IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 10, 2021, accepted October 3, 2021, date of publication October 21, 2021, date of current version October 27, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3121998

Network Anomaly Detection With Temporal Convolutional Network and U-Net Model

ANZHELIKA MEZINA[1], RADIM BURGET[1], AND CARLOS M. TRAVIESO-GONZÁLEZ[2]

[1]Department of Telecommunications, Faculty of Electrical Engineering and Communication (FEEC), Brno University of Technology, 616 00 Brno, Czech Republic

[2]Signals and Communications Department (DSC), Institute for Technological Development and Innovation in Communications (IDeTIC), University of Las Palmas de Gran Canaria (ULPGC), 35001 Las Palmas de Gran Canaria, Spain

Corresponding author: Anzhelika Mezina (xmezin00@vutbr.cz)

ABSTRACT Anomaly detection in network traffic is one of the key techniques to ensure security in future networks. Today, the importance of this topic is even higher, since the network traffic is growing and there is a need to have smart algorithms, which can automatically adapt to new network conditions, detect threats and recognize the type of the possible network attack. Nowadays, there are a lot of different approaches, some of them have reached relatively sufficient accuracy. However, the majority of works are being tested on old datasets, which do not reflect current network conditions and it leads to overfitted results. This is caused by high redundancy of the data and because they fail to reflect the performance of the latest methods in the real-world anomaly detection applications. In this work, we applied a couple of new methods based on convolutional neural networks: U-Net based and Temporal convolutional network based for network attack classification. We trained and evaluated methods on the old dataset KDD99 and the modern large-scale one CSE-CIC-IDS2018. According to results, Temporal convolutional network with LSTM has achieved accuracy 92% and 97% on the KDD99 and the CSE-CIC-IDS2018 respectively, the U-Net model has accuracy 93% and 94% on the KDD99 and the CSE-CIC-IDS2018 respectively. Additionally, we utilized the focal loss function in the Temporal convolutional network with Long Short-Term Memory model, which has positive effect on class imbalance in time-series data. We showed, that the Temporal convolutional network in combination with Long Short-Term Memory network and U-Net model can give higher accuracy compared to other network architectures for network traffic classification. In this work we also proved, that methods trained on the old dataset can easily overfit during training and achieve relatively good results on the testing set, but at the same time, these methods are not so successful on more complex and actual data.

INDEX TERMS Convolutional neural network, deep learning, intrusion detection system, multi-class classification, security, imbalanced dataset.

I. INTRODUCTION

Information technologies are rapidly spreading throughout the world. Unfortunately, together with this, also the number of cyber-attacks grows. Information security is critical for ensuring the safety of data in networks, identification of malicious communication and protection of their users. It is a quite challenging task since the attacks differ significantly and new kinds of attacks never been before appear each month. This is the reason why classical supervised machine learning methods like classification or regression often fail. In coming years, it is expected that due to the growing network traffic and increased number of attacks, the importance of attack detection methods will grow. Qualified estimates predict global cyber-security market growth between 2020 to 2026 by approximately 14% each year [1].

Intrusion Detection Systems (IDS) are commonly used to monitor the traffic and detect possible threats in network systems. A possible position of the IDS in the network architecture is shown in fig. 1. As can be seen from the scheme, the IDS is placed after the firewall to monitor Internet traffic further. Although the firewall first filters the traffic, it can never filter every malicious traffic. There is still a significant risk of some abnormal connections by which the possible attacker can cause damages in a local network. Unfortunately, it is impossible to predict all possible attack scenarios and to define them in an internal firewall rules manually, because the attack types differ significantly. For example, the attacker

The associate editor coordinating the review of this manuscript and approving it for publication was Yuan Tian.

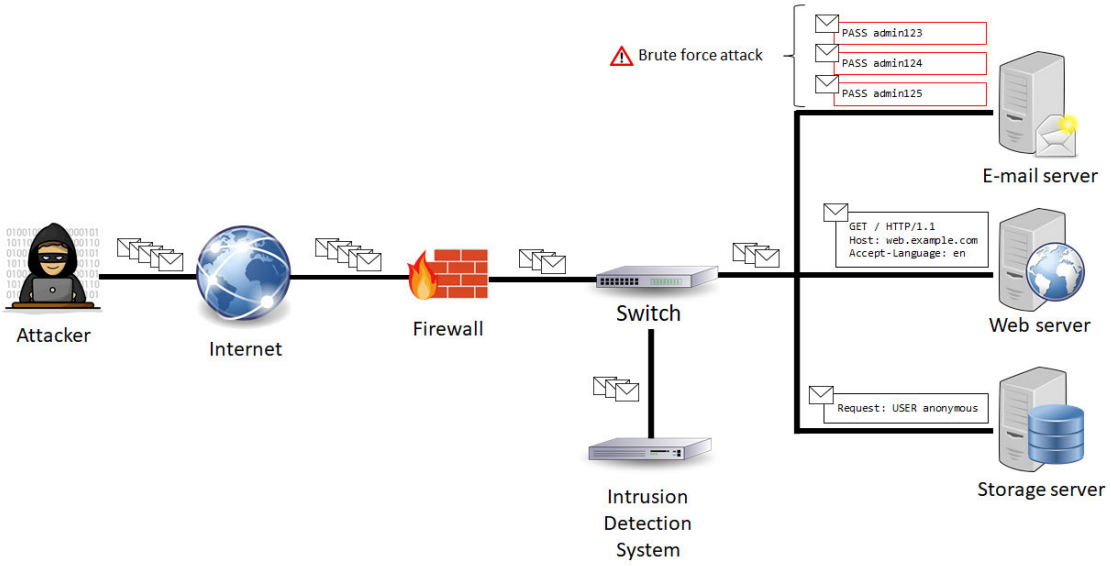


FIGURE 1. Overview of network with IDS.

can perform a brute force attack against POP3 to guess the password, if an administrator has not set up the security system properly.

Another effective method for attack detection is focused on detection of unusual behaviour in the network. For this purpose an existing signature database is being used. However, it is costly to keep it up-to-date with the latest attacks, but there will always be some delay. Because of the growing amount of network traffic and increasing value of data transmitted over the network, it is essential to have systems, which adapt to a concrete network and detect the suspicious behaviour, even without preexisting signature database of known attacks.

This is the reason why there is a big demand for technologies that can process massive volumes of data and detect anomalies in them. In particular, more attention is given to supervised learning: adaptive methods based on machine learning approaches. There are many works in this field and many of them also report relatively high accuracy. However, the main problem is lack of data in general, and many works refer to relatively old datasets, which unfortunately don't reflect the characteristics of network communication in 2021. Consequently, the performance is significantly lower, when models are trained on old datasets and deployed on new networks. The best known and frequently used datasets are KDD99[1] or NSL-KDD [2]. These datasets are more than 20 years old. Even if the intrusion detection systems prove their performance on those data, in real scenarios they can fail. That is why methods trained on these data can fail when deployed to real-world networks.

Recently, a relatively new dataset CSE-CIC-IDS2018 [3] was released. This dataset contains network communication data with traffic adapted to current network style and current attack types. The dataset contains a various range of different attacks which are labelled. It allows not only to detect but also to recognize different types of attacks (multi-class classification). Moreover, this dataset contains data from real network traffic. That fact allows to evaluate the capability of algorithms to work in real network communication.

In this work we did the study on current state of the problem of network anomaly detection and provided an overview regarding latest works. We selected, tested and compared four classifiers on KDD99 and CSE-CIC-IDS2018: convolutional neural network, autoencoder, fully connected network and recurrent neural network. We examined these two datasets to determine if the old dataset is still actual for this field of research and how the results of training and testing will differ. Moreover, we proposed two models of multi-class classification for the problem of anomaly detection: the first one is based on U-Net, which was adapted for processing time-series to perform a classification task and the second one is based on Temporal Convolutional Network and Long Short-Term Memory, which is also adapted for classification tasks. Moreover, taking into account the imbalance problem of datasets, we utilized the focal loss function, which is able to deal with imbalanced datasets.

Main contributions of this paper can be summed up in the next points. First of all, we designed a new architecture which provides more accurate network attacks classification. This architecture is based on Temporal Convolutional Network [4] and Long Short-Term Memory with focal loss function, which was not applied for this task before. The method was validated and tested on one of the largest public available datasets (CSE-CIC-IDS2018) for intrusion detection system and compared to the four classical architectures. We proved, that the proposed architecture is able to provide better results

[1] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

on time-series tasks, including anomaly detection of network attacks, and achieve accuracy 97%.

Secondly, we adapted the U-Net [5] model for anomaly detection tasks, which was not applied for these tasks before, at least at the moment of writing of this paper. It also outperforms classical architectures and achieved accuracy 94%. Additionally, we have shown, that this model is effective in prediction of attack classes, which have relatively small number of training and testing samples.

Thirdly, we compared results on two datasets, KDD99 and CSE-CIC-IDS2018 and proved, that nowadays the application of old datasets is not the best choice for evaluation of methods, because of their limitations and out-dating. The results of the evaluation of the methods on these two datasets are different, which is why it is necessary to pay attention to used data for training some models in real application.

The rest of this paper is structured as follows. Firstly, related works are described. This paper focuses on approaches based on neural networks, which are divided by the type of architecture. The next section introduces the experiment, including how the data were pre-processed together with proposed architectures, and shows the results of the experiment. The last section concludes the paper.

II. RELATED WORK

Nowadays, many different approaches exist for network anomaly detection, processing of time-series and classification of attacks. The works can be basically divided according to their functionality into methods for attack detection (i.e. binary), which classify whether the network behaviour is normal or malicious, and attack recognition (i.e. multi-class classification), which determine the concrete type of attack. Another criterion for dividing the algorithms is according to the type of training data - i.e. supervised or unsupervised. The unsupervised methods have the advantage of availability of huge amount of data and there is no need for labelled data. On the other hand, the supervised methods have better control over their evaluation and can reach higher accuracy.

The supervised methods require huge training datasets, which usually must be manually annotated by experts and updated in time. This is often very costly and also time-consuming. Their advantage is usually accuracy and ability to objectively estimate their performance. On the other hand, it is not easy to obtain large labelled datasets which contain all the possible types of attacks and keep them up-to-date. The well-known approaches are Support Vector Machine (SVM) [6], Random Forests (RF) [7], K-Nearest Neighbours (KNN) [8], Naive Bayes (NB) [9], etc.

An alternative to supervised algorithms are unsupervised methods. These methods have the advantage of being trained from data that contains no labels, which allows utilization of huge amounts of data and allows real time data update. This type of algorithms aims to find the hidden patterns in unlabelled data [10]. The common algorithms used in unsupervised learning are one-class support vector machines, k-means clustering algorithm, etc.

In recent years, deep learning methods have shown promising results in different fields, including anomaly detection. There are two main fields, which researchers investigate for intrusion detection: dimensionality reduction and classification of network traffic. The first one addresses to the problem of high dimensionality. Since the data is getting more complex and its dimensionality increases, it is more challenging to find any abnormal characteristics in a high-dimensional space. For that reason, it is required to apply dimensionality reduction. For example, in work [11] the authors proposed a method, which reduces the number of features from 81 to 10 on the CIC-IDS2017 dataset and has achieved high accuracy for binary and multi-class classification. Sometimes, the authors combine these two field of research to achieve better results. In [12] the authors proposed an approach, which based on a novel dimensionality reduction, which combines feature selection and extraction techniques as the first step, and the second step is an application of different classifiers on extracted features. This approach was tested on such datasets as ISCX 2012, NSL-KDD, and Kyoto 2006+ and has achieved accuracy 98–99%.

The main focus of this work is the problem of attack classification: more detailed description of existing deep learning approaches is introduced below and the summary of them is shown in table 1. For some methods, accuracy is not provided in the original paper, that is why some values are missing.

TABLE 1. Summary of recent approaches.

Paper	Year	Method	Dataset	Accuracy, %
[13]	2020	Autoencoder	CSE-CIC-IDS2018 CIC-IDS2017	96.20 (averaged) 96.02 (averaged)
[14]	2019	Autoencoder	KDD99 UNSW-NB15	99.996 89.134
[15]	2020	CNN	KDD99 CSE-CIC-IDS2018	99.98 91.5
[16]	2020	CNN	NSL-KDD	88.82
[17]	2019	CNN	CSE-CIC-IDS2018	–
[19]	2019	Residual learning	NSL-KDD	87.28
[20]	2020	Residual learning	NSL-KDD UNSW-NB15	99.21 86.64
[21]	2019	RNN	NSL-KDD ISCX	92 97.5
[22]	2019	RNN	CIC-IDS2017 NSL-KDD	99.1 99.4
[27]	2019	RL	NSL-KDD AWID	80.16 95.90
[28]	2020	RL	NSL-KDD AWID	89.78 95.70
[23]	2019	GAN	SWaT WADI KDD99	– – –
[24]	2019	GAN	KDD99	–

A. AUTOENCODERS

One of the recent works which utilized and autoencoder is described in [13]. The authors proposed new approaches for anomaly detection. The first one is based on training a model on non-malicious traffic and detection of anomalous and normal inputs based on suitable threshold reconstruction error value. The second approach is based on double-loop learning, which focuses on the minimization of the number of false

positives and reduction of the number of false negatives. The authors made a conclusion, that the application of the double learning approach can lead to improvement in performance: the detection time for a single flow is 1 microsecond. The main disadvantage of this work is a lack of comparison with others.

Another encoder-based approach, called two-stage deep learning (TSDL) [14], was introduced in 2019. The authors process network traffic in two steps. The first one is used to classify between the normal and abnormal states of network communication. The second step is used to determine also the class of attack, with an input vector of features together with the output of the first step. This method allows to classify various types of attacks efficiently. However, the authors achieved accuracy 99.98% for the KDD99 and only 89.13% for the UNSW-NB15. A possible reason for that difference in accuracy values is overfitting of the model, when it was trained by the KDD99. This dataset suffers from redundancy of samples. However, dataset UNSW-NB15 provides more realistic validation, because it is more complex when compared to KDD99.

B. CONVOLUTIONAL NEURAL NETWORK (CNN)

Convolutional neural networks are often used in image processing and have been defined to solve image tasks. They have a special layer - convolutional layer, which allows to extract the important features from the input data. CNN was also applied to anomaly detection, for example, in work [15]. The authors transformed a feature vector into an image and applied a 2D CNN on it. It allows to utilize the effectiveness of CNN in image processing. The authors trained and tested their proposed model on the KDD99 and the CSE-CIC-IDS2018. In spite of high accuracy on KDD99, the accuracy on testing on the CSE-CIC-IDS2018 is 91.5%. An interesting point in this work is the accuracy of benign traffic is only 73.5%, but samples for benign label take the biggest part of the dataset. In spite of that, the proposed method is successful for DoS attack detection. The similar way of data processing was used in [16] and [17]. In work [16] CNN was applied for packet preprocessing and the performance was improved by more than 10%.

In another work, the authors experimented with different depths of the CNN network to see the impact on performance. They used 1D convolutional and max-pooling layers instead of a transformation of the input vector into 2D matrix/image [18]. They proposed three architectures: shallow CNN, moderate CNN and deep CNN. According to their results for binary classification, adding more layers does not give better results.

C. RESIDUAL LEARNING

The main advantage of residual learning is the utilization of skip connections, which allow training deep neural networks without saturation of accuracy. This method found application in different tasks, including anomaly detection. For example, in approach [19] the authors proposed a new Residual Learning and split-transform-merge based CNN architectural block, which was applied in one class CNN classifier. Modelling of normal network traffic distribution is done with Stacked Autoencoders. The proposed method was evaluated on the NSL-KDD dataset and the best achieved accuracy 89.41%.

One of the latest approaches, which is called Pelican [20], also utilizes the residual learning to solve the performance degradation, since it is the usual problem in deep neural networks. Authors combined CNN and RNN in the sub residual network. It allows to effectively capture both spatial and temporal features in the input data. The proposed framework was tested on the NSL-KDD and the UNSW-NB15 and achieved 99.21% and 86.64% accuracy.

D. RECURRENT NEURAL NETWORK (RNN)

Recurrent neural networks are primarily used to process texts and time-series thanks to utilization of previous state outputs as inputs for the next state. For that reason, the RNNs are often applied in anomaly detection.

One of the recent works consists of the following steps: preprocessing of the original dataset, generation of feature subsets, measurement of a loss score and an accuracy of subsets with decision tree and selection of the best feature set, application of classifiers - RNN, Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) models and the final step is the evaluation of various IDS RNNs models [21]. As the result, thanks to feature selection, computation time and memory usage were reduced, and the GRU model achieved also a better accuracy – 92% on the NSL-KDD dataset and 97.5% on the ISCX dataset. Also, an RNN is utilized in work [22]. The authors proposed a system, which uses the misuse detection approach and the anomaly-based detection approach. This system consists of a Misuse Detection Engine, Anomaly Detection Engine and Signature Generation Engine, which filters known attacks, detects malicious packets, generates signatures and updates the signature repository of IDS respectively. Authors used the CIC-IDS2017 and NSL-KDD datasets and achieved accuracy approximately 99% for binary and multi-class classification.

E. GENERATIVE ADVERSARIAL NETWORKS (GAN)

In recent years, Generative Adversarial Networks are becoming very popular and have been applied in different fields of research, including anomaly detection. This architecture usually has two parts: generator and discriminator. The generator tries to create data, which would be very similar to real data. The input to the discriminator can be the output from the generator or the real data. The aim of the discriminator is to determine if the input is real or produced by the generator. This concept forces neural networks to be trained on larger datasets than it is available.

One of these works is MAD-GAN [23]. This model consists of a generator and a discriminator, which are two Long-Short-Term-Memory Recurrent Neural Networks (LSTM-RNN). The discriminator is trained to correctly

classify both real and fake sequences, and the generator is trained to fool the discriminator. The model was tested and compared on three datasets: SWaT, WADI and KDD99. They choose metrics such as Precision, Recall and F1 and the best values MAD-GAN achieved on the SWaT dataset. The results were 0.9999 precision, 0.9998 recall and 0.77 F1 score. On the WADI dataset with 0.4698 precision, 0.99 recall and 0.37 F1 score. On the KDD99 dataset the best results were 0.9492 precision, 0.9633 recall and 0.94 F1 score. In spite of good results, the disadvantage of this approach is the potential instability of the GANs in real production.

Another GAN-based model, Fence GAN [24], has a generator and a discriminator, which are fully connected networks. Authors modified loss functions to adapt the model for anomaly detection: encirclement and dispersion losses are used for the generator, and weighted discriminator loss is used for the discriminator. The disadvantage of modification of loss function is that the generated samples lie close to the boundaries of the real data distribution and the model does not succeed to detect anomalies near discontinious boundaries [25]. This model was tested on different datasets: MNIST, CIFAR10 and KDD99 and showed the best anomaly classification accuracy. On KDD99 it achieved results: 0.9546 accuracy, 0.9697 recall and 0.9553 F1 score.

F. REINFORCEMENT LEARNING (RL)

Reinforcement learning which stands between supervised and unsupervised learning, has succeeded in couple of works. The main goal is to map situations to actions – using maximization of a numerical reward of the signal [26]. RL is primarily used in the field of game playing. However, it also found application in anomaly detection. For example, in [27] the authors created a classifier based on RL: the environment sends information about its state to an agent, and the agent responds with action. In this framework, the agent is a classifier, which tries to predict the label from a given state of the environment. The agent and environment use the fully-connected neural network to train a model. The main advantage of this approach is that it requires less time for training and testing. The accuracy of classification is similar to the state-of-the-art and achieved 95.90% on the AWID dataset and 80.16% on the NSL-KDD dataset.

In another paper [28], the authors also applied the RL method for anomaly detection. They modified the classical concept of deep reinforcement learning by replacing the environment with a sampling function of recorded training intrusions. It allows to generate rewards based on detection errors found during training. Moreover, they compared different deep reinforcement learning models with this new technique and showed that RL can improve results and works faster than alternative models.

G. SUMMARY OF EXISTING APPROACHES

Based on the above mentioned literature research, neural network architectures were one of the most successful approaches to the problem of anomaly detection and traffic classification. It should be emphasized, that many studies use relatively old datasets for their evaluation. Those studies which evaluated their methods on both datasets showed significantly worse results on the new dataset (see table 1) which indicates, that new architectures reflecting especially new data are needed. The datasets have also some limitations which are described in more detail below. As well, not many works are addressing to the imbalance problem of datasets.

H. DATASETS

There are several public datasets, which allow to test and to evaluate algorithms for network anomaly detection or classification of network traffic. The most significant and most frequently used ones are described below.

KDD99 dataset was created in 1999 as a filtered version of the DARPA98 dataset. This dataset is still widely used in many research works as a benchmark dataset. It contains approximately 4,900,000 samples, each with 41 features. The attacks are classified into four classes: DoS (Denial of Service), U2R (User to Root), R2L (Remote to Local) and Probe (Probing Attack). Also a normal traffic category is provided [29]. In spite of the huge number of records, there are a lot of duplicates, so there are in fact approximately only 1,400,000 unique data samples for training, and testing [30]. The NSL-KDD dataset was introduced to solve the problem of the duplicated samples in the KDD99. It has only about 126,000 and 22,000 training and testing samples, respectively.

Another dataset, the ISCX-IDS-2012, contains 2,381,532 normal samples and 68,792 malicious traffic of seven days [31]. The authors prepared 4 scenarios of attacks: infiltrating the network from the inside, HTTP denial of service, distributed denial of service using an IRC Botnet and brute force SSH.

UNSW-NB15 [32] was created in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) in 2015. The dataset contains labelled samples with 49 attributes and nine types of attacks, namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. The dataset consists of four CSV files with the total of 2,540,044 labelled records.

One of the modern datasets is CIC-IDS2017 [3], which was developed by the Canadian Institute of Cyber Security. This dataset is getting attractive for researchers, who want to implement and test algorithms, for example, for the classification of attacks. This dataset includes collected data from five days and contains the following attacks: Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS. The dataset contains full packet payloads in PCAPs and processed data in CSV format for machine learning [3].

The updated version of CIC-IDS2017 was released in 2018. CSE-CIC-IDS2018 is a collaborative project between the Canada's Communications Security Establishment (CSE) and the Canadian Institute for Cybersecurity (CIC). This dataset has a class imbalance and is similar

to CIC-IDS2017. It contains about 16,000,000 instances in 10 files collected from 10 days of network traffic. Nine files have samples with 79 attributes and one file has records with 83 attributes. The full list of features can be found on the official page of the dataset. The total number of label classes is 15. They can be grouped into DDoS, DoS, Brute force, Botnet, Infiltration, Web attack and Benign [33].

Also, there are some other datasets, which are used for evaluation of anomaly detection: Aegean WiFi Intrusion Dataset (AWID), which contains real flows of normal and intrusive 802.11 traffic [34], Secure Water Treatment (SWaT) dataset [35], Water Distribution (WADI) [36], which are testbeds for security research and training.

I. LIMITATIONS OF OLD DATASETS

The main problem of the outdated datasets is that they do not represent the modern attack scenarios [37]. Since the behavioural patterns of network attacks have changed, it is necessary to have up-to-date datasets for research. This problem was already mentioned in some papers as one of the challenges, for example, in [38] and [39]. The next problem is the difference of distribution of the testing set and training set because of the appearing of new attack vectors in the testing set. Moreover, the well-known problem of old datasets, is overfitting. According to provided experiment results in paper [40], it is obvious that it is not difficult to achieve high results. Additionally, it can be seen from table 1 that methods tested on the KDD99 or the NSL-KDD achieved accuracy 99%, but on latest datasets, such as UNSW-NB15 or CIC-IDS2017, the achieved less accuracy than on the older datasets. This fact supports the necessity of utilization of modern datasets for evaluation of proposed methods and it proves, that it is necessary to improve algorithms and that there is a space for increase of accuracy.

III. METHODOLOGY

One of the goals of this work is to compare existing methods for network attacks classification and to propose new methods for this task, which perform better, especially on the newest datasets. For this purpose we proposed methods using the U-Net model and Temporal Convolutional Network. The U-Net model had been already applied for processing time series [41], however, it has not been applied for anomaly detection in networks yet. TCNs were also applied for time series and were more successful than LSTM and GRU [42]. Moreover, in contrast to the most existing methods, the modern dataset (CSE-CIC-IDS2018) is used for evaluation. Also, the KDD99 dataset was used for training and testing models to compare results with the CSE-CIC-IDS2018 to evaluate robustness of the methods with other data and compare results with other works.

A. DATA PROCESSING

The first step is data preprocessing. For this experiment we selected two datasets: KDD99 and CSE-CIC-IDS2018, which were already described in section II-H. We choose these two datasets to compare the results of training and testing on the old dataset and the modern one to see how the results will be different, since the KDD99 has limitations and the CSE-CIC-IDS2018 is more complex and corresponds to current network traffic.

KDD99 has been already divided into train and test sets and the number of training and testing samples for each kind of attacks is shown in table 2. The training set contains 4,898,431 samples, and the testing set consists of 311,029 samples. The total number of features is 41. The types of attacks were grouped into 4 sets: Denial of Service, User to Root, Remote to Local and Probing Attack. Features, like services, flags, and protocol types were also encoded with numbers since they had the dataset's string values. After that, the prepared dataset was normalized. The labels were one-hot encoded: the output of this method is a vector of "0" and "1", where the position of "1" indicates the correct label of the input sample.

TABLE 2. Used attack types from KDD99 dataset.

Label	Training set		Testing set	
	Total	Percentage, %	Total	Percentage, %
Normal	972,781	19.859	60,593	19.481
DOS	3,883,370	79.278	229,853	73.901
Probe	41,102	0.839	4,166	1.339
R2L	1,126	0.023	16,347	5.256
U2R	52	0.001	70	0.023

CSE-CIC-IDS2018 contains 10 CSV files. For this experiment only 9 of the 10 CSV files were used. They contain the total of 79 attributes, where only 78 features are applied: timestamp has been removed, since it has no impact on attack classification of network traffic. The used attack types and numbers of samples are shown in table 3. The next step is cleaning data from null and infinite values. The samples with these values were removed. Also, the data were normalized in the range 0 to 1. The one-hot encoding method is used to encode the labels. The final number of data samples for training, validation and testing is 8,247,888. The cross-validation with $k = 5$ was used for training and evaluation of the performance of the neural network models. For training (including validation) is used 90%, for testing 10% of whole the dataset. Since 5-fold cross validation is used, 20% of the training set is used for validation. The final split of the whole dataset is: training set – 72%, validation set – 18% and testing set 10%.

B. EXPERIMENTAL SETUP

For the experiment, totally 6 neural networks are used. The hyperparameters of the training have been: the number of epochs 10, the batch size 512 for training and validation steps. The scheme of the utilization of CSE-CIC-IDS2018 dataset using cross-validation is shown in fig. 2. Cross-validation is also applied for training KDD99, as it is shown in fig. 3. Since this dataset contains training and testing parts, 20% of the training dataset is used for validation in each iteration.

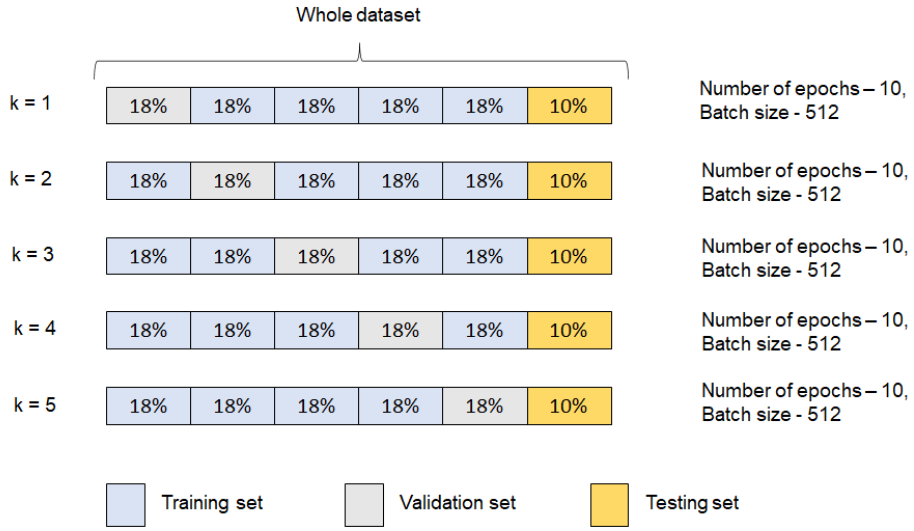


FIGURE 2. Cross-validation for CSE-CIC-IDS2018.

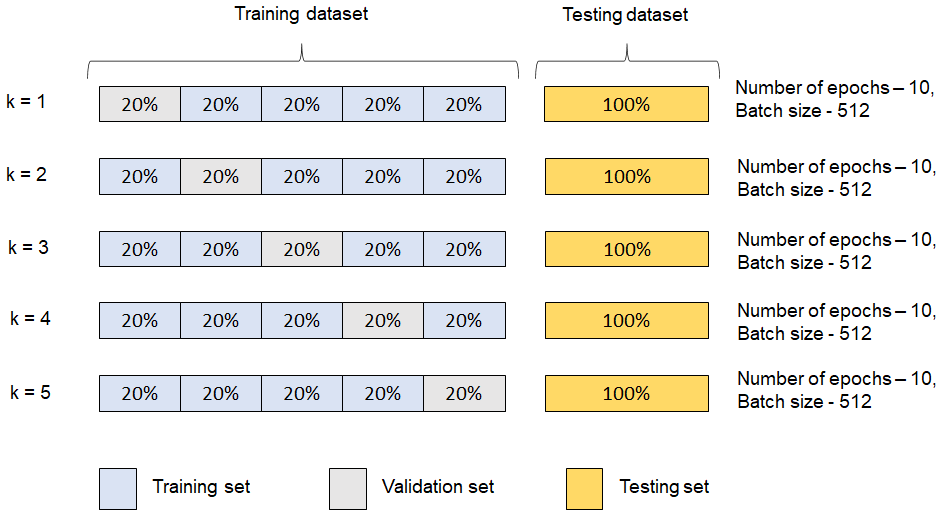


FIGURE 3. Cross-validation for KDD99.

TABLE 3. Used attack types from CSE-CIC-IDS2018 dataset.

Label	Total	Percentage, %
Benign	6,112,151	73.781
DDoS attack-HOIC	686,012	8.281
DoS attacks-Hulk	461,912	5.576
Bot	286,191	3.455
FTP-BruteForce	193,360	2.334
SSH-Bruteforce	187,589	2.264
Infiltration	161,934	1.955
DoS attacks-SlowHTTPTest	139,890	1.689
DoS attacks-GoldenEye	41,508	0.501
DoS attacks-Slowloris	10,990	0.133
DDOS attack-LOIC-UDP	1,730	0.021
Brute Force-Web	611	0.007
Brute Force-XSS	230	0.003
SQL Injection	87	0.001

The framework used for the implementation of models is Tensorflow.[2]

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

The hardware used for training is the Graphical Processing Unit (GPU) NVidia GeForce RTX 2080 Ti with CUDA acceleration technology. First of all, the existing models were trained and tested on the mentioned datasets to create a baseline, which is described below, and to compare the proposed methods.

C. BASELINE

For the experiment, several modern architectures were used: convolutional neural network, fully-connected network (FCN), autoencoder and recurrent neural network. The detailed description of them can be found in [43]. These models use categorical cross-entropy as the loss function, except RNN, which uses Mean Square Error (MSE), and Adam as the optimizer with learning rate 0.001 for CNN, FCN, RNN and 0.00001 for the autoencoder. These parameters are usually applied for processing time-series tasks and for this

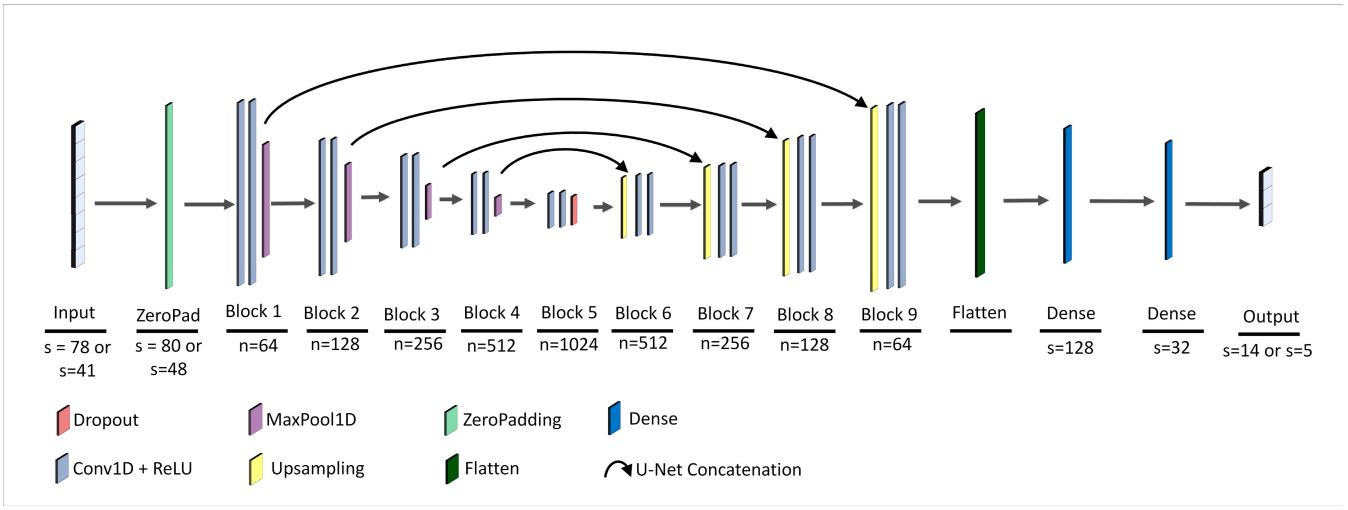


FIGURE 4. Used U-Net model.

experiment, we used the same parameters as it was proposed in original implementations. The last layer in models has the softmax activation layer since it is usually used for multi-class classification.

1) CONVOLUTIONAL NEURAL NETWORK

The used CNN contains 6 layers: two convolutional layers, two average pooling layers, the flatten layer and the fully connected layer. Convolutional layers have kernel size 7, 6 and 12 filters respectively, the activation function is sigmoid, which is formulated as [44]:

$$sigmoid = \frac{1}{1 + e^{-x}} \tag{1}$$

The average pooling layer has the pool size 3. The last layer has the number of neurons equal to the number of classes.

2) FULLY CONNECTED NETWORK

FCN consists of three blocks of layers, the global average pooling layer, and the dense layer. Each block contains a convolutional layer, a batch normalization layer and an activation function ReLU, which is defined as [44]:

$$f(x) = max(0, x) \tag{2}$$

The convolutional layer in the first block has 128 filters and kernel size 8, the second block has 256 filters and kernel size 5, the third block has 128 filters and kernel size 3. The last layer is the fully connected layer and has the same number of neurons as the number of classes.

3) AUTOENCODER

Autoencoder has three blocks and an attention mechanism. The first and second blocks have the convolutional layer with stride 1, instance normalization, layer with activation function PreLU, dropout layer with rate 0.2 and max-pooling layer with pool size 2. The convolutional layer in the first block has 128 filters and kernel size 5, in the second block it has 256 filters and kernel size 11.

The third block has a convolutional layer with stride 1, 512 filters, kernel size 21, instance normalization, layer with activation function PreLU and dropout layer with rate 0.2. After that, the attention mechanism is applied. It allows to learn which parts of input are important for a certain classification.

The last part is the softmax layer, which is a fully connected layer with number of neurons equal to the number of classes needed to be classified.

4) RECURRENT NEURAL NETWORK

RNN network has two repeated pairs of layers: recurrent layer with 64 units and dropout layer with rate 0.2. After that the fully connected layer follows with units, whose number is equal to the number of classes.

D. U-Net

In this work, the U-Net architecture is proposed for the classification of network traffic. This model is primarily used for segmentation of biomedical images [5]. It consists of two branches: encoder (left side) and decoder (right side) part, see fig. 4. The left part contains repeated blocks with convolutional layers with kernel size 3 and activation function ReLU and max pooling layers with stride 2 for downsampling. The right part consists of blocks with convolutional layers and upsampling layers which are concatenated with corresponding blocks from left side. These concatenations help to utilize the extracted features from the encoder and to improve the learning process.

We decided to do an experiment with this model, because it is able to efficiently extract general and detailed features and this model has achieved good results in other research areas. Since the data from network traffic are complex, there is a need to apply architecture, which is able to find patterns by

taking into account global and detailed features. The principle of this problem is familiar with the segmentation task, that is why successful model in segmentation problem can be adapted to a problem of processing network traffic. Another main point is that this model is able to efficiently extract features in the encoder and to connect them in the decoder part, which allows to transfer information from the encoder. Moreover, this model is easy to understand and implement, consequently, it can be easily modified and adapted for other fields of research.

In our experiment the input vector has the size 41 for the KDD99 dataset and 78 for the CSE-CIC-IDS2018, but for this architecture it is required to pad input data with some zeros to allow the model to process the data: correctly perform max pooling and upsampling operations and to perform concatenation of the left and right side of the model.

The proposed model is the modification of the original U-Net architecture. Since the input data is a vector of features, it is necessary to use layers for 1D: convolutional, max pooling, and upsampling layers. The activation function is ReLU in convolutional layers, and in the last fully connected layer it is the softmax activation function, which is usually used for multi-class classification. The softmax function is defined as below:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}, \tag{3}$$

where

- z – feature vector,
- j – index of class,
- K – number of the classes.

The kernel size of the convolutional layers is 3. The dropout layers have the rate of 0.5. The overall scheme of the architecture is shown in fig. 4, where s is the size of the layer and n is the number of feature maps.

To adapt this architecture to the classification task, fully connected layers are appended. For multi-class classification a vector of 5 for the KDD99 and for the CSE-CIC-IDS2018 a vector of 14 are the outputs. Thanks to the softmax function in the last layer, this vector has normalized values and contains the probabilities how an input vector relates to the classes.

The used optimizer is Adam with learning rate 0.0001. The batch size for training and validation is 512. The loss function is the categorical cross-entropy, which is formulated as [45]:

$$H(p, q) = -\sum p_i \log q_i \tag{4}$$

where

- p_i – true distribution,
- q_i – predicted probability distribution.

E. TEMPORAL CONVOLUTIONAL NETWORK AND LSTM

The second proposed neural network, which is evaluated for anomaly detection, is a Temporal Convolutional Network (TCN) combined with Long Short-Term-Memory (LSTM).

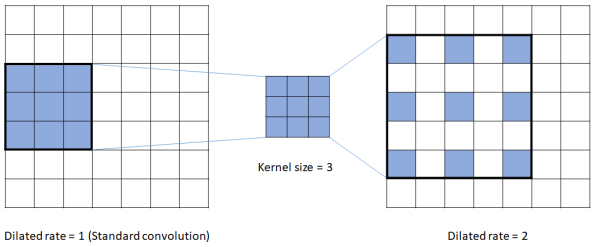


FIGURE 5. Dilated convolution operation.

There are two branches with different architectures to classify network traffic. A similar way for time series classification was already described in [46]. However, instead of the utilization of a fully connected network, the temporal convolutional network is used. The architecture is shown in fig. 6, where s is a size of the layer and n is the number of feature maps.

The first branch consists of the TCN and the FCN. Originally, TCN was proposed for action segmentation [48]. However, it is possible to utilize this model for time-series processing. This model can capture long-range patterns using a hierarchy of temporal convolutional filters. The original paper describes two types of the model: the encoder-decoder TCN, which utilizes only a hierarchy of temporal convolutions, pooling and upsampling, and the Dilated TCN, which contains skip connections and uses dilated convolutional layers instead of upsampling and pooling layers.

Additionally, the effectiveness of this model was compared with RNN as an alternative model for sequence modeling. The TCN model has some properties, which would make this model more effective [47]:

- Application of convolutional layers with the same filter allows to perform computations in parallel.
- The number of dilated convolutional layers, dilation factors and the filter size can be set up manually, that makes the TCN a very flexible model and can be adapted for different fields of applications.
- In opposite to RNNs, which store the partial results, the filters in the TCN are shared across a layers, consequently, it leads to lower memory usage.

The branch with the TCN has two blocks in the encoder and two blocks in the decoder. Each block has 4 layers: Convolutional layer with the dilated convolution, Spatial Dropout, ReLU activation and Normalization. Convolutional layers in Block 1 and Block 4 have dilated rate 1 and number of features is 64 (in fig. 6 it is shown as variable n), and in Block 2 and Block 3 the dilated rate is 2 and the number of features is 96. The dilated convolution operation allows analyzing a wider field of view, because it has "holes" between kernel elements. The principle of the dilated convolution is shown in fig. 5 and formally defined as [47]:

$$F(s) = (\mathbf{x} *_d f)(s) = \sum_{i=0}^{k-1} f(i) \cdot \mathbf{x}_{s-d \cdot i}, \tag{5}$$

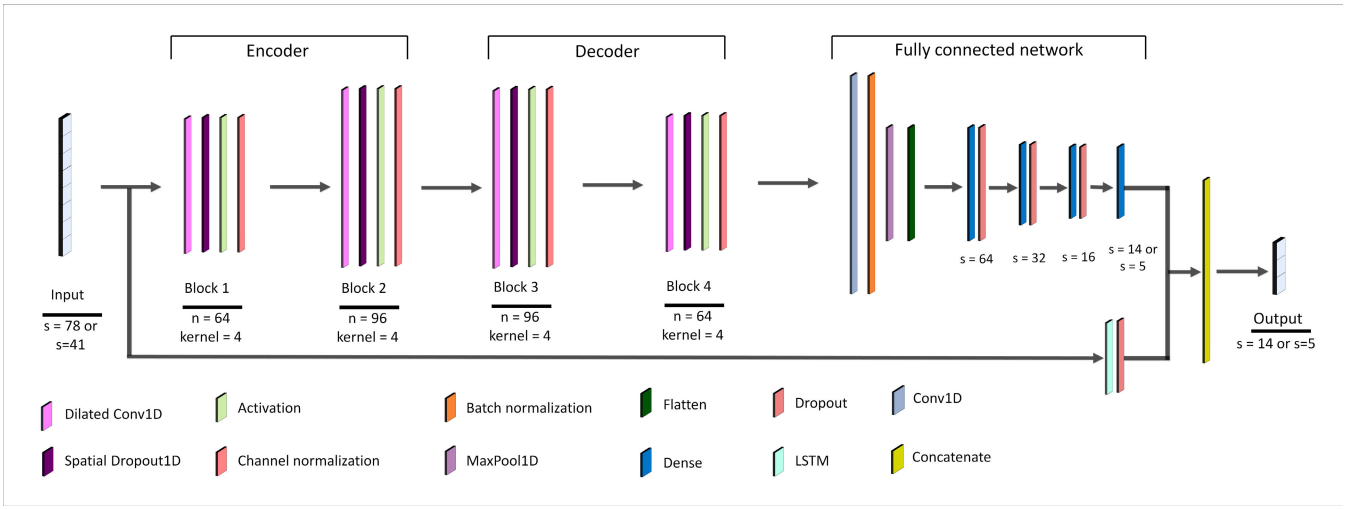


FIGURE 6. Proposed TCN+LSTM model.

where d is the dilation factor, k is the filter size and $s - d \cdot i$ are accounts for the direction of the past.

Spatial dropout layers have the rate 0.3 and perform the dropout on full convolutional filters to improve performance [48]. The channel normalization layer normalizes values by the highest number from the previous ReLU activation layer. All used parameters are left the same, as in the original implementation.

After extracting features by the TCN, the fully connected network is used for classification. It contains blocks of dense layers with the Leaky ReLU activation function and dropout layers with rate 0.5. Another branch in this architecture has only two layers: LSTM with 8 units and Dropout. This branch allows to achieve more accurate results. The final step is concatenation of outputs from those two branches and processing it with fully connected layer with softmax activation function.

The used optimizer is Adam with learning rate 0.00001. The batch size for training and validation is 512. The loss is computed by the focal loss function [49]. This loss function was originally used for object detection to solve the problem of imbalance between foreground and background classes, however, it was also used for classification tasks for the class imbalance problem [50]. The general definition of the focal loss is:

$$FL(p_t) = -\alpha_t (1 - p_t)^{\gamma} \log(p_t), \tag{6}$$

where

p_t – the model's estimated probability for the class with label $y = 1$,

α_t – balancing factor,

γ – modulating factor.

IV. RESULTS AND EVALUATION

All models were evaluated on a testing dataset of the CSE-CIC-IDS2018, which is 10% of the total number of samples, and on already prepared testing dataset of KDD99. Since the cross-validation method is used for training, the testing is done for each iteration. As a result, there are five results of testing for each method, which were averaged.

A. METRICS

The used metrics are: accuracy, precision, recall and F-score, which are usually used in evaluation of methods for anomaly detection. The definitions of mentioned metrics are shown below [51]:

Accuracy describes how the trained model is right.

$$Accuracy = \frac{TN + TP}{TP + TN + FP + FN}, \tag{7}$$

where TN – True Negatives, TP – True Positives, FP – False Positives, FN – False Negatives.

Precision is the ratio of right positive predictions over all positive predicted labels.

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

Recall is the number of right positive predictions over the total count of all relevant samples.

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

F-score is the harmonic mean of precision and recall, which is used to a measure model trained from the imbalanced dataset.

$$\text{F-score} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{10}$$

B. RESULTS

In this experiment, in total 6 different neural network models: CNN, autoencoder, FCN, RNN, U-Net and TCN+LSTM, were evaluated. To prove that the LSTM branch has a positive effect on prediction accuracy, the results of testing the TCN model without the LSTM branch are

TABLE 4. Averaged evaluation of methods for multi-class classification on KDD99.

Methods	Accuracy	Precision	Recall	F-Score	Time for one epoch	Trainable parameters
CNN	0.9174	0.9176	0.9173	0.9175	1.5 min	809
Autoencoder	0.9213	0.9215	0.9212	0.9214	2.5 min	3,196,293
FCN	0.9243	0.9244	0.9242	0.9243	1.5 min	266,373
RNN	0.9016	0.9020	0.9015	0.9018	3.5 min	12,805
U-Net (Our)	**0.9303**	**0.9305**	**0.9302**	**0.9304**	5 min	11,209,925
TCN (our)	0.9204	0.9243	0.9187	0.9215	3 min	278,149
TCN + LSTM (our)	0.9205	0.9245	0.9187	0.9216	17 min	278,539

TABLE 5. Results of testing U-Net for each class of KDD99.

Class	Precision	Recall	F-Score	Total number
Normal	0.74	1.00	0.85	60,593
Probe	0.87	0.70	0.77	4,166
DoS	1.00	0.97	0.99	229,853
U2R	0.94	0.10	0.19	70
R2L	0.99	0.15	0.25	16,347

TABLE 6. Results of testing TCN+LSTM for each class of KDD99.

Class	Precision	Recall	F-Score	Total number
Normal	0.72	0.98	0.83	60,593
Probe	0.82	0.72	0.72	4,166
DoS	1.00	0.97	0.98	229,853
U2R	0.00	0.00	0.00	70
R2L	0.00	0.00	0.00	16,347

also provided. The CSE-CIC-IDS2018 dataset and the classical dataset KDD99 were used for evaluation. The last two mentioned models were newly proposed for this task. Details about the experiment results are provided in the following subsections.

1) RESULTS FOR THE KDD99

Quantitative results of testing on the KDD99 show, that the best model in the baseline is the FCN with 0.9243 accuracy, 0.9244 precision, 0.9242 recall and 0.9243 F-Score. On the other side, the proposed method based on the U-Net outperforms FCN with 0.9303 accuracy, 0.9305 precision, 0.9302 recall and 0.9304 F-Score. In spite of reported results in other works, which achieved higher accuracy on this dataset, the new methods outperform on the CSE-CIC-IDS2018, that will be presented in the next subsection. The averaged results are shown in table 4 and detailed results for each group of attacks are presented in table 5 and table 6. According to the results, the TCN+LSTM model is able to recognize normal traffic, DoS attacks and Probing attacks. The U-Net model also succeeded in classification of these attacks. The worst results were with R2L and U2R classes: the U-Net model has low values for recall and F-score, but the TCN+LSTM model was not able to correctly classify these attacks at all. The possible reason for this is the small number of samples, which were given in the training set. In spite of that, the U-Net model outperforms the FCN classifier. Another important point is that it is time consuming to train TCN+LSTM: it takes 17 min for one epoch. It is caused by the LSTM branch, because without it the time of training of one epoch is 3 min.

2) RESULTS FOR CSE-CIC-IDS2018

According to the results of testing on the CSE-CIC-IDS2018, which are shown in table 7, the worst model is the FCN, which achieved only 0.6943 accuracy, precision is 0.6965, recall is 0.6943 and F-score is 0.6954. The best one from the baseline is the autoencoder with 0.9303 accuracy, 0.9236 recall, 0.9285 F-score and 0.9337 precision.

As it can be seen, the autoencoder is able to extract features better from given data and achieves good objective results. On the other side, it takes more time for training when compared to CNN, FCN and RNN.

The U-Net and the TCN+LSTM outperform the baseline methods. The TCN+LSTM shows the best results and has the accuracy 0.9777, so it is more accurate by 4% than the autoencoder. The precision is 0.9794, that is by 0.0457 better than the autoencoder. Recall is 0.9753, that is by 0.0517 better than autoencoder and F-score is 0.9773.

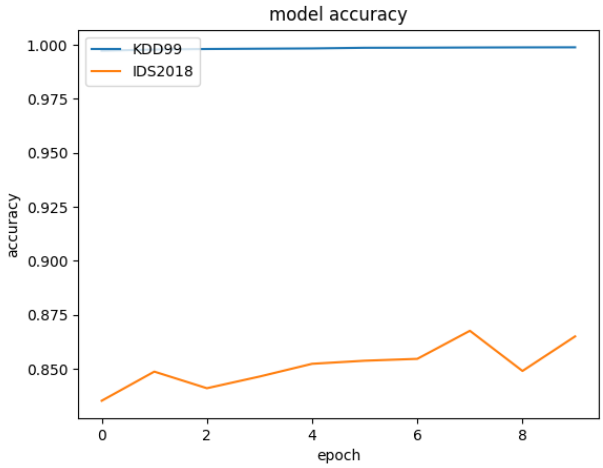
Application of the LSTM branch helps to improve metrics. The results without the LSTM branch are: accuracy 0.9761, precision 0.9789, recall 0.9725, and the training takes 5 min.

The proposed U-Net architecture also outperforms well-known architectures. However, the difference from the best baseline methods is not as big as the TCN+LSTM has: accuracy is 0.9465, precision is 0.9488, recall is 0.9455 and F-score is 0.9471.

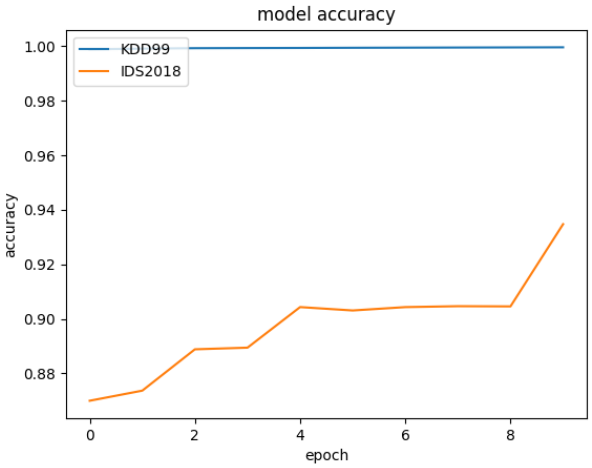
More detailed results for the proposed models are shown in table 8 and table 9. The values from each iteration of the k-folder are also averaged. According to the results, there is a problem of detection of such attacks as Brute Force - Web, Brute Force - XSS, SQL Injection and Infiltration. A possible reason for this is a small number of testing samples. However, the accuracy of detection of the other attack types is relatively high.

V. DISCUSSION

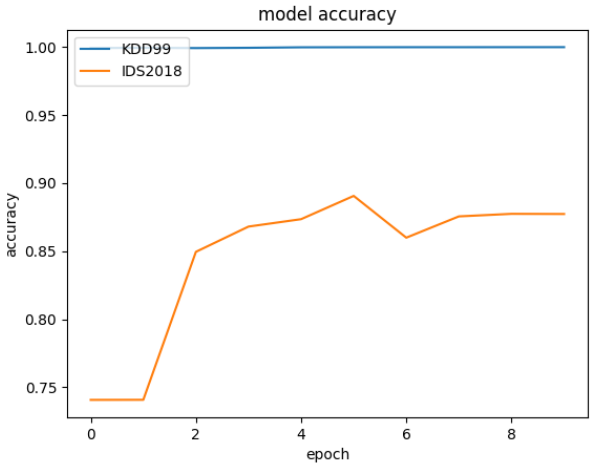
From the presented results it can be seen that the proposed models are successful in experiments on both datasets. The U-Net achieved the best results on the KDD99 dataset among the selected architectures, the TCN+LSTM is the best for the CSE-CIC-IDS2018 dataset. In spite of good results, it is time consuming to train the proposed methods: it takes 5-17 min. In the case of the U-Net model, it can be described



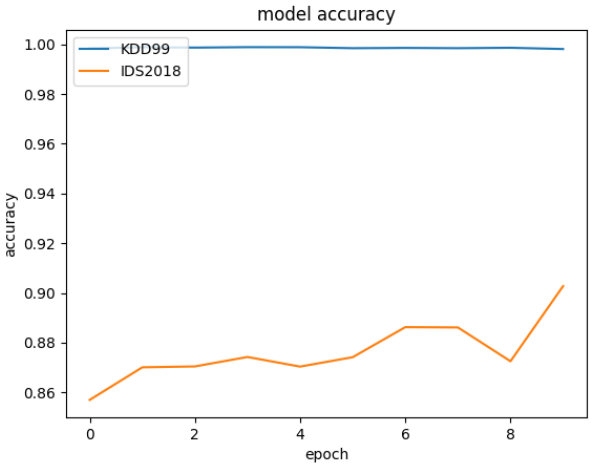
(a) CNN



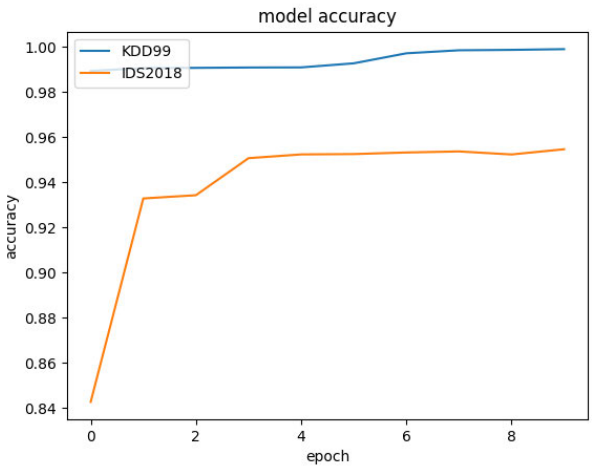
(b) Encoder



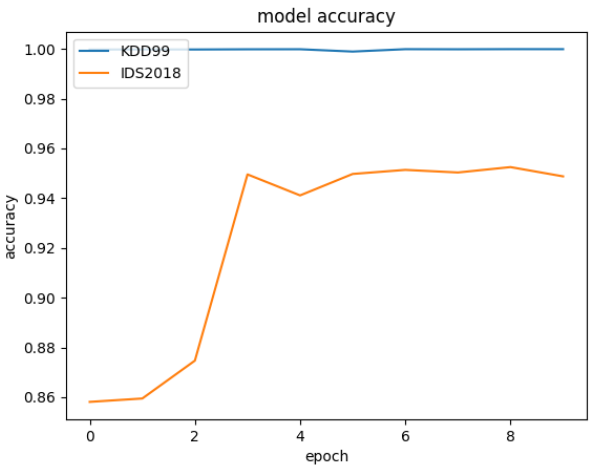
(c) FCN



(d) RNN



(e) TCN+LSTM



(f) U-Net

FIGURE 7. Validation accuracy.

TABLE 7. Averaged evaluation of methods for multi-class classification on CSE-CIC-IDS2018.

Methods	Accuracy	Precision	Recall	F-Score	Time for one epoch	Trainable parameters
CNN	0.8635	0.8907	0.8627	0.8764	2 min	1,418
Autoencoder	0.9303	0.9337	0.9236	0.9285	5 min	3,251,598
FCN	0.6943	0.6965	0.6943	0.6954	3 min	266,510
RNN	0.9166	0.9245	0.9034	0.9137	2.5 min	18,190
U-Net (Our)	0.9465	0.9488	0.9455	0.9471	10 min	11,472,366
TCN (our)	0.9753	0.9789	0.9725	0.9722	5 min	433,950
TCN + LSTM (our)	**0.9777**	**0.9794**	**0.9753**	**0.9773**	11 min	434,592

TABLE 8. Results of testing TCN+LSTM for each class of CSE-CIC-IDS2018.

Class	Precision	Recall	F-Score	Total number
Benign	0.97	1	0.988	607,551
Bot	1	0.988	0.994	28,577
Brute Force - Web	0	0	0	61
Brute Force - XSS	0	0	0	25
DDOS attack - HOIC	1	0.998	1	68,696
DDOS attack-LOIC-UDP	0.6	0.588	0.594	182
DoS attacks-GoldenEye	0.982	0.77	0.862	4,153
DoS attacks-Hulk	1	1	1	46,021
DoS attacks-SlowHTTPTest	1	1	1	13,910
DoS attacks-Slowloris	0.78	0.596	0.672	1,105
FTP-BruteForce	1	1	1	19,428
Infiltration	0.022	0	0	16,151
SQL Injection	0	0	0	6
SSH-Bruteforce	0.998	1	1	18,922

TABLE 9. Results of testing U-Net for each class of CSE-CIC-IDS2018.

Class	Precision	Recall	F-Score	Total number
Benign	1	0.328	0.466	607,551
Bot	1	0.888	0.934	28,577
Brute Force -Web	0.956	0.256	0.392	61
Brute Force -XSS	1	0.328	0.466	25
DDOS attack-HOIC	1	1	1	68,696
DDOS attack-LOIC-UDP	1	1	1	182
DoS attacks-GoldenEye	1	0.79	0.874	4,153
DoS attacks-Hulk	1	0.982	0.99	46,021
DoS attacks-SlowHTTPTest	1	0.748	0.774	13,910
DoS attacks-Slowloris	1	0.756	0.842	1,105
FTP-BruteForce	1	1	1	19,428
Infiltration	0.2	0	0	16,151
SQL Injection	0.634	0.202	0.296	6
SSH-Bruteforce	1	1	1	18,922

by a huge number of trainable parameters (11,472,366). TCN+LSTM takes time for training because of the second branch with LSTM as it can be seen from Table 4 and Table 7. On the other side, thanks to training architectures in independent branches and concatenation of outputs of them, it is possible to achieve better results.

Additionally, utilization of the focal loss function in the TCN+LSTM model helps to get more accurate results, because this loss function was originally proposed to train architectures on class imbalanced data. It was shown in practice, that the focal loss function can be effectively used for time series classification. This experiment has shown, that there is still a space for improvements of existing methods for anomaly detection and classification. There were a lot of approaches, which have achieved high accuracy values. However, they were trained on old datasets, which suffer from redundancy samples and old data. We used a modern dataset additionally to old datasets, because it has samples from actual network behaviour and allows us to test the models whether they are really able to provide a correct classification.

Moreover, we applied two different architectures, which has not been applied for the task of anomaly detection on the CSE-CIC-IDS2018 dataset. Additionally, according to charts in fig. 7, where the validation accuracy from training is drawn, it can be concluded, that for KDD99 the values are always around 0.99–1.00. However, results of testing show accuracy around 0.90–0.92. Obviously, the models are overfitted. On the other hand, the validation accuracy on CSE-CIC-IDS2018 grows gradually and the testing results correspond with the validation accuracy. The possible reasons for such different testing results are: the CSE-CIC-IDS2018 dataset is really huge and contains much more information, from which the model can learn, the KDD99 contains a lot of redundant information, so the models are not able to get enough information to be trained well. An interesting thing is that FCN is successful on the KDD99, but has worse results on the CSE-CIC-IDS2018. This fact can signalize that methods, which were evaluated on old datasets can fail on newer and more complex ones and different complexity should be used. Consequently, these methods would fail in real network applications. This difference in results on old and new datasets is also proved by some works, which we introduced in Table 1.

VI. CONCLUSION

The problem of anomaly detection and network traffic classification is challenging, because of the lack of actual datasets. Most approaches are trained and tested on old datasets, such as KDD99 or NSL-KDD. Since technology is developing rapidly, the amount of data also increases and the behaviour of network communication is also changing. Consequently, the network attacks are also improved: their behaviour is more complex than it was some years ago, new types have also appeared.

For security reasons, it is necessary to have systems and algorithms, which are able to detect these attacks and to classify them correctly. The old datasets are not suitable for these goals, because the proposed methods can achieve good results on a relatively simple dataset, but they can fail on something more complex. This is our motivation to utilize the dataset from 2018, which covers the modern attacks and has been created from real network traffic. The second reason to pay attention to new datasets is to avoid overfitting, because

the known problem of KDD99 is suffering from the redundant data.

The first step was to test on KDD99 and CSE-CIC-IDS2018 existing classifiers: CNN, Encoder, FCN and RNN, to create the baseline. The results have shown, that they are not so accurate and there is a space for improvements.

As an attempt to get more accurate results, the two models were proposed: the first one is based on U-Net and the second one is based on Temporal Convolutional Network and Long Short-Time Memory. Both models were adapted for the classification task. Taking into account, that the used dataset is imbalanced, the focal loss function was used in the second model. According to results, both architectures outperform existing models on the CSE-CIC-IDS2018, and the U-Net model gives better results on the KDD99. The achieved accuracy is very promising: the U-Net model has accuracy 93% and 94% on KDD99 and CSE-CIC-IDS2018 respectively, TCN+LSTM has 92% and 97%. Moreover, U-Net model is more able to classify attacks, in spite of the fact, that there were small numbers of samples in training and testing sets.

We compared the training and testing results and proved, that models trained on the KDD99 have the tendency to overfit, however, models trained on CSE-CIC-IDS2018 have almost the same results for training and testing sets. The next point we noticed, that the model, which is successful on the old dataset can easily fail on a modern dataset. It means, that for real-world applications it is necessary to evaluate the developed algorithms on more complex and modern data, than it was 20 years ago.

The future work in this field of research are improvements of methods used for attack classification, for example, experimenting with hyperparameters, loss functions and architectures. For sure, it is necessary to utilize modern datasets, which allow to evaluate algorithms on complex data, so the methods can be useful for real-world applications. We believe, that our conclusions and proposed methods would be helpful for other researches in this field of study.

REFERENCES

[1] *Cybersecurity Market Growth, Trends, COVID-19 Impact, and Forecasts (2021–2026)*. Market Research Reports. [Online]. Available: https://www.researchandmarkets.com/reports/4591630/cybersecurity-market-growth-trends-covid-19

[2] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.

[3] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, 2018, pp. 108–116.

[4] C. Lea, "Temporal convolutional networks: A unified approach to action segmentation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 47–54.

[5] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, 2015, pp. 234–241.

[6] G. Yan, "Network anomaly traffic detection method based on support vector machine," in *Proc. Int. Conf. Smart City Syst. Eng. (ICSCSE)*, Nov. 2016, pp. 3–6.

[7] N. Farnaaz and M. A. Jabbar, "Random forest modeling for network intrusion detection system," *Proc. Comput. Sci.*, vol. 89, pp. 213–217, May 2016.

[8] Y. Liao and V. Vemuri, "Use of K-nearest neighbor classifier for intrusion detection," *Comput. Secur.*, vol. 21, no. 5, pp. 439–448, Oct. 2002.

[9] W. Li and Q. Li, "Using naive Bayes with AdaBoost to enhance network anomaly intrusion detection," in *Proc. 3rd Int. Conf. Intell. Netw. Intell. Syst.*, Nov. 2010, pp. 486–489.

[10] V. Kotu and B. Deshpande, *Data Science: Concepts and Practice*. San Mateo, CA, USA: Morgan Kaufmann, 2018.

[11] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features dimensionality reduction approaches for machine learning based network intrusion detection," *Electronics*, vol. 8, no. 3, p. 322, 2019.

[12] F. Salo, A. B. Nassif, and A. Essex, "Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection," *Comput. Netw.*, vol. 148, pp. 164–175, Jan. 2019.

[13] M. Catillo, M. Rak, and U. Villano, "2L-ZED-IDS: A two-level anomaly detector for multiple attack classes," in *Proc. Workshops Int. Conf. Adv. Inf. Netw. Appl.* Cham, Switzerland: Springer, 2020, pp. 687–696.

[14] F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, "A novel two-stage deep learning model for efficient network intrusion detection," *IEEE Access*, vol. 7, pp. 30373–30385, 2019.

[15] J. Kim, H. Kim, M. Shim, and E. Choi, "CNN-based network intrusion detection against denial-of-service attacks," *Electronics*, vol. 9, p. 916, Jun. 2020.

[16] W. Jo, S. Kim, C. Lee, and T. Shon, "Packet preprocessing in CNN-based network intrusion detection system," *Electronics*, vol. 9, no. 7, p. 1151, Jul. 2020.

[17] J. Kim, Y. Shin, and E. Choi, "An intrusion detection model based on a convolutional neural network," *J. Multimedia Inf. Syst.*, vol. 6, no. 4, pp. 165–172, Dec. 2019.

[18] D. Kwon, K. Natarajan, S. C. Suh, H. Kim, and J. Kim, "An empirical study on network anomaly detection using convolutional neural networks," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2018, pp. 1595–1598.

[19] N. Chouhan, A. Khan, and H.-U.-R. Khan, "Network anomaly detection using channel boosted and residual learning based deep convolutional neural network," *Appl. Soft Comput.*, vol. 83, Oct. 2019, Art. no. 105612.

[20] P. Wu, H. Guo, and N. Moustafa, "Pelican: A deep residual network for network intrusion detection," in *Proc. 50th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, Jun. 2020, pp. 55–62.

[21] T.-T.-H. Le, Y. Kim, and H. Kim, "Network intrusion detection based on novel feature selection model and various recurrent neural networks," *Appl. Sci.*, vol. 9, no. 7, p. 1392, Apr. 2019.

[22] S. Kaur and M. Singh, "Hybrid intrusion detection and signature generation using deep recurrent neural networks," *Neural Comput. Appl.*, vol. 32, no. 12, pp. 7859–7877, Jun. 2020.

[23] D. Li, "MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks," in *Proc. Int. Conf. Artif. Neural Netw.* Cham, Switzerland: Springer, 2019, pp. 703–716.

[24] P. C. Ngo, A. A. Winarto, C. K. L. Kou, S. Park, F. Akram, and H. K. Lee, "Fence GAN: Towards better anomaly detection," in *Proc. IEEE 31st Int. Conf. Tools With Artif. Intell. (ICTAI)*, Nov. 2019, pp. 141–148.

[25] N. Dionelis, M. Yaghoobi, and S. A. Tsaftaris, "Boundary of distribution support generator (BDSG): Sample generation on the boundary," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 803–807, doi: 10.1109/ICIP40778.2020.9191341.

[26] R. S. Sutton and G. A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

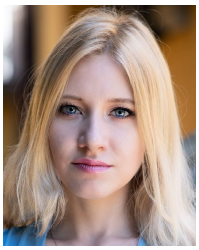
[27] G. Caminero, M. Lopez-Martin, and B. Carro, "Adversarial environment reinforcement learning algorithm for intrusion detection," *Comput. Netw.*, vol. 159, pp. 96–109, Aug. 2019.

[28] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, "Application of deep reinforcement learning to intrusion detection for supervised problems," *Expert Syst. Appl.*, vol. 141, Mar. 2020, Art. no. 112963.

[29] A. Aldweesh, A. Derhab, and A. Z. Emam, "Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues," *Knowl.-Based Syst.*, vol. 189, Feb. 2020, Art. no. 105124.

[30] A. Divekar, M. Parekh, V. Savla, R. Mishra, and M. Shirole, "Benchmarking datasets for anomaly-based network intrusion detection: KDD CUP 99 alternatives," in *Proc. IEEE 3rd Int. Conf. Comput., Commun. Secur. (ICCCS)*, Oct. 2018, pp. 1–8.

[31] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, 2012.
[32] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.
[33] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 big data," *J. Big Data*, vol. 7, no. 1, pp. 1–19, Dec. 2020.
[34] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 184–208, 1st. Quart., 2016.
[35] J. Goh, "A dataset to support research in the design of secure water treatment systems," in *Proc. Int. Conf. Crit. Inf. Infrastruct. Secur.* Cham, Switzerland: Springer, 2016, pp. 88–89.
[36] C. M. Ahmed, V. R. Palleti, and A. P. Mathur, "WADI: A water distribution testbed for research in the design of secure cyber physical systems," in *Proc. 3rd Int. Workshop Cyber-Physical Syst. Smart Water Netw.*, Apr. 2017, pp. 25–28.
[37] Z. Chkirbene, S. Eltanbouly, M. Bashendy, N. AlNaimi, and A. Erbad, "Hybrid machine learning for network anomaly intrusion detection," in *Proc. IEEE Int. Conf. Informat., IoT, Enabling Technol. (ICIoT)*, Feb. 2020, pp. 163–170.
[38] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *J. Netw. Comput. Appl.*, vol. 60, pp. 19–31, Jan. 2016.
[39] N. Moustafa and J. Slay, "The significant features of the UNSW-NB15 and the KDD99 data sets for network intrusion detection systems," in *Proc. 4th Int. Workshop Building Anal. Datasets Gathering Exper. Returns Secur. (BADGERS)*, Nov. 2015, pp. 25–31.
[40] A. Özgür and H. Erdem, "The impact of using large training data set KDD99 on classification accuracy," *PeerJ PrePrints*, vol. 5, Mar. 2017, Art. no. e2838v1.
[41] T. Wen and R. Keyes, "Time series anomaly detection using convolutional neural networks and transfer learning," 2019, *arXiv:1905.13628*. [Online]. Available: http://arxiv.org/abs/1905.13628
[42] Y. Liu, H. Dong, X. Wang, and S. Han, "Time series prediction based on temporal convolutional network," in *Proc. IEEE/ACIS 18th Int. Conf. Comput. Inf. Sci. (ICIS)*, Jun. 2019, pp. .
[43] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Müller, "Deep learning for time series classification: A review," *Data Mining Knowl. Discovery*, vol. 33, no. 4, pp. 917–963, Jul. 2019.
[44] N. Oliveira, I. Praça, E. Maia, and O. Sousa, "Intelligent cyber attack detection and classification for network-based intrusion detection systems," *Appl. Sci.*, vol. 11, no. 4, p. 1674, Feb. 2021, doi: 10.3390/app11041674.
[45] S. Iqbal, M. U. Ghani, T. Saba, and A. Rehman, "Brain tumor segmentation in multi-spectral MRI using convolutional neural networks (CNN)," *Microsc. Res. Technique*, vol. 81, no. 4, pp. 419–427, Apr. 2018.
[46] F. Karim, S. Majumdar, H. Darabi, and S. Chen, "LSTM fully convolutional networks for time series classification," *IEEE Access*, vol. 6, pp. 1662–1669, 2017.
[47] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*. [Online]. Available: http://arxiv.org/abs/1803.01271
[48] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 156–165.
[49] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.
[50] K. Pasupa, S. Vatathanavaro, and S. Tungjitnob, "Convolutional neural networks based focal loss for class imbalance problem: A case study of canine red blood cells morphology classification," *J. Ambient Intell. Hum. Comput.*, 2020, doi: 10.1007/s12652-020-01773-x.
[51] R. Yao, N. Wang, Z. Liu, P. Chen, and X. Sheng, "Intrusion detection system in the advanced metering infrastructure: A cross-layer feature-fusion CNN-LSTM-based approach," *Sensors*, vol. 21, no. 2, p. 626, Jan. 2021.



ANZHELIKA MEZINA received the bachelor's and master's degrees in information security from the Brno University of Technology, in 2018 and 2020, respectively, where she is currently pursuing the Ph.D. degree. Her research interests include information security, artificial intelligence, and computer vision.

RADIM BURGET is currently an Associate Professor with the Brno University of Technology, where he is heading the Signal Processing Program at the SIX Research Centre. He has been involved in the research of artificial intelligence for many years and in plenty of research projects, which include projects funded on European level, national level, or privately funded projects. He is cooperating with the companies, such as Honeywell, Mitsubishi Electric, RapidMiner, and Konica-Minolta.

CARLOS M. TRAVIESO-GONZÁLEZ is currently a Full Professor in signal processing and pattern recognition and the Head of the Signals and Communications Department, ULPGC, teaching on subjects on signal processing and learning theory, since 2001. He has published journal and conference papers and patents from the participation of European, international, and Spanish research projects. He is an Evaluator of project proposals for European Union (H2020), for Medical Research Council (MRC, U.K.), Spanish Government (ANECA), for ANR (France), for DAAD (Germany), and other institutions.