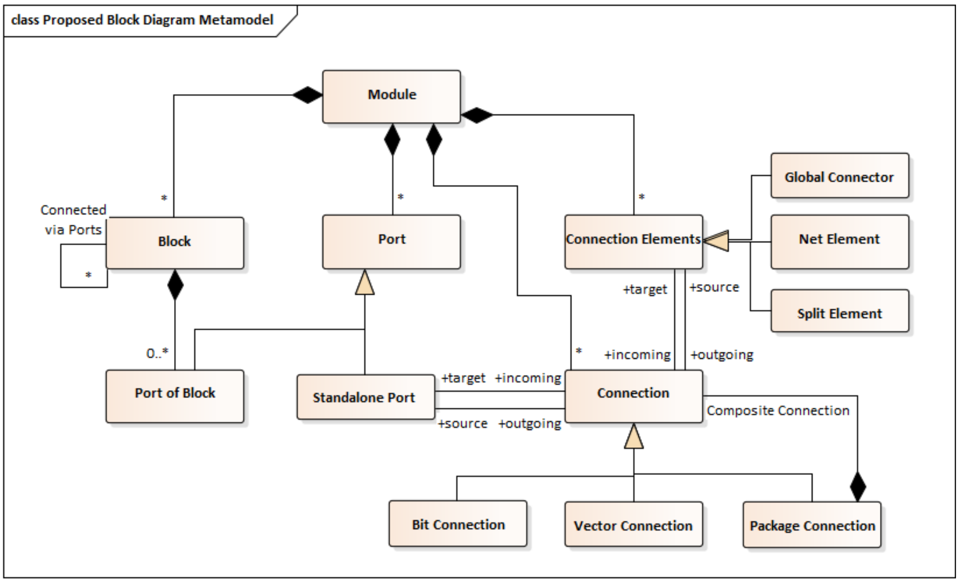


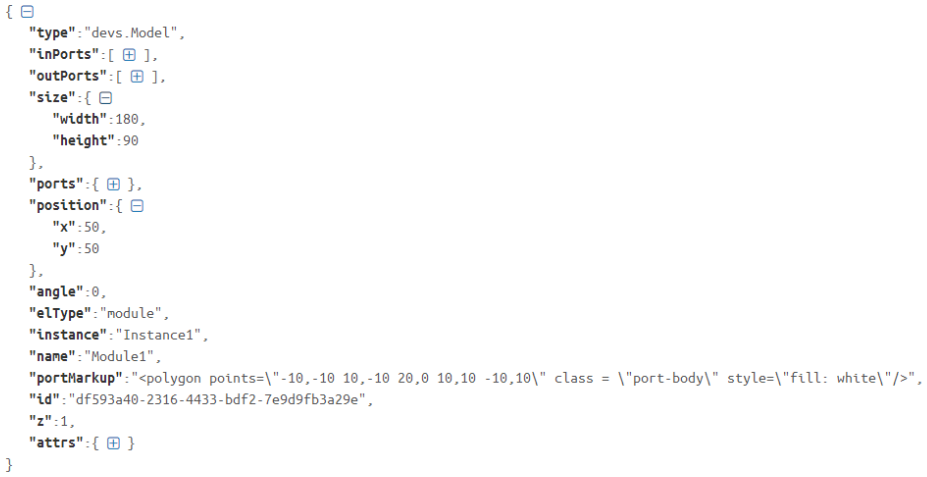
Fig. 3. Metamodel, in progress, specifying our proposed block diagram

In our designs we use abstraction to achieve modules hierarchy. It means, that a module (block) in one diagram can be a representation of a real single module, or another whole block diagram. This way we achieve a possibility to build unique functional blocks/modules and we also greatly increase reusability of modules.

2.2 Module serialization

As the nature of the solution indicates, the end users does not have to carry their designs always with themselves. One of our goals is to create a fine way to save and archive designs in the progress and then later deliver them on demand. This requires a well created data format to store. Usual approaches are XML (example [3, 4]) or Object Oriented representations (example [7]).

During our analysis and previous work, we proposed a custom XML representation [8], similar but simpler than K. Jelemenská et al proposed in their paper [3]. Now, however, we are using an open-source diagraming library JointJS[9] for core visualization. It uses JSON objects to represent and preserve the inner structure of diagrams and so the both, simplest and most synchronized way, how to store diagram information was to reuse these JSON objects. The only thing we did was that we enhanced the objects to serve our cause. Most of the time, it meant containing more information (**Fig. 4.**).



```
{ ⊟
   "type":"devs.Model",
   "inPorts":[ ⊞ ],
   "outPorts":[ ⊞ ],
   "size":{ ⊟
      "width":180,
      "height":90
   },
   "ports":{ ⊞ },
   "position":{ ⊟
      "x":50,
      "y":50
   },
   "angle":0,
   "elType":"module",
   "instance":"Instance1",
   "name":"Module1",
   "portMarkup":"<polygon points=\"-10,-10 10,-10 20,0 10,10 -10,10\" class = \"port-body\" style=\"fill: white\"/>",
   "id":"df593a40-2316-4433-bdf2-7e9d9fb3a29e",
   "z":1,
   "attrs":{ ⊞ }
}
```

Fig. 4. JSON representation of block diagram elements. Original schema of JointJS [9] library edited for our purposes.

3 Related work

During our research in the problematic domain we discovered multiple similar solutions. These solutions are sourced from distinguished spheres so we mention a few related works from different sectors.

We would like to begin with the research sector, because this is the closest to our own paper. In academic works and papers we do not have to go further, than just our own university. Multiple papers from multiple authors, but all including the author K. Jelemenská, have devoted themselves to this issue [3, 4, 10]. Many of the conceptual approaches in these papers are similar to ours and at the beginning they served as initial point of our project. The main differences are on the practical level including implementation details. As the papers suggest [3, 4], they work with another HDL languages. However, the differences are also in the architecture and used technologies of completed prototypes. For example in “VHDL Structural Model Visualization” [4] autors use .NET, C# and explicitly expressed their aim in Windows operating system platform, while our approach tries to be more modern and more platform independent. In addition to this work, we ourselves participated in similar project in the past, this time only in the form of a more monolithic C++/Qt application that also visualized SystemVerilog [8].

Lets proceed to the commercial sphere of existing industrial solutions. In this sector, again, multiple similar solutions can be found. From the perspective of software quality, these solutions are on high level, probably only excluding the usability and portability aspect of software quality. This is of course due to the size and complexity of these software solutions. In our opinion, the main problem with these products is their price and inaccessibility for an ordinary user. We tested 2 student accessible solutions HDL

Designer [5] and Active-HDL [6]. Both of these tools support block diagram viasualization.

The last area which we have analyzed was the open source sector. Generally, there were some tools like Symbolator [11] which were working really well, were doing what they proposed, but the functionality itself was not enough. Next, these solutions are very technology dependent and hard to install. Symbolator is a component diagramming tool for VHDL and Verilog. The tool analyzes (and parses) the source codes of the languages, searches for and extracts components or modules and plots them as a picture. Supported raster formats include PNG and vector formats include SVG. Symbolator is implemented in Python and it needs Pycairo and Pango libraries to run.

4 Web Integrated Development Environment implementation

We have already described some base information, processes and formats which supported our visualization and now it is time to present the prototype application, which delivers the upper-mentioned functionality.

4.1 System architecture and core processes

Our web IDE is based on a web application priciple which is furthermore extended by microservices architecture. System architecture is demonstrated in **Fig. 5**. Regarding the used technologies, we used Spring Boot framework [12] and Maven [13] software project management tool for backend. Application frontend is implemented using Angular [14] framework. Backend and frontend communication are delivered through REST API calls, which are defined using the Swagger tool [15]. This means that our services fulfill the OpenApi specification.

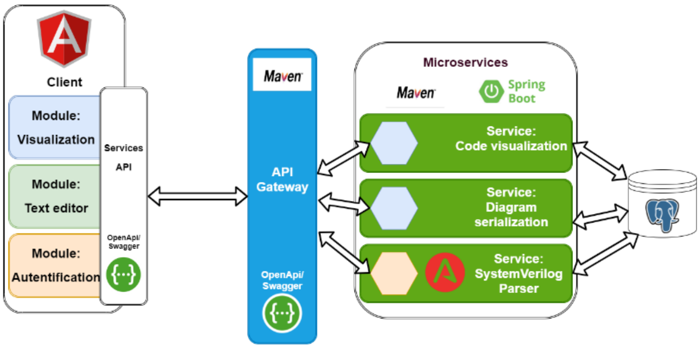


Fig. 5. Diagram representing the architecture of the system. Functionally the diagram is focused on the visualization module.

In terms of functionality, there are three functional modules worth mentioning. We will separate these modules in logical wholes according to the direction of visualization (diagram to code or code to diagram).

The first is the core diagram visualization functionality module. As we mentioned before, core diagram rendering is delivered using library JointJS[9] integrated with angular framework on the frontend of the application. This module is shared between the two logical directions. Before we generate source code, we draw the diagram in this module. On the other hand, when we parse the source code and need to visualize it, we draw the diagram using this module.

Next is the code generation module. It uses our completely custom algorithm, which passes through given diagram using the JSON representation and generates SystemVerilog source code. For now, the logic of this functionality is implemented directly on the frontend because the algorithm itself is not very resource-intensive and by bypassing the API calls we lighten the server.

For code visualization, that means the second direction of visualization, we again start on frontend of the application with source code. This source code is delivered to SystemVerilog parser module using services. In this module, the source code is parsed, using Antlr [16]. Important information are extracted and then delivered to the code visualization module. This module enhances the extracted data with spatial information and delivers them back on the frontend and to the upper-mentioned diagram visualization module.

4.2 System user interface

At the moment, we are having a low fidelity user interface that allows us to test the developed functionality on the frontend of the prototype system (**Fig. 6.**).

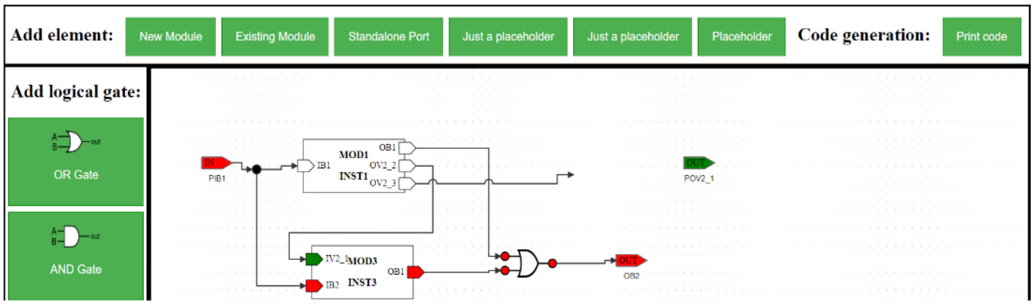


Fig. 6. In progress low fidelity user interface of prototype system.

4.3 Example functionality

For the propagation of the system functionality lets design a simple scenario. The user of our system designed a diagram which we can see in **Fig. 7**.

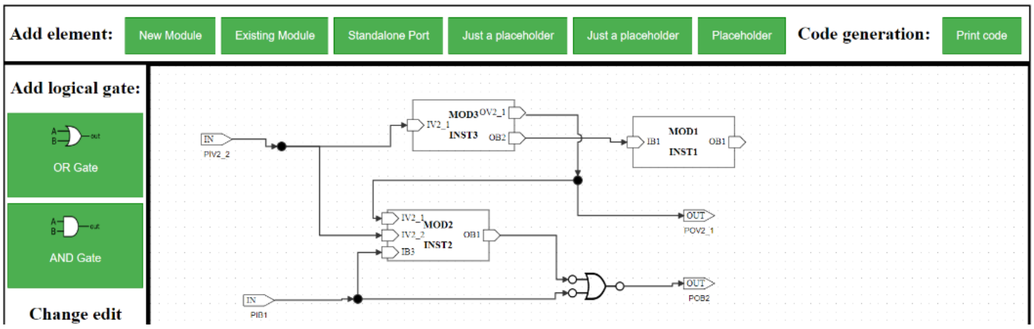


Fig. 7. Representation of system functionality using simple scenario. The diagram showed on this picture was drawn only by our prototype.

After pressing the "Print Code" button, the text in the makeshift window is displayed, showing the generated SystemVerilog code skeleton describing its diagram. Based on the output (**Fig. 8.**), you can see that the test scenario will run as expected and correct formatting will be displayed when code is generated.

```
module TestModule (
  input logic        PIB1,
  input logic [2:0]  PIV2_2,
  output logic [2:0]   POV2_1,
  output logic         POB2
);

logic         INST3_OB2_to_INST1_IB1;
logic [2:0]  INST3_OV2_1;

MOD1 INST1 (
  .IB1    (INST3_OB2_to_INST1_IB1),
  .OB1    ()
);

MOD2 INST2 (
  .IV2_1    (INST3_OV2_1),
  .IV2_2    (PIV2_2),
  .IB3      (PIB1),
  .OB1      (INST2_OB1)
);

MOD3 INST3 (
  .IV2_1    (PIV2_2),
  .OV2_1    (INST3_OV2_1),
  .OB2      (INST3_OB2_to_INST1_IB1)
);

assign POV2_1  = INST3_OV2_1;
assign POB2    = PIB1 | INST2_OB1;

endmodule: TestModule
```

Fig. 8. Generated source code based on the **Fig. 7.** block diagram.

5 Conclusion and further work

In our present work, we achieved to learn the essential processes on how to perform the visualization of SystemVerilog source code designs and also completed an in progress prototype application. This application, for now, stays not so refined and sophisticated as the upper-mentioned commercial systems, but in our opinion, it has a big potential and since it is made as a group of multiple bachelor/master thesis, there is a possibility that other students will enhance and complete this goal even after us.

Right now in the field of visualization, we are at the point when we can generate code from everything that can be drawn in our application. Specifically, we can visualize and generate to code: modules, ports and connections. We also started including some special elements that make the designs simpler. For example, we work with duplicator of connections and also started to generate and visualize some logical gates. To delineate the future work in this domain, we have already documented a specification of all elemenets we could visualize and generate, including some complex logical gates (e.g. multiplexer or encoder). But to be honest, this work theoretically never ends, because we can always come up with some new inovative graphical element that we need to visualize and then convert to code.

Considering the other direction, code to diagram, now we are at the stage of succesfully parsing and extracting important data from source code and we need to come up with some mechanism on how to enhance the plain code data with dimensional informations required to create the desired diagram. On the other hand, we already can make use of our parsing, because when an user wants to visualize a single existing module, this functionality is everything we need in order to accomplish it. At the moment, we can parse parent module's interface and its submodule declarations, identified submodules interfaces, existing signals (connections) and we can also parse the section of declared submodules outgoing ports signals (**Fig. 8**. lines 8 and 9) more precisely, because these give us better idea about the bandwidth of the connection and its relation with connected input ports.

Next, we have already mentioned the aim to achieve visualization of multiple layers of abstraction by showing the diagram of the visualized module by clicking on the module. We managed to create a serializing format for the diagrams-but now, we need to decide and create an optimal storing process of these diagrams. And last but not least, it would be right to implement better and automated way of testing our solution.

More distant improvements, which would fulfill the upper-mentioned big picture idea of our web IDE, are: extending the visualization options (e.g. State diagrams) and implementing a real-time collaborative designing and diagram completion.

References

1. Palnitkar, S.: Verilog HDL: A Guide to Digital Design and Synthesis. Prentice Hall Professional, 2003. ISBN: 0130449113.
2. Sutherland, S., Davidmann, S., Flake, P., and Moorby, P.: SystemVerilog for Design: A Guide to Using SystemVerilog for Hardware Design and Modeling. 2nd edit. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN: 0387333991.

3. Jelemenská K., Nosáľ M., Čičák P.: Visualization of Verilog Digital System Models, in book Emerging Trends in Computing, Informatics, Systems Sciences and Engineering, pages: 805-818, Springer 2012. ISBN: 978-1-4614-3557-0.
4. Macko, D., Jelemenská, K.: VHDL Structural Model Visualization, at IEEE EUROCON conference, Lisbon, Portugal, 2011, ISBN: 978-1-4244-7486-8.
5. Web page of HDL Designer tool, online, visited: 29.02.2020, link: https://www.mentor.com/products/fpga/hdl_design/hdl_designer_series/
6. Web page of Aldec's Active-HDL tool, online, visited: 29.02.2020, link: https://www.aldec.com/en/products/fpga_simulation/active-hdl
7. Willis, J. C., Peterson, G. D. and Gregor, S. L.: The advanced intermediate representation with extensibility/common environment (AIRE/CE), IEEE 1998 National Aerospace and Electronics Conference, Dayton, OH, USA, 1998, pp. 557-564.
8. Kohútka, L., Stopjaková, V.: ChipDE - A Development Environment for System Verilog-Based Digital IC Design, 2018 16th International Conference on Emerging eLearning Technologies and Applications (ICETA), Stary Smokovec, 2018, pp. 273-278.
9. Opensource diagraming library JointJS, homepage, visited: 29.02.2020, online, link: https://www.jointjs.com/opensource
10. Jelemenská, K.: Making digital systems design more convenient to students by means of visualization. In 2013 IEEE 11th International Conference on EmergingeLearning Technologies and Applications (ICETA), 2013 pages: 183–188, link: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6674426
11. Symbolator, web page, online, visited: 1.3.2020, link: https://kevinpt.github.io/symbolator/
12. What is Spring Boot, web page, visited: 29.02.2020, online, link: https://www.tutorialspoint.com/spring_boot/spring_boot_introduction.htm
13. What is Maven, web page, visited: 29.02.2020, online, link: https://maven.apache.org/what-is-maven.html
14. Angular, web page, visited: 29.02.2020, online, link: https://angular.io
15. Swagger, web page, visited: 29.02.2020, online, link: https://swagger.io/docs/specification/about/
16. Antlr, web page, visited: 29.02.2020, inline, link: https://www.antlr.org

Processing of medical data by artificial intelligence for medical diagnosis support

Matej Horniak*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
xhorniakm@stuba.sk

Abstract. One of the most reliable methods to confirm cancer diagnosis is a biopsy, which is a very difficult and lengthy process because it consists of a large amount of microscopic(histological) data. Nowadays, we are able to see the rise of automatic data processing using computer vision and deep learning. This automation provides us with a lot of benefits, such as efficient and faster data processing or more accuracy.
In this work we focus on classification of histological data using deep neural networks and namely convolutional neural networks. The main idea is to compare different approaches for initialization of kernels in the first layers of the neural networks, We compare automatic and manual initialization of kernels. We chose backpropagation, autoencoder and transfer learning for automatic initialization of kernels. And for manual kernels initialization we chose Gabor filters.

Keywords: Artificial intelligence · Machine learning · Medical data · Histological images · Gabor filters · Transfer learning · Autoencoder · Backpropagation

1 Introduction

Processing of histological data is one of the hardest techniques of diagnostics. As the result of this issue, the method is executed as the last, even though its results are the most accurate. Thanks to this examination, we can determine the correct diagnostics with the highest probability. Histological examination starts with the biopsy of the infected part of the body, as the sample is taken from tissue. In laboratory the sample is closely inspected by great magnification through the microscope, to the distance at which, every single cell can be tested for cancer.

Since this last part of the process is usually a routine, and there is no need of interaction with the patient, we believe we can automate the process, with the help of the computed science and artificial intelligence. Due to the deep

* Bachelor study programme in field: Informatics. Supervisor: doc. Ing. Vanda Benešová, PhD., Faculty of Informatics and Information Technologies STU in Bratislava

learning of neural networks, it is possible to increase the amount of data we can process and their accuracy. The only disadvantage of the method we suggest, is the requirement of huge numbers of data samples to train on.

The aim of our thesis is the presentation of numerous types of filters of convolution neuron networks. We will compare the approaches of both manual and automatic filter creation, with their main purpose being the detection of cancer in histological data

2 Related Work

2.1 Publication Transfer learning based deep CNN for segmentation and detection of mitoses in breast cancer histopathological images [4]

In the work about segmentation and detection of mitosis in breast cancer two convolutional neural networks are used. The networks are mutually connected. The segmentation uses Transfer learning, where the model from which the weights are taken is ImageNet network, that stores about thousand categories for detection of natural colored pictures.

The first convolutional neural network is used for mitosis segmentation. Patches that are extracted, further create the input for the second network, that is hybrid- convolutional in the way that in combines Weight Transfer and a custom layer for the final classification. The outcome of classification is either “mitosis” or “non-mitosis”. Thanks to these 2 phases, and the deep learning neural networks, the effect of imbalance of availability of individual categories in histological data is decreased.

Data from the dataset for TUPAC16 resulted in the accuracy of 66,67%. If the data were validated by both MITOS12 and MITOS14, result decreased to 65,1%. But if the model was trained on all the data from the beginning, the individual accuracies decreased, in the case that TUPAC16 was the validation dataset, with the accuracy of 65%. Since the validation dataset was both of the sources, the accuracy was only 63%, plus regarding only the MITOS12 and MITOS14 – 59,7%. These results are included in the table 1.

Train	Validate	F-measure
Source-A	Source-A	0.667
	Source-A and B	0.651
	Source-B	0.599
Source-A and B	Source-A	0.651
	Source-A and B	0.638
	Source-B	0.597

Table 1. Results segmentation and detection of neural networks. (source A is TUPAC16, source B is MITOS12 and MITOS14). [4]

2.2 Publication Rotation Equivariant CNNs for Digital Pathology

In this work, instead of the classical convolutional neural network, they used G-CNNs. G-CNNs are a generalization of CNNs that are equivariant under more general symmetry groups, such as the group G = p4 of 90 rotations, or G = p4m, which additionally includes reflection. In a G-CNN, the feature maps are thought of as functions on this group. For p4 and p4m, this simply means that feature channels come in groups of 4 or 8, corresponding to the 4 pure rotations in p4 or the 8 roto-reflections in p4m.

Subsequently, these layers of G-CNN create dense layers, using them to create a special architecture called DenseNet. The architecture is based on the densely connected convolutional network (DenseNet), which consist of dense blocks with layers that use the stack of all previous layers as input, alternated with transition blocks consisting of a 1x1 convolutional layer and 2x2 strided average pooling. We use one layer per dense block due to the limited receptive field of the model, with 5 dense-block/transition-block pairs. The model spatially-pools the input by a factor of 25.

Data from the PCAM dataset reached accuracy for DenseNet with P4M 89.8 and accuracy for DenseNet with P4 89.

Network	K	#W	NLL	Acc	AUC
P4M-DenseNet	64	119K	0.260	89.8	96.3
P4M-DenseNet M	24	19K	0.273	89.3	95.8
P4-DenseNet	48	125K	0.329	89.0	94.5
DenseNet+	24	128K	0.306	88.1	95.1
DenseNet+ M	64	902K	0.365	87.2	94.6
DenseNet	24	128K	0.315	87.6	95.5

Table 2. Results for PCAM dataset. [5]

3 Our work

We implemented our work the Python language, and the main part with the help of Keras
TensorFlow, Numpy, Scikit-learn and Matplotlib libraries.

3.1 Dataset

The results are gained through the PatchCamelyon (PCAM) dataset [5], that is divided into 262 144 images for learning, 32 768 images for validation and 32 768 for testing. These images are patches from the Camelyon16 dataset. This dataset consisting of colored WSI slides of sentinel lymph nodes. Slides were obtained and digitized in 2 different centers using a 40x objective.

3.2 Results

The best results were set by the unet architecture, with the accuracy of 78- 82%. The other architectures we tried are VGG 16, with only 50%, and ResNet with the depth of 8, that resulted in 67.2% accuracy. We also tried networks, with maximum of 5 convolutional layers. The outcome can be found in the table 3.

Count of layers	Kernel size	Accuracy
3 conv	2x3 1x5	74,9%
3 conv	2x3 1x5	73%
3 conv without dropout	2x3 1x5	72,29%
5 conv	2x5 3x3	63%
2 conv	2x3	74,6%

Table 3. Results for CNN with maximum 5 layers.

The best solution to our classification problem seems to Unet architecture, that contains 9 convolutional layers. It was taken as the base for further examination and comparison. The following models are based on Unet with the layer of continuously changing approaches of filter creation. 4 types of filters are used on the first layer 1:

1. Backpropagation, created as the regular convolutional layer for Unet arch. that consists of Conv2D, Batch Normalization, Activation function, with the hidden second layer of Conv2D, Batch Normalization, Activation function. After the combination we add MaxPooling2D with the size of 2 and dropout layer for decrease of overlearning. On picture number 4 you can see kernel vizualizetion.
2. Autoencoder, that consists of Conv2D, Dropout, Conv2D, MaxPooling2D with the size of 2 and Conv2D, Dropout, Conv2D, with the last layer merged by Concatenate and UpSampling2D with the size 2. On picture number 3 you can see kernel vizualizetion.
3. Gabor filters, created by the prohibition of weights learning, followed up by creation of Gabor filters with the help of OpenCV library, that is later inserted into Conv2D layer as the weights. On picture number 2 you can see kernel vizualizetion.
4. Transfer learning, we use pretrained model, from Keras library as VGG16 and ResNet 152, after that we connect this model to our Unet Architecture.

We use activation function and we add padding type “same”. We come to following results, with the restricted parameters of Backpropagation, that included 16 filters with the size of 16 – 80% accuracy, Autoencoder reached 80,524% accuracy with 32 filters with the size of 3, and accuracy of 80,47% for Gabor filers with 32 filters with the size of 7. Transfer learning reached % accuracy with ResNet and with VGG reached % accuracy. Extensive results with multiple parameters can be seen in Table number 4.

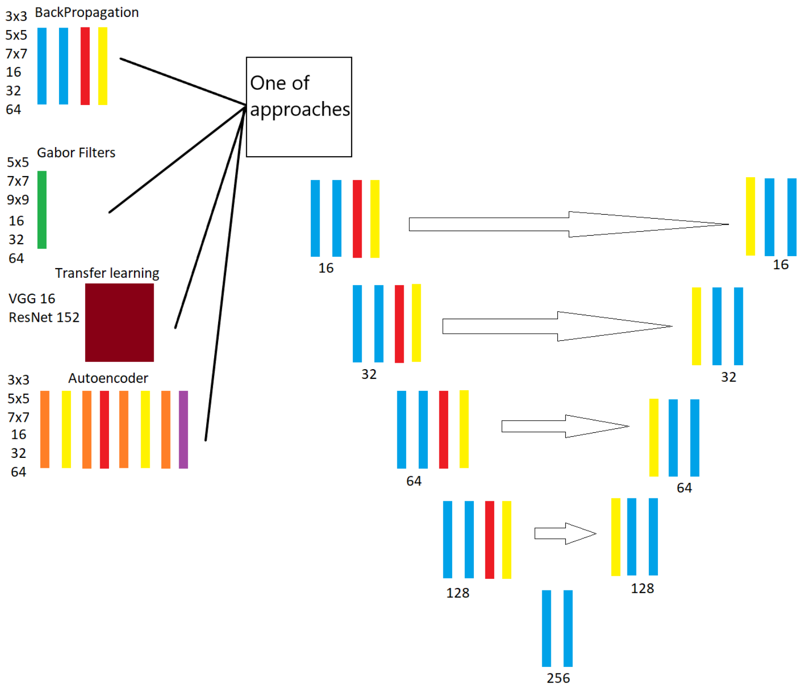


Fig. 1. Unet Achitecture with additional first layer (different approaches).

CNN approaches	Kernel size	Accuracy
Backpropagation	3x3 5x5 7x7	0.81387 0.83279 0.78588
Autoencoder	3x3 5x5 7x7	0.70355 0.70458 0.61163
Gabor filters	9x9 7x7 5x5	0.81155 0.83612 0.77017
TransferLearning ResNet VGG16		 0.58160 0.65578

Table 4. Extensive results with multiple parameters.

4 Conclusion and future work

Classification of histological data is very difficult process, because of restricted amount of data for learning, complexity of differentiation, where even the human has of finding the cancer cells, or the time consumption of the learning. Models

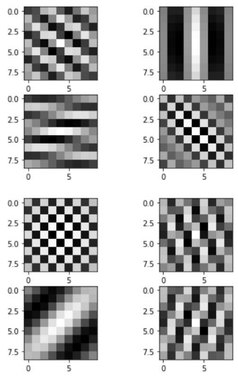


Fig. 2. Kernels vizualization from Gabor layer.

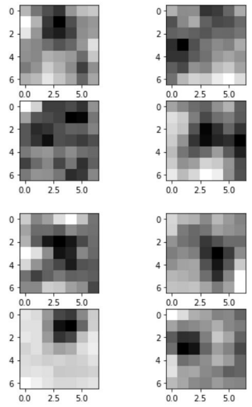


Fig. 3. Kernels vizualization from Autoencoder layer.

and comparisons, that we created and tested, resulted in different findings which are; The architecture of even small models, created by low number of layers, are very effective, sometimes ever more than the complex versions. Regarding the individual types of filters in first layers, the results were similar, with the differences in decimal numbers.

Therefore we would like to spread the scope of our observations, and to find out advantages and disadvantages of individual approaches. For this comparison we will use a synthetic dataset of textured surfaces, where the neural network should easier learn to decide between positive and negative classes. Further we will try to find out, which principle of the 4 approaches (Backpropagation, Autoencoder, Gabor Filters, Transfer learning) is the best. The observation is going to consist of minimum 11 neural networks in each principle, where we are going to change only the parameters of first layers that are; number of filters (16, 32, 64) and their size – Backpropagation and autoencoder (3, 5, 7). Regarding

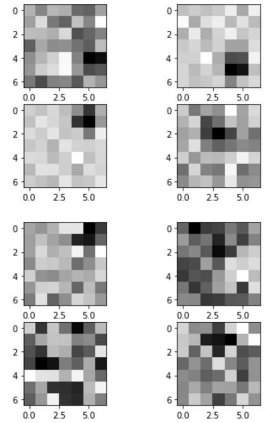


Fig. 4. Kernels vizualization from Backpropagation layer.

Gabor filters we will choose the sizes of 5, 7 and 9. As of transfer learning, we are using already created filters that we are not willing to change, so only their number that network can learn will change. Overall results will be widened by multiple metrics, such as precision, recall, f1 score and true positives. We only take the results from a test set, a data network has never seen before.

References

1. Goodfellow, I. and Bengio, Y. and Courville, A.: Deep Learning, MIT Press, 167-523 (2016)
2. Khan. S. U. and Islam, N. and Jan, Zahoor and Din, I. U. and Rodrigues, J. J. P. C: A novel deep learning based framework for the detection and classification of breast cancer using transfer learning. Pattern Recognition Letters
3. Araújo, T. and Aresta, G. and Castro, E, and Rouco, J. and Aguiar, P. and Eloy, c. and Polónia, A. and Campilho, A.: Classification of breast cancer histology images using convolutional neural networks. PloS one, (2017)
4. Wahab, N. and Khan, A. and Lee, Y. S.: Transfer learning based deep CNN for segmentation and detection of mitoses in breast cancer histopathological images, Microscopy, 216-233 (2019)
5. Veeling, Bastiaan S and Linmans, Jasper and Winkens, Jim and Cohen, Taco and Welling, Max: Rotation Equivariant CNNs for Digital Pathology, (2018)

Volumetric medical data processing with methods of artificial intelligence to support medical diagnostics

Ján Krčmář*

Slovak University of Technology, Bratislava, Slovakia
krcmar.jan@gmail.com

Abstract. Accurate segmentation of the brain tumor is essential when diagnosing, planning surgery, or monitoring disease progression. However, this time-consuming work still depends on the operator's experience and his/her subjective decisions, which are difficult to repeat. Although automatic segmentation will not completely replace a radiologist for a long time, its results can provide a valuable second opinion without requiring additional (human) resources. The underlying trend in image segmentation is represented by deep neural networks, whose usability is radically increasing with the ever-improving hardware and expanding data sets.

In this paper, we will present and compare the capabilities of a U-Net architecture introduced in 2015. Despite the trend of modifying existing architectures, a thoroughly trained U-Net achieves competitive results even today, as proven in 2018, when U-Net won the second place in Brain Tumor Segmentation Challenge (BraTS).

Keywords: CNN · U-Net · Brain tumor segmentation

1 Introduction

Primary malignant brain tumor is one of the most dreadful types of cancer, for the low survival rates it accompanies and its direct impact on patients cognitive functions, physical health or perception. Around 80% of all cancerous brain tumors are accounted by gliomas. Even with current diagnostic and treatment technologies the overall survival rates of patients with high-grade glioma (HGG) are very low. The average survival time lies between 12 and 18 month, where only 25% of the patients survive longer than one year and only 5% live longer than five years. [6] Patients with low-grade glioma (LGG) do not have much better life expectations, with an overall 10-year survival rate of 57%. Even non malignant cancers can be life threatening if they interfere with parts of brain responsible for vital body functions. [2]

* Bachelor study programme in field: Informatics, Supervisor: Assoc. Professor Vanda Benešová, Institute of Informatics, Information Systems and Software Engineering, Faculty of Informatics and Information Technologies STU in Bratislava

Due to the apparent severity of this decease, a significant number of computational algorithms have been proposed to address the complicated tumor segmentation task. To accelerate these efforts the BraTS Challenge provides a large dataset of labeled HGG and LGG cases, where for each patient there are T1 weighted, post-contrast T1 weighted, T2 weighted and Fluid-Attenuated Inversion Recovery (FLAIR) MRI scans available. Tumors are segmented into four regions: edema, necrosis, non-enhancing tumor and enhancing tumor.

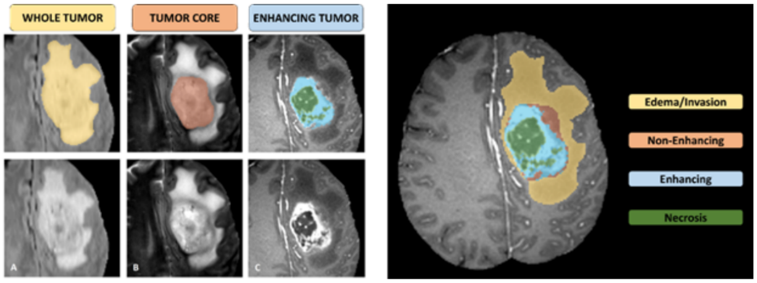


Fig. 1. Segmented tumor.[1]

2 Our proposed solution

As the base of this work we are going to use the previously mentioned U-Net architecture [5], while following some of the procedures used by the No New-Net team from BraTS 2018 [4]. Caused by the limited hardware resources, our network will only work with 2D slices, however the emphasis on proper training and evaluating different strategies will not be neglected.

2.1 Preprocessing

During the preprocessing stage we use a standard normalization (subtracting the mean and dividing by the standard deviation) of the brain region in the image. The rest of the image is set to zero which ensures that the values inside of the brain region will not be influenced by the size of the background. The normalization is necessary because the BraTS dataset is made of data coming from different organizations that use different machines and protocols to enquire them. In order to reduce the overfitting of the network, we use a number of data augmentation methods such as random rotations, scaling and gamma correction. Both normalization and augmentation is done while training or testing the network, since it does not cause any significant delay (at least for our purposes).

2.2 Segmentation using U-Net architecture

U-Net, a convolutional neural network for biomedical image segmentation, proposed by Prof. Olaf Ronnenberg et al. in 2015 [5], consists of a contracting and an expanding path.

The function of U-Net's first, contracting or downsampling path is to capture the context of an image by learning its features. It consists of four steps made of two convolutional layers followed by a max pooling layer. Each convolutional layer uses a 3x3 filter with no padding which slightly shrinks the dimensions of an image and extracts part of its features. The output of each convolutional layer is defined by the ReLU activation function. At the end of every downsampling step a max pooling layer with stride 2 is used to halve the image and to pass it to the following step with its number of activation maps doubled.

At the bottleneck, the network reaches the highest number of feature maps and leaps to the expanding path.

During the expanding or upsampling path, skip connections provide local information from the downsampling layers that is used together with the newly recovered spatial information. [5] In the upsampling path two convolution operations with a ReLU activation are followed by an upsampling layer that doubles the images resolution.

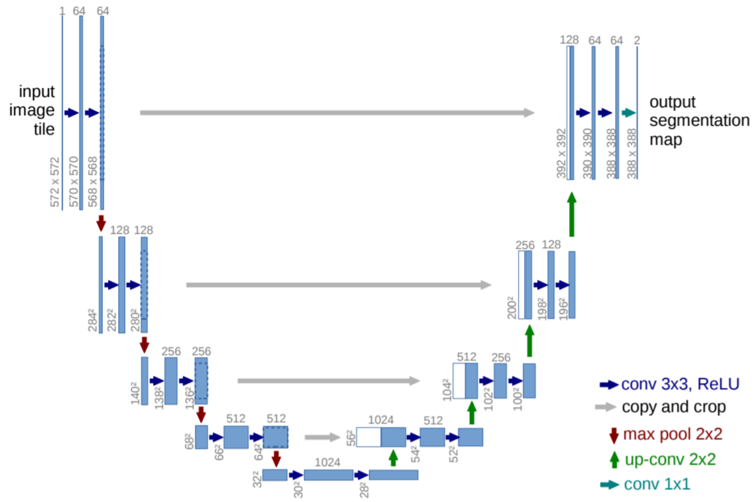


Fig. 2. U-Net architecture. The dotted blue rectangles (in the contracting path) represent feature maps, white rectangles (in the expansive path) represent copied feature maps. [5]

2.3 Training

Our network's input comprises of 4 channels, each representing the corresponding slice from a MRI sequence included. The training will go on for a maximum of 50 epochs, with early stopping and best weight restoration set when the validation loss does not change at least for 0.5 during 10 epochs. The model is compiled with the categorical crossentropy loss function and Adam optimizer.

3 Future work

Our proposal for a future work is to train a set of U-Nets, each designed to segment a different part of tumor. Such ensemble would be later compared to the fist model trained to locate and segment the whole tumor on its own. An advantage of this proposal is the possibility to chain the networks, where every following network can use and focus solemnly on the region marked by its predecessor. Any plans to modify the architecture of U-Net itself were discouraged by the [3].

4 Conclusion

In this paper, we took a look at the bald approach of [4], that motivated us (alongside with other benefits of the architecture design) to build and train a U-Net network. This architecture was able to produce fair segmentation even after first deficient training. We are looking forward to improving the training and experimenting with an ensemble of expert networks.

References

[1] Spyridon Bakas et al. "Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the BRATS challenge". In: *arXiv preprint arXiv:1811.02629* (2018).

[2] braintumor.org. *Quick brain tumor facts.* URL: https://braintumor.org/brain-tumor-information/brain-tumor-facts/.

[3] Fabian Isensee and Klaus H Maier-Hein. "An attempt at beating the 3D U-Net". In: *arXiv preprint arXiv:1908.02182* (2019).

[4] Fabian Isensee et al. "No new-net". In: *International MICCAI Brainlesion Workshop*. Springer. 2018, pp. 234–244.

[5] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention.* Springer. 2015, pp. 234–241.

[6] thebraintumourcharity. *Glioblastoma prognosis.* URL: https://www.thebraintumourcharity.org/brain-tumour-diagnosis-treatment/types-of-brain-tumour-adult/glioblastoma/glioblastoma-prognosis/.