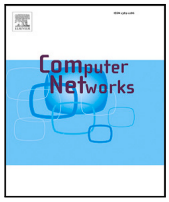
Computer Networks 249 (2024) 110493

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



Reinterpreting Usability of Semantic Segmentation Approach for Darknet Traffic Analysis

Anzhelika Mezina [a,*], Radim Burget [a], Aleksandr Ometov [b,*]

[a] Brno University of Technology, FEEC, Department of Telecommunications, Technicka 12, Brno, 616 00, Czech Republic
[b] Electrical Engineering Unit, Faculty of Information Technology and Communication Sciences, Tampere University, Tampere, 33720, Finland

ARTICLE INFO

Keywords:
Deep learning
Darknet detection
UNet++
Feature analysis
Traffic classification

ABSTRACT

With a growing number of smart interconnected devices and services, managing and controlling network traffic is getting more complicated. Among the network traffic, the Darknet-related one is particularly interesting, as it is often used for anonymous and illicit activities that pose cyber security threats. Therefore, designing and developing methods for detecting and categorizing Darknet traffic is essential. Applying Deep Learning (DL) is one of the most suitable options in this case. The main reasons are the ability to process a large amount of data and detect the hidden patterns and relationships in these data. This work proposes a DL architecture based on UNet++, which can detect and categorize anonymous traffic. The core idea of this model is semantic segmentation, which can identify meaningful segments that share some common patterns in given data. Hereby, semantic segmentation is postulated as a possible way to investigate Darknet traffic to find some common and related features instead of widely used Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM). According to the results on comparison with other Machine Learning (ML) and DL models, the UNet++ model outperforms the methods with a higher accuracy of 98.19% and 87.27% for Darknet detection and traffic categorization. Our work shows the potential of using UNet++ for network traffic analysis and Darknet traffic detection. We have also demonstrated that more advanced architecture with skip connections and trainable blocks provides more accurate results than pure U-Net, CNN, and other evaluated models.

1. Introduction

In recent years, the number of devices connected to the Internet, for example, smartphones, the Internet of Things (IoT), wireless sensors, and others, is growing at a tremendous pace [1]. Consequently, there is a vast opportunity for cybercrimes that seriously threaten network security and user privacy. One of the related examples is the utilization of *Darknet*, which is the *darkest* layers of the World Wide Web, as shown in Fig. 1.

Generally, Darknet is an overlay part of the Internet, which can be reached with special techniques, for example, The Onion Router (Tor) or Virtual Private Network (VPN). It is designed to provide anonymity and preserve the identity of sides involved in communication [2]. In most cases, the Darknet is associated with certain illegal processes. However, Darknet can be used for both legitimate and illegitimate purposes: from protecting privacy and identity in communication to selling something prohibited by the law, for example, drugs, weapons, and others [3]. For this aim, the Darknet markets are actively used. They provide an anonymous platform for selling illicit services and goods [4]. Therefore, analysis of Darknet traffic is essential because of the detection of unauthorized behavior and the prevention of possible malicious activities.

One self-explanatory and raising example of Darknet usage is related to the cryptocurrency operations directly connected to money laundering. According to The 2023 Crypto Crime Report by Chainalysis [6], the illicit transaction volume is still rising. Notably, the Darknet market also rose until 2021. However, the situation changed in 2022 because of several sanction restrictions. The mentioned situation is depicted in Fig. 2, which shows the detected transactions. However, there is also the possibility that some transactions and traffic are still undetected. That means the number of illegal actions can be much larger.

The amount of transferred information constantly changes because of the evolving number of devices involved in communications. Consequently, it is impossible to control it manually, thus, Artificial intelligence (AI) techniques, such as ML or DL, come out of shade [7]. Its main advantage is the ability to process and analyze a massive amount of information automatically. By training a model on such data, it becomes possible to assist network administrators and security analysts in detecting anomalies, threats, and attacks in real time and on a scale.

* Corresponding author.
E-mail addresses: anzhelika.mezina@vut.cz (A. Mezina), burgetrm@vut.cz (R. Burget), aleksandr.ometov@tuni.fi (A. Ometov).

https://doi.org/10.1016/j.comnet.2024.110493
Received 25 November 2023; Received in revised form 22 April 2024; Accepted 6 May 2024
Available online 10 May 2024

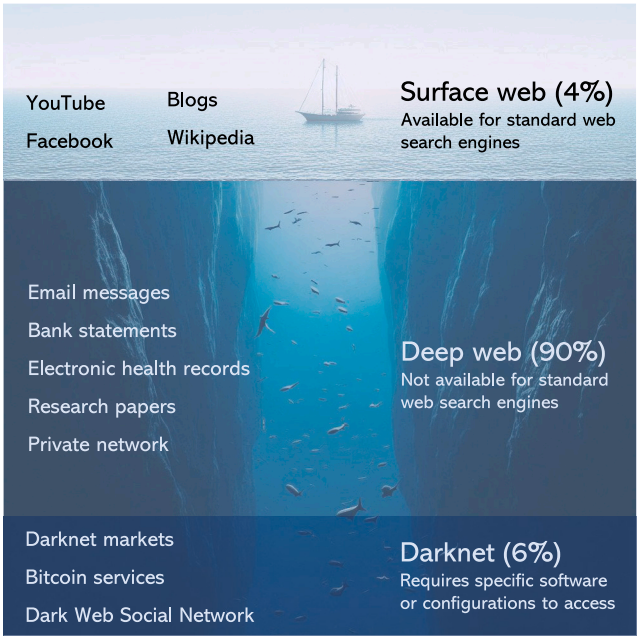


Fig. 1. Structure of the World Wide Web. Statistical data retrieved from [5]. The background image was created with the assistance of Bing Chat.

Today, many approaches for cyber security are based on the application of ML or DL, e.g., Intrusion Detection System (IDS) [8], steganography [9], malware detection in memory [10], etc. ML and DL approaches could also be used for Darknet traffic detection and categorization of activities.

Most state-of-the-art works focus on applying traditional ML methods, such as Random Forest (RF), Decision Tree (DT), and XGBoost (XGB). However, only some approaches utilize DL methods for this research field.

While traditional ML algorithms are less complex, making them less time-consuming, and they are interpretative. DL methods offer many benefits for analyzing such data, as their highly complex structure can handle the increasing complexity and volume of big data. Unlike traditional ML methods, which need a high-quality preprocessing step, such as feature selection, DL methods can automatically learn and select features from the data [11]. Additionally, the appropriate selection and design of architecture allow efficient extraction of essential features for processing a growing amount of data [12].

Nowadays, several groups of solutions can be defined and applied to this problem. The approaches based on CNN and LSTM can be good performing according to the metrics, but they still can be time-consuming, especially with the utilization of LSTM [13]. This point has already been discussed in a similar topic. Some other works use the translation of 1D to 2D data representation and apply CNN for image classification. There can be a potential increase in time processing since the CNN models for images would use more trainable weights. Additionally, the process of transformation from 1D to 2D is also an operation that can take time. The widespread use of ML algorithms can be efficient regarding time and hardware requirements. On the other side, these methods require extensive feature analysis and selection. This is a possible problem in the future regarding generalization and automation of the processes.

Nonetheless, to the best of the authors' knowledge, no work has studied the possibilities of semantic segmentation in this field of research. Originally, semantic segmentation was used for computer vision tasks and aimed at assigning labels to each "pixel" in the given "image" [14], essentially analyzing the 2D data. Consequently, it was possible to divide the 2D data into areas with similar patterns or features. Such a way of processing helps better understand the context of the given data. The application of this technique found its roots in the processing of 1D data [15–17].

The described technique can extract more detailed information and provide meaning to those segments. The majority of works used semantic segmentation for 2D data. However, some approaches are applied in the medical field of research to process 1D signals, for example, electrocardiogram (ECG) [16], plethysmography (PPG) [18].

Today, work must be done to understand how to apply semantic segmentation for traffic analysis. At the same time, this technique can benefit this field of research since the analyzed data are complex, and it is necessary to use advanced techniques to recognize those patterns. Instead of the utilization of widely used CNN and LSTM, this work studies the capabilities of semantic segmentation architectures for analysis of such data and compares them with traditional ML methods and several DL models.

Here can also be clarified what exactly is attempted to identify in this work. As it was mentioned above, the Darknet has the nature of anonymity and requires special applications for access.

Consequently, Darknet traffic **detection** is required to initially spot the related applications in the data flow, which is a process of analyzing and identifying traffic that is forwarded to the Darknet. On the other side, it is necessary to define, which signs can be used for identification. One of the possible solutions is to focus on the identification of applications used to reach the Darknet – Tor or VPN.

Another research task raised in this work is traffic **categorization**. This can be defined as the process of classifying network traffic according to the application that generated this data for transmission. Here, the aim is to find patterns that can identify the related application.

The main contribution: The paper introduces a novel approach based on NN architecture to detect and categorize Darknet traffic, demonstrating superior performance over existing methods. Our architecture is based on the semantic segmentation principles to Darknet traffic analysis, providing a refined categorization of applications.

The rest of this paper is structured as follows. Section 2 introduces recent works in this field of research. Section 3 presents the data preprocessing step. Sections 4 and 5 describe the used traditional ML and DL models for comparison and proposed model, respectively. Sections 6–8 present achieved results and discuss them. Section 9 proposes the future directions in this field of research. Section 10 concludes the work.

2. Related work

This Section represents the existing approaches for general encrypted traffic detection using DL architectures (see summary in Table 6) and methods focused on the target problem – Darknet traffic detection and categorization (see summary in Table 1).

2.1. Encrypted traffic detection

Encrypted traffic, in contrast to traditional packet-level analysis, requires more complex research and development activities. It could be considered having an increasing tendency to preserve users' privacy and make the utilization of technologies secure and safe (for non-malicious cases). On the other hand, inspecting the traffic to detect malicious behavior is becoming more difficult. To overcome this challenge, some studies focus on analyzing encrypted traffic.

One of the possible ways is to apply ML and DL algorithms for this field of research, e.g., the paper [19] provides experiments for three ML algorithms: Support Vector Machines (SVM), RF, and XGB. This work aims to identify the features that help distinguish encrypted malicious network traffic from benign one.

The work [20] proposes a method that combines natural language processing, such as Term Frequency - Inverse Document Frequency (TF-IDF) and ML for malicious encrypted traffic detection. The TF-IDF

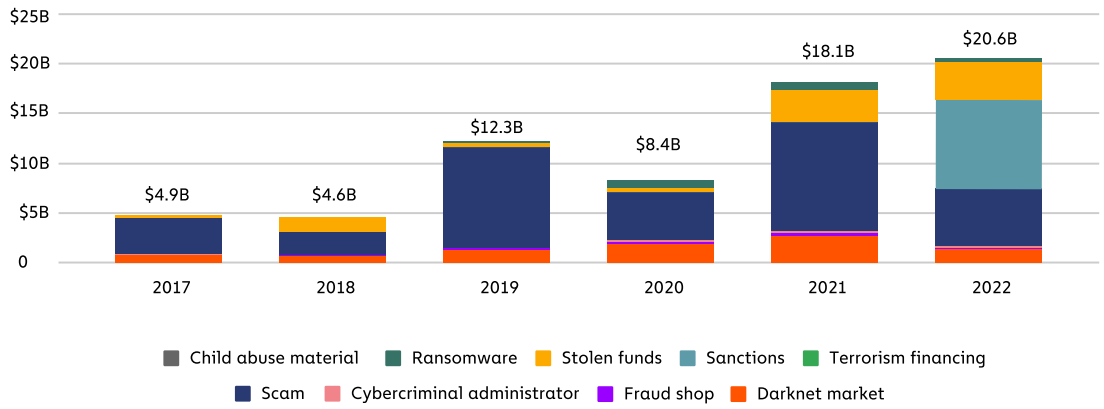


Fig. 2. Total cryptocurrency value received by illicit addresses.
Source: Reproduced from [6].

method is used for feature extraction, and the 1D CNN is used as a classifier. The achieved accuracy is 93.3% on the private dataset, which was created using the sky dome sandbox of QiAnXin Technology Research Institute.

Also, Reinforcement Learning (RL) can detect malicious encrypted traffic. The approach [21] is based on Q-networks and deep convolution Generative Adversarial Networks (GANs) to overcome the problem of the unbalanced dataset. The classification module is based on ResNet. The accuracy of the proposed model is 91.43% on the private dataset.

The Cost-Sensitive CNN handles the dataset imbalance in [22]. The main idea is to utilize the cost matrix to assign a cost to misclassification based on the class's distribution. The achieved precision for traffic description is 0.977, and the accuracy for traffic categorization is 97.9% on the ISCX VPN-nonVPN dataset.

The approach described in [23] is based on transforming tabular data into grayscale images. The prepared images are processed in parallel with an inception module and LSTM model for encrypted traffic classification. According to the results, it is possible to achieve an accuracy of 98% for the identification task on the ISCX 2016 dataset. Furthermore, the paper [24] also proposes an approach based on transforming flow data into images and applying the CNN for the categorization.

The authors of the work [25] utilized the multihead attention mechanism in a lightweight NN to efficiently classify encrypted traffic. According to the authors, the key point is the one-step interaction of all packets and the parallel computation of the multi-head attention mechanism.

Also, Recurrent Neural Network (RNN) can be used in this field of research. In [26], authors proposed an architecture consisting of the preprocessing and classification phase. The first phase aims to prepare data using flow segmentation, sampling, and vectorization techniques. The classification part is supposed to train end-to-end extraction of spatial features using CNN and to learn the temporal characteristics by stack Bidirectional Long Short-Term Memory (Bi-LSTM). The achieved accuracies are 99.4% and 95% in Tor/non-Tor binary and sixteen classification tasks, respectively.

A similar work was introduced in [27]. The authors also used CNN and LSTM to classify services, such as video streaming, social media, webmail, etc., focusing on new encrypted web protocols. For the experiments, the real-world mobile traffic dataset is used.

The study [28] also utilizes CNN. However, combined with the ant-lion metaheuristic algorithm and the self-organizing map, the issue of automated feature extraction will be addressed.

Another combination of learning approaches is presented in [29]. The authors propose a model based on CNN and RL to address the issue of optimal packet sampling amount to achieve a high classification rate in high-performance networks. The proposed method is supposed to reduce overhead on monitored entities.

Work [30] introduced using Vision Transformer (ViT) and demonstrated success. Additionally, they use augmentation by Bidirectional GAN to address the high-class imbalance problem. They have utilized the ISCX-Tor2016 dataset and achieved 99.59% accuracy.

2.2. Darknet traffic detection

In recent years, there has been much attention to the application of ML algorithms to security tasks.

The work [31] compares traditional ML algorithms for binary and multiclass classification to distinguish Darknet traffic. The best results were achieved by RF with an accuracy of 98%.

A similar work is introduced in [32]. The authors conducted several experiments for binary classification, quadruple classification, and traffic classification. The authors also applied the Synthetic Minority Over-sampling Technique (SMOTE) method to balance the sizes of the classes and the feature selection method. The best results are achieved by RF.

The authors of work [33] combine several algorithms, RF, k-Nearest Neighbours (k-NN), and DT to improve the performance and accuracy for Darknet categorization. The authors also proposed the two-layered Autoencoder (AE)-based defense mechanism against adversarial attacks.

The RF is also used in work [34]. They provided the feature selection with the algorithm Recursive Feature Elimination and selected 30 features for the following classification with algorithms.

The work [7] compared six ML methods, such as bagging DT ensembles, AdaBoost DT ensembles, RUSBoosted DT ensembles, optimizable DT, optimizable k-NN (O-KNN), and optimizable discriminant.

Notably, most works that focus on traditional ML methods give preference to DT or their modifications. It may happen not only because of efficiency but also because of the interpretability of the model. This feature makes DT worthwhile for the analytician, who can understand the patterns and rules of the model's decision. With this motivation, work [35] used the Gradient Boosting DT in combination with federated learning framework for IDS.

Additionally, many successes have been reported with applying DL algorithms to security tasks, including the Darknet traffic analysis.

The most frequently used NN architectures for this purpose were 1D CNN and LSTM, which are similar to encrypted network detection and categorization.

Most notably, most approaches use the dataset proposed in [36]. The authors of this dataset have also proposed a method for this task. The selected features are transformed into images and processed with a 2D CNN model with an accuracy of 86%.

Table 1
Summary of approaches for Darknet traffic detection (sorted by the publication year).

Ref.	Year	Main idea	Used dataset	Used technique
[36]	2020	The selected features are transformed into image. After that, the image is processed with 2D CNN.	CIC-Darknet-2020	CNN
[31]	2021	Traditional ML algorithms trained for binary and multiclass classification	CIC-Darknet-2020	*k*-NN, Multilayer Perceptron (MLP), RF, DT, XGB
[37]	2021	ML algorithms, such as DT, XGB, RF were compared for data balancing.	CIC-Darknet-2020	CNN, LSTM
[39]	2021	The numerical features are transformed into image data. 10 pretrained classification models were evaluated.	CIC-Darknet-2020	AlexNet, ResNet18, ResNet50, ResNet101, DenseNet, GoogLeNet, VGG16, VGG19, Inceptionv3, and SqueezeNet
[33]	2022	Model combines 3 learner: RF, *k*-NN, DT. The AE based mechanism is utilized against adversarial attacks.	CIC-Darknet-2020	Stacking Ensemble model
[7]	2022	Detection of Darknet traffic using ML methods for IoT networks	CIC-Darknet-2020	BAG-DT, ADA-DT, RUS-DT, O-DT, O-*k*-NN, O-DSC
[2]	2022	A self-attentive DL method, which extracts side-channel features from payload statistics.	CIC-Darknet-2020	CNN, Bi-LSTM
[32]	2022	Several ML algorithms were trained and evaluated. Additionally, the authors used the feature selection method and SMOTE method for class balancing.	CIC-Darknet-2020	DT, RF, Simple CART, *k*-NN, Naive Bayes, AdaBoost
[35]	2022	The proposed framework is based on federated learning and Gradient Boosting DT. The solution is supposed to be privacy-preserving, interpretable, and scalable Network Intrusion Detection System (NIDS).	CIC-Darknet-2020, DDoS2019, MalDroid2020, DoHBrw2020	Gradient Boosting DT, Federated Learning
[34]	2023	Extracted features are grouped with n-gram approach	CIC-Darknet-2020	DT, RF, MLP
[38]	2023	The approach is based on RF. The data is augmented using SMOTE and AC-GAN.	CIC-Darknet-2020	RF
[40]	2023	The numerical features are transformed into image data using several methods. The classification is performed using XGB and ResNet-50	CIC-Darknet-2020	XGB, ResNet-50
Proposed		**The proposed method utilizes the principle of semantic segmentation for Darknet network analysis. For this purpose, the Unet++ model was modified for application on 1D data.**	**CIC-Darknet-2020**	**Unet++**

For example, the work [2] combines 1D CNN, Bi-LSTM, and the self-attention mechanism. The proposed system captures local spatial–temporal features and global intrinsic dependency relationships. The achieved accuracy is 92.22%.

Another approach is introduced in paper [37] that also combines CNN and LSTM. Additionally, the authors used Principal Component Analysis (PCA), DT, and XGB to select the 20 most significant features. The best results achieved by XGB feature selection are AUC is 0.95, F1 score is 0.89, recall is 0.88, and precision is 0.9.

Another way of processing tabular data containing the features' vectors with corresponding labels is transforming them into 2D representations- into grayscale images. After that, it is possible to apply 2D CNN and Auxiliary-Classifier GAN, which was done fin approach [38]. The augmentation method was a SMOTE. The results using CNN are promising – accuracy is 89.1%.

A similar method, based on the representation of input data as a 2D image, was used in work [39]. However, the authors used pretrained CNN, such as AlexNet, ResNet18, VGG16, etc., to extract the features and feed them into the baseline classifier. The highest accuracy was achieved by combining VGG19 and RF – 94.89%.

The work [40] used ResNet-50 recombination with several tabular-to-image algorithms, such as Image Generator for Tabular Data, DeepInsight, vector-of-feature wrapping, and newly introduced Binary Image Encoding (BIE). They trained the model to categorize network application types.

According to the studied literature, most works apply the combination of CNN with LSTM or translate tabular data into an image and perform the classification task to detect encrypted or Darknet traffic. However, no work would apply the more advanced architecture of CNN for fast and accurate predictions. The complex preprocessing step also increases the latency of predictions, which has an impact on the whole system. Therefore, our goal is to introduce a methodology utilizing the UNet++ model, designed to detect and categorize Darknet traffic accurately.

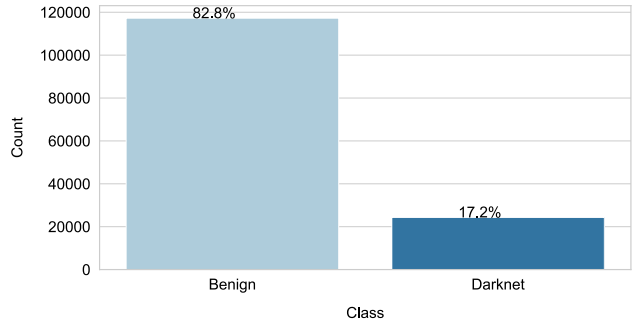
3. Dataset preprocessing

The well-known dataset, CIC-Darknet2020 [36], from the Canadian Institute for Cybersecurity was used for all experiments. This dataset's authors merged ISCXTor2016 and ISCXVPN2016 to create a complete Darknet dataset covering Tor and VPN traffic.

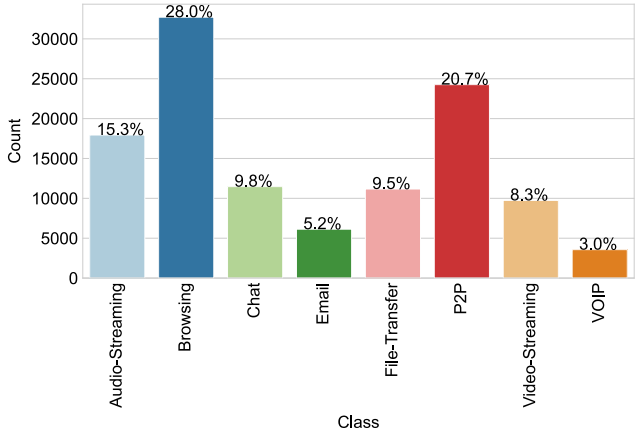
In this paper, 63 features describing the traffic were used. We have excluded such features as "Flow ID", "Src IP", "Dst IP", and "Timestamp", since they do not provide significant information for the classification task. We also excluded the following features: "Bwd PSH Flags", "Fwd URG Flags", "Bwd URG Flags", "URG Flag Count", "CWE Flag Count", "ECE Flag Count", "Fwd Bytes/Bulk Avg", "Fwd Packet/Bulk Avg", "Fwd Bulk Rate Avg", "Bwd Bytes/Bulk Avg", "Subflow Bwd Packets", "Active Mean", "Active Std", "Active Max", "Active Min", because the values are the same for all samples, consequently, have no information value.

Tables 7 and 8 present the *p*-values for features not excluded from experiments. Generally, *p*-value means the probability of the significance of an observed effect (the less value is better) [41]. According to the introduced tables, most features have the *p*-values less than 0.05, and most of the works consider this threshold statistically significant.

The next step is converting categorical variables, such as "Fwd PSH Flags", "FIN Flag Count", "SYN Flag Count", "Subflow Fwd Packets", "Fwd Seg Size Min", into indicator variables. Also, the samples with empty fields and duplication were dropped.



(a) Binary classification



(b) Multiclass classification

Fig. 3. Class distribution.

The resulting dataset contains two labels indicating if the traffic is Darknet (VPN or Tor), which has 24,094 samples, and benign (Non-VPN, Non-Tor), which has 92,872 samples.

The second label categorizes the samples into 8 classes: Audio-Streaming (17,942 samples), Browsing (32,713 samples), Chat (11,468), Email (6145), File-Transfer (11,169), Peer-to-Peer (P2P) (24,260), Voice over Internet Protocol (VoIP) (3565), and Video-Streaming (9740). The class distributions are introduced in Figs. 3(a) and 3(b).

In the next step, the data were normalized and scaled with Quantile Transformer [42], which transforms the features into a uniform distribution.

Generally, the dataset was divided into training and testing sets for ML algorithms. The training set is 80% and testing set is 20% of the whole dataset.

As a common practice for DL models, the dataset was split into training, validation, and testing sets. The training set is 64%, the validation set is 16%, and the testing set is 20%.

Figs. 3(a) and 3(b) show that the dataset is imbalanced, especially in the case of binary classification. It can signalize the possible overfitting problem. To address this issue, the dataset split was done based on label distribution in the initial dataset. In this way, the distribution in training and testing sets is preserved and corresponds to the initial dataset.

4. Baseline

In this work, several traditional ML and DL models were used for evaluation. They were trained and evaluated under the same conditions as the proposed model. In comparison to all other approaches, only the chosen ML and DL models were utilized, as the publicly accessible source codes for other methods were unavailable, thereby preventing their replication.

4.1. Traditional ML models

RF [43] is the classifier, which consists of multiple DTs, and the result of this method is aggregated from the outputs of all these trees. This method is popular because of its simplicity and good performance. Such parameters, as the number of trees and the number of split variables at each tree node, should be considered during training RF.

DT [43] is a representative method for classification or regression tasks, which is learned in a supervised manner. The high dimensional data are split into partitions in iterations. Each branch represents the decision rule, and the leaf shows the outcome. Since this representation is easily interpreted, it is often used for expert or recommendation systems.

***k*-NN** [43] is the supervised method for classification, which labels the samples based on the majority of k-nearest patterns in data space.

MLP [43] is a simple NN, which usually consists of 3 layers: input layer, hidden layers – fully connected layers, and output layer. It transforms the input dimension to the desired dimension.

Logistic Regression (LR) [43] is the statistical technique to find the relationships between independent variables and binary outcome values. The main advantages of this method are its easy implementation and its efficiency for binary classification. However, it fails in the prediction of continuous outcomes.

4.2. Optimal parameters for binary classification

To ensure that all the methods employed in this work were optimized, a randomized search technique with cross-validation $n = 5$ was utilized. This technique was selected for its ability to efficiently explore the large search space for ML algorithms for large datasets. Compared with Grid search, Random search is less time-consuming, making it more suitable. The search space for hyper-parameters is as below.

- Space for RF:
  - Number or estimators: 1 to 60;
  - Number of features: 1 to 15;
  - Depth: 2 to 10;
  - Criterion: $gini$, $entropy$.
- Space for DT:
  - Max features: $auto$, $sqrt$, $log2$;
  - Depth: 2 to 15;
  - Criterion: $gini$, $entropy$.
  - Minimum number of samples in leaf: 1 to 20.
- Space for *k*-NN:
  - Number of neighbors: 3 to 30;
  - Weights: $uniform$, $distance$;
  - Algorithm: $auto$, $ball\ tree$, $kd\ tree$, $brute$.
- Space for MLP:
  - Solver: $lbfgs$, sgd, $adam$;
  - Hidden layer size: 2 to 150;
  - maximum iterations: 2 to 150.
- Space for the LR:
  - C: 0 to 10;
  - Solver: $newton-cg$, $lbfgs$, sag, $saga$;

In spite of the relatively large size of the dataset, the possible overfitting problem can appear due to dataset imbalance. That is why, the application of cross-validation in 50 iterations is required to find the best combination of hyper-parameters.

The found ones, which allow to achieve accurate results, are:

1. **RF**: Number of estimators: 46; max features: 13; max depth: 9; criterion: entropy;
2. **DT**: Max depth: 14; max features: auto; criterion: entropy;
3. *k*-**NN**: Weights: distance; number of neighbours: 3; algorithm: ball tree;
4. **MLP**: Solver: Adam; max iterations: 148; hidden layer sizes: 90;
5. **LR**: Solver: lbfgs.

4.3. Optimal parameters for multiclass classification

Similar to binary classification, the Random search with cross-validation was done to find the optimal parameters for each algorithm, with similar search space, as in Section 4.2. The found hyper-parameters are introduced below:

1. **RF**: Number of estimators: 55; max features: 13; max depth: 9;
2. **DT**: Max depth: 14; max features: auto; criterion: entropy; minimum samples leaf: 1;
3. *k*-**NN**: Weights: distance; number of neighbours: 2; algorithm: brute;
4. **MLP**: Solver: Adam; max iterations: 148; hidden layer sizes: 90;
5. **LR**: Solver: newton-cg; C: 10.

4.4. DL models

CNN consists of several convolutional blocks that are composed of two 1D convolutional layers, a dropout layer with a rate of 0.1, layer normalization, and a max pooling layer with a pool size of 2. Three blocks are used with the parameters in convolutional layers: the number of filters are 32, 64, and 128; kernel sizes are 9, 7, and 5. After convolutional layers, the global average pooling is applied to reduce and sum up the extracted information from previous blocks. The final classification is performed with the fully connected network, which consists of Dense layers with 64 and 16 neurons and a dropout layer with a rate of 0.3. The classification is performed with a Dense layer with several neurons and activation functions corresponding to the number of classes (for binary classification – one neuron and sigmoid activation function; for multiclass classification – eight neurons and softmax activation function). The loss functions are binary cross-entropy for binary classification and categorical cross-entropy for multiclass classification. The used optimizer is Adam, with a learning rate of 0.0001.

AE is frequently used architecture for 1D data processing. The used architecture consists of the encoder and decoder parts. The encoder contains convolutional blocks composed of two convolutional layers and max pooling layers, with kernel size 3 and feature maps of 64, 128, 256, 512, and 1024. The decoder part is performed with blocks composed of an Upsampling layer and two convolutional layers with a kernel size of three and feature maps of 512, 256, 128, and 64. The classification is performed with a Flatten layer, a Dense layer with 512 neurons, and a Dropout with a rate of 0.5. The output layer, loss function, and optimizer are the same as applied in CNN.

LSTM is widely used for processing network traffic. The architecture used for comparison consists of LSTM layers with 64 and 32 units and dropout layers with a rate of 0.2, which are places between them. After that, a Flatten layer is applied, and the output layer performs the final classification. The output layer and used hyper-parameters and optimizer are similar to those used in CNN.

5. Proposed model

This section represents the description of the proposed model and used metrics for evaluation.

5.1. Description

This work adapts the architecture of UNet++ [44] for Darknet traffic detection and categorization. Initially, this architecture was proposed for the segmentation task, similar to the original U-Net [45] model. However, in our previous research [13], we have applied the U-Net architecture for network anomaly detection and have proved its efficiency for this task. Continuing our research, we have utilized the modified version of this architecture, UNet++, and changed it to process the 1D data.

Generally, this model consists of two branches: encoder and decoder. The encoder aims to downscale the input data and represent them in a so-called latent space, that extracts the important information. The decoder part aims to reconstruct the output signal based on the latent space. Additionally, the U-Net-based architectures apply additional so-called skip connections, which tune the information from the encoding part to the decoder.

Compared with the original U-Net model, the UNet++ model has trainable blocks in these connections, allowing more efficient tuning of information.

In the initial phase, the input vector is padded with two zeros. It allows to safely downscale and upscale the extracted features from input data and concatenate them at each level.

The proposed model is depicted in Fig. 4. It consists of 5 levels: the first has 5 blocks, the second has 4 blocks, and so on. The blocks $X^{1,1}$, $X^{2,1}$, $X^{3,1}$, and $X^{4,1}$ perform the downscaling of features using the Average Pooling layer with stride 2, and the upscaling operation is done with the Upsampling layer.

Each internal block (the ConvBlocks $X^{3,2}$, $X^{2,2}$, $X^{2,3}$, $X^{1,2}$, $X^{1,3}$, $X^{1,4}$) has a connection with the related block from the encoder part, bridging the semantic gap between the encoder and the decoder. Additionally, this model has connections using the upsampling layer between different levels, allowing information to be extracted on different levels of abstraction.

ConvBlocks consist of a Convolutional layer, Batch normalization layer, and Activation layer with activation function Gaussian Error Linear Unit (GELU) [46], which is defined for input x as

$$GELU(x) = xP(X \leq x) = x\Phi(x) = \frac{x}{2}[1 + erf(\frac{x}{\sqrt{2}})], \tag{1}$$

where $\Phi(x) = P(X \leq x)$, if $X \sim \mathcal{N}(0, 1)$ – the standard Gaussian cumulative distribution function. Each convolutional layer has a different number of extracted feature maps. In this way, all convolutional layers in level 5 have 512 feature maps, in the level 4 – 256, the level 3 – 128, level 2 – 64, and level 1 – 32. The used kernel size is 3.

After the UNet++ model, the Flatten layer follows. The final classification part consists of a fully connected network with a Dense layer with 512 neurons, a Dropout with a rate of 0.5, and a Dense layer with a number of neurons corresponding to the number of classes.

The proposed model applied for two types of classification, binary and multiclass. In the case of binary classification, the last layer contains one neuron with a sigmoid activation function with input x and Euler's Constant e, which is formulated as follows:

$$f(x) = \frac{1}{1 + e^{-x}}. \tag{2}$$

For the multiclass classification, the last layer has eight neurons with softmax activation function, which is defined as:

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

Considering that the dataset is imbalanced, the focal loss function was utilized for training the model. The application of this loss function proved to be efficient for training over the imbalanced dataset. It is defined as [47]:

$$FL(p_t) = -\alpha_t(1 - p_t)^{\gamma} \log(p_t), \tag{4}$$

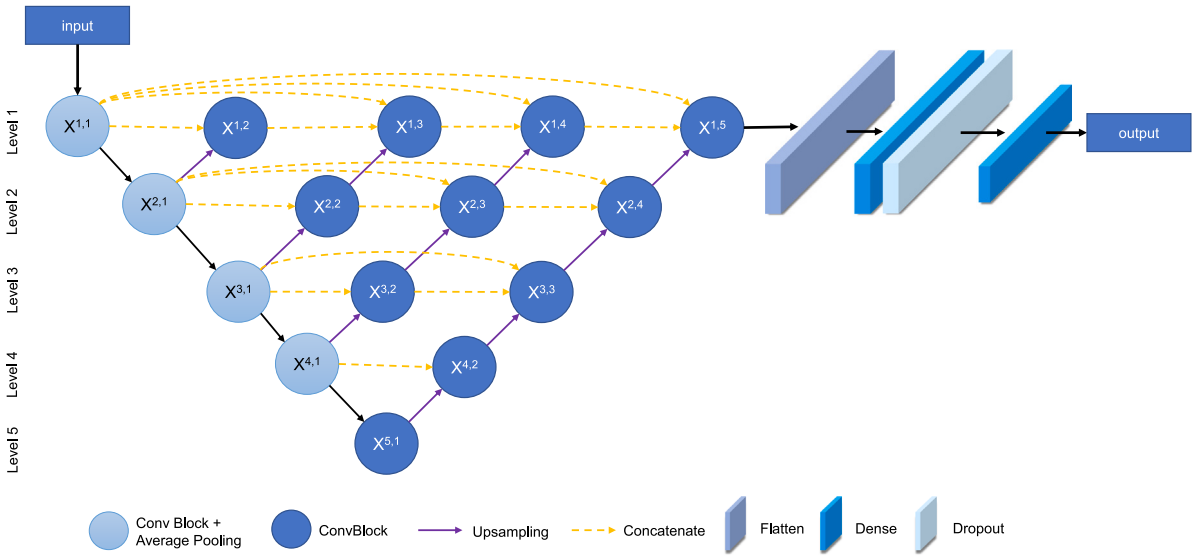


Fig. 4. Architecture of proposed NN model.

where p_t – the model's estimated probability for the class with label $y = 1$, $\alpha_t = 0.50$ – balancing factor, $\gamma = 1.5$ – modulating factor, which were selected empirically.

Instead of a widely used optimizer Adam, the new optimizer, so-called Lion [48] is applied in this work because of its memory efficiency and potential for accuracy improvement. The used learning rate is 0.0001.

5.2. Metrics

To evaluate the performance of the tested and proposed models, the following metrics were used [49]:

Accuracy describes how correct the trained model is in making predictions.

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{5}$$

Precision is the ratio of correct positive predictions to all predicted labels, determined as positive.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{6}$$

Recall is the ratio of true positive predictions to the total number of positive samples.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{7}$$

F1 score is a harmonic average of precision and recall, used to evaluate models trained on imbalanced datasets.

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{8}$$

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

6. Darknet traffic detection results

This section represents the results of evaluated models, including traditional ML, selected DL models, and the proposed one. This section consists of two parts. The first one is the results for the baseline, which shows and comments on the results of models included in the baseline, tacking the possible reasons for the achieved results. The second part demonstrates the results achieved by the proposed model.

To ensure that the evaluation was fair and objective, comprehensive experiments were conducted, and all models were trained and tested under identical conditions. The achieved results are compared with existing approaches.

6.1. Results for baseline

Firstly, the following ML algorithms were trained: RF, DT, *k*-NN, MLP, and LR. After finding the optimal hyper-parameters, the models were evaluated on the testing set. The achieved results are represented in Table 2 and Fig. 11. The best result among ML algorithms is the DT, which has achieved an accuracy 0.9762, F1 0.9413, precision 0.9563, balanced accuracy 0.9579, ROC-AUC 0.9579.

The second successful model is the RF, which performed with an accuracy 0.9738, F1 0.9353, precision 0.9515, balanced accuracy 0.9538, ROC-AUC 0.9538. The advantage of these models is the possibility of providing the representation as a DT, which can be used as a recommendation system with an explanation. However, the graphical representation of it in the tree's form is enormous since the minimum leaf sample size is one and the maximum depth is 9 – 14. It may indicate that the model has achieved high performance due to certain specific cases. Therefore, DL models are a more suitable option for generalizing on large-scale datasets.

On the other hand, the DL models used for comparison (LSTM, CNN, AE, and U-Net) achieved better results (except LSTM) than traditional ML methods. After analysis of the results of CNN, AE, and U-Net, the tendency is that more complex architectures can detect Darknet traffic more accurately. In this case, U-Net architecture reaches an accuracy 0.9803, F1 0.9516, precision 0.9618, recall 0.9415, balanced accuracy 0.9659, ROC-AUC 0.9659. The worst results are obtained by LSTM: an accuracy 0.9270, F1 0.8235, precision 0.8207, recall 0.8263, balanced accuracy 0.8897, ROC-AUC 0.8897. Notably, this model performs worse than the traditional ML methods.

Figs. 6 and 7 also show changes in loss and accuracy during the training and validation process. As can be seen in Fig. 6, the CNN and LSTM perform better than others in terms of avoiding overfitting. The training and validation values are almost matched. AE and U-Net have differences in training and validation accuracies: even with increasing training accuracy, the improvements during the validation phase are almost unchanged after 200 epochs. On the other hand, Fig. 7 proves that CNN and LSTM perform well in terms of the training and validation process: the changes of training loss correspond to the changes of validation loss. However, AE and U-Net models show that despite the loss decreasing during the training phase, it increases in the validation phase, indicating the overfitting problem.

Table 2
Results for Darknet traffic detection.

Method	Accuracy	F1	Precision	Recall	Balanced accuracy	ROC-AUC
RF	0.9738	0.9353	0.9515	0.9197	0.9538	0.9538
DT	0.9762	0.9413	0.9563	0.9267	0.9579	0.9579
k-NN	0.9698	0.9264	0.9310	0.9218	0.9520	0.9520
MLP	0.9616	0.9046	0.9257	0.8844	0.9330	0.9330
LR	0.8862	0.7707	0.6589	0.9282	0.9018	0.9018
LSTM	0.9270	0.8235	0.8207	0.8263	0.8897	0.8897
CNN	0.9747	0.9380	0.9479	0.9284	0.9576	0.9576
AE	0.9792	0.9486	0.9648	0.9330	0.9621	0.9621
U-Net	0.9803	0.9516	0.9618	0.9415	0.9659	0.9659
Karagöl, H, et al. [32]	0.9722	**0.97**	**0.97**	**0.97**	–	–
DeepImage [36]	0.94	–	–	–	–	–
DarkDetec [37]	–	0.96	**0.97**	0.95	–	–
Proposed	**0.9819**	0.9556	0.9663	0.9452	**0.9683**	**0.9683**

Table 3
Results for network traffic categorization.

Method	Accuracy	F1	Precision	Recall	Balanced accuracy	ROC-AUC
RF	0.8521	0.7421	0.7982	0.7238	0.7238	0.8510
DT	0.8310	0.7429	0.7585	0.7322	0.7322	0.8537
MLP	0.8146	0.6677	0.7877	0.6674	0.6674	0.8204
k-NN	0.8562	0.7760	0.7779	**0.7750**	**0.7750**	**0.8772**
LR	0.7082	0.6013	0.6067	0.6334	0.6334	0.7959
LSTM	0.8183	0.6994	0.7415	0.6870	0.6870	0.8302
CNN	0.8714	0.7815	0.8098	0.7693	0.7693	0.8754
AE	0.8691	0.7769	0.8119	0.7648	0.7648	0.8730
U-Net	0.8687	0.7741	0.8120	0.7586	0.7586	0.8698
Karagöl, H, et al. [32]	0.8599	0.86	0.87	0.86	–	–
DeepImage [36]	0.86	0.86	0.86	0.86	–	–
FedForest [35]	0.8676	–	–	–	–	–
Proposed	**0.8727**	**0.7829**	**0.8147**	0.7699	0.7699	0.8758

6.2. Results for proposed model

On the other hand, the results for the proposed UNet++ show that the model performs better: accuracy 0.9819, F1 0.9556, precision 0.9663, recall 0.9452, balanced accuracy 0.9683, and ROC-AUC 0.9683. The confusion matrix in Fig. 5 for UNet++ shows that the model can identify 99.14% of normal samples and 94.52% Darknet samples correctly and mistakenly identified 5.48% samples as normal and 0.86% as Darknet.

It is worth noticing that accuracy and balanced accuracy for all evaluated models are different only for 2%. It indicates that models are not overfitted, and the proposed methodology of feature processing is suitable for this case. Also, Figs. 6 and 7 show that the proposed model generally performs well without overfitting since the loss decreases and accuracy increases. However, during the validation phase, it can be noticed that accuracy has achieved some high values but stopped improving. However, accuracy in the training phase continued to increase. The same issue can be seen in graphs with loss values. The possible problem is the limited capabilities of the model, which leads to overfitting. In this case, the model can perform well, even achieve the best results according to the metrics, but it can show much worse results during the validation and testing phase. In some cases, the possible problem can be in the dataset, for example, the small size of the dataset.

This objective evaluation proves that utilizing a more complex architecture of the NN is more suitable for this kind of dataset and can extract important features more efficiently.

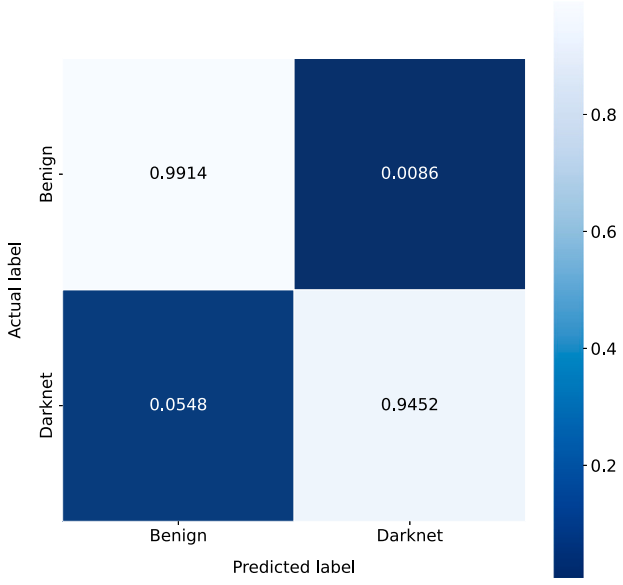


Fig. 5. Confusion matrix for binary classification with the proposed model.

7. Network traffic categorization results

This section provides results for the second part of the experiment: network traffic categorization. Similar to the previous part, Darknet traffic detection, this part compares the proposed model with baseline models. Since this task also suffers from a lack of available source codes, a wide range of experiments were conducted to ensure a fair evaluation. The summary of results can be found in Table 3 and Fig. 12.

7.1. Results for the baseline

For this task, the most successful ML method is *k*-NN, which achieved accuracy 0.8562, F1 0.7760, recall 0.7750, balanced accuracy 0.7750, and ROC-AUC 0.8772. However, the important point is worth noting. These results are achieved with a number of neighbors of 2.

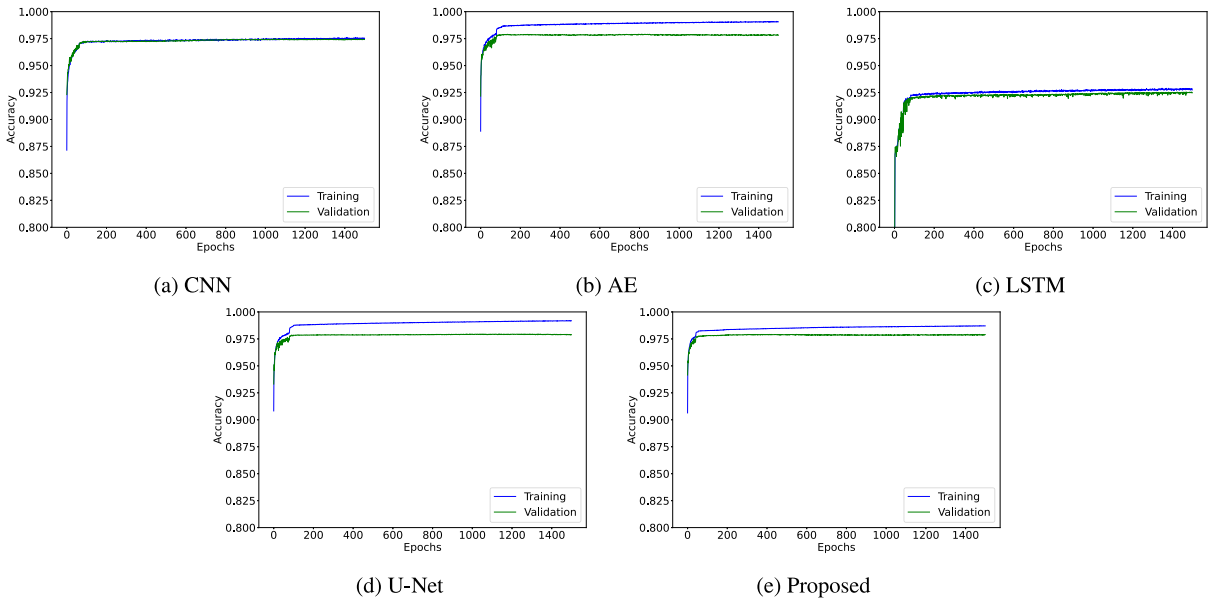


Fig. 6. Training and validation accuracy for each algorithm (Darknet traffic detection). The best possible result is 1.

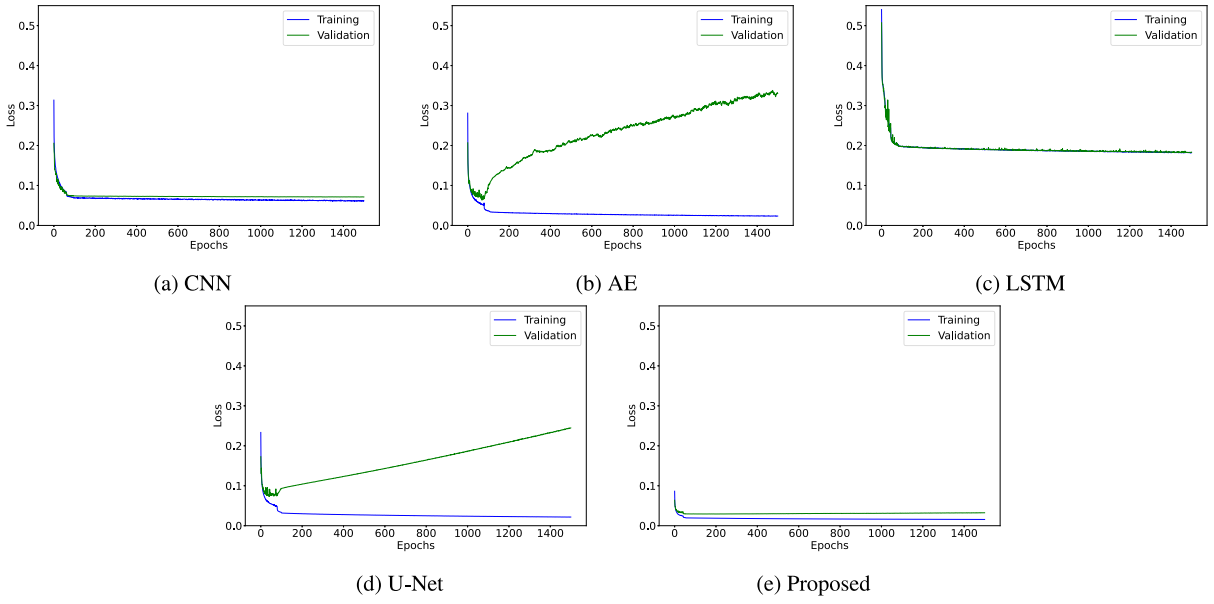


Fig. 7. Training and validation loss for each algorithm (Darknet traffic detection). The best possible result is 0.

Consequently, the model's decision can be based on some particular cases, which indicates the overfitting problem.

The other ML algorithms also performed well. For example, RF and DT achieved relatively close results regarding accuracy, F1, recall, and balanced accuracy. The advantages of these models have been already mentioned in the previous section. However, according to the best combination of hyper-parameters, the problem is similar to the case of Darknet traffic detection: the minimum number of samples is 1, and the max depth is 14, which indicates that some particular cases were considered and because of many conditions the accurate results achieved. Here, the problem of generalization can raised, and, similarly, can indicate the overfitting problem.

Like Darknet traffic detection, the LSTM achieves worse results than other models. On the other hand, CNN shows better results than AE and U-Net models: accuracy 0.8714, F1 0.7815, recall 0.7693, balanced accuracy 0.7693. AE and U-Net models achieved similar results with very small differences.

According to Fig. 9, the CNN and LSTM during training and validation phases provide almost similar results for accuracy. The same situation is in Fig. 10. It can be seen that loss decreases in the training and validation phases. The situation is different with AE and U-Net models. Although the accuracy grows in the training phases, it keeps an almost constant value in the validation phase and does not increase. The loss decreases in the training phase. However, it tends to increase in the validation phase.

7.2. Results for the proposed model

Furthermore, the results of UNet++ are also promising and outperform other evaluated approaches. The accuracy 0.8727, F1 0.7829, precision 0.8147, recall 0.7699, balanced accuracy 0.7699, and ROC-AUC 0.8758.

The confusion matrix is shown in Fig. 8. According to it, the most problematic classes for detection are Email and VoIP. Instead of

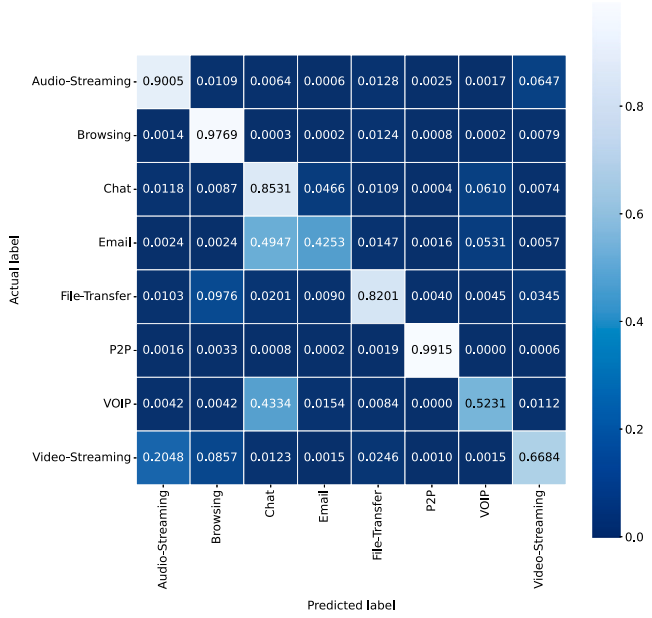


Fig. 8. Confusion matrix for multiclass classification with UNet++.

Table 4
Detailed numerical results of UNet++.

Category	Precision	Recall	F1
Audio-Streaming	0.87	0.9	0.89
Browsing	0.93	0.98	0.95
Chat	0.66	0.85	0.74
Email	0.78	0.43	0.55
File-Transfer	0.89	0.82	0.85
P2P	0.99	0.99	0.99
VoIP	0.62	0.52	0.57
Video-Streaming	0.77	0.67	0.71

correctly detecting Email, the model categorizes the samples as Chat. Alternatively, instead of VoIP class, it indicates them as Chat too. The model also has the same problem with Video and Audio Streaming. It is not very clear to decide between these categories. Despite that, the proposed model has a relatively good detection capability for the following traffic categories: Audio streaming, Browsing, Chat, File Transfer, P2P, and Video Streaming.

The more detailed results per each class are introduced in Table 4. The results correspond with the confusion matrix in Fig. 8. The model has the least number of mistakes for the P2P category, and the metrics, such as precision, recall, and F1, have almost ideal results, which is 0.99. The same situation applies to the browsing, audio streaming, and file transfer categories. In most cases, the model correctly identified it, and values of recall, precision, and F1 are above 0.80. VoIP class has worse results than others: recall 0.52, precision 0.62, and F1 0.57.

The training and validation process for the proposed model is also introduced in Figs. 9 and 10. The accuracy was achieved very fast at 0.85 in the validation phase, and almost the same values were kept during the training process. Contrary to this, the training accuracy kept improving during the whole training. The loss changes are also correct in the case of training, but they tend to increase during validation. It can signal about overfitting, and, consequently, there is a need to correct the model's hyper-parameters and, probably, experiment with different optimizers, which would be more suitable for this task. Notably, only LSTM has the training and validation process correctly: with progressive improvements and without overfitting problem.

In this case, the proposed model is more efficient for traffic categorization than traditional ML methods and other DL architectures. Here, it was also proved that using a more complex network can more accurately predict the type of application. Additionally, the comparison with an original U-Net proves that convolutional blocks in skip connections, which transfer the extracted features between levels, are useful in this field of research.

8. Discussion

This work conducts Darknet traffic detection and categorization experiments based on a semantic segmentation approach. As the first step, the traditional ML methods were trained with hyper-parameters optimization and evaluated. According to the results, the RF and DT perform better than other ML algorithms for Darknet traffic detection. Considering the nature of this algorithm, it has a good potential to be applied in some expert systems. On the other hand, the resulting interpretative composition of DTs should be studied carefully since there is always a possibility that high results are achieved thanks to some particular or single cases. Consequently, in the future, the model's efficiency can be decreased on the new samples. This behavior can be addressed as an overfitting problem or a problem with generalization. It can have a very large impact on real-world applications.

Another point is applying the ML method for traffic categorization. The results show that the *k*-NN algorithm outperforms the other. On the one hand, the results are promising, but this performance was achieved with a number of neighbors of 2. This point indicates that the algorithm was not generalized but used just a couple of samples to determine the class. This problem can also be defined as the overfitting of the model.

Despite the mentioned problem of a lack of source code and the impossibility of reproducing the experiments with other approaches, a general comparison with existing works is provided and included in the Tables 2 and 3.

Our work outperforms the recent approach [32], achieving an accuracy of 97.22% for Darknet traffic detection and 85.99% for traffic categorization using the RF algorithm. The authors applied SMOTE to eliminate the dataset imbalance.

Compared with DL architecture [37], CNN-LSTM, our model also performs better. The mentioned approach achieved for classification of 4 classes (Tor, Non-Tor, VPN and Non-VPN) a precision 0.97, recall 0.95, F1 0.96. It can be challenging to compare with our results since the authors provided results rounded to hundredths. Here, the use of LSTM layers can be time-consuming and computation-demanding.

The authors of the used dataset [36] conducted an experiment using 2D CNN and achieved an accuracy of 94% for binary classification and 86% for multiclass classification. It can also be concluded that our model is more accurate.

Also, compared with the work [35], which used the combination of Gradient Boosting DT and Federated Learning framework, the U-Net models give better results. The mentioned approach achieved an accuracy of 86.76% on the Darknet dataset, which is less than CNN and Unet++, which can achieve 87.14% and 87.27%, respectively, according to our experiments. According to our comparison, the proposed architecture is competitive with existing approaches and outperforms them.

The next arising question is whether to apply ML or DL model. The main advantage of ML models is the fast processing compared with DL. Notably, the ML models are mainly launched on Central processing unit (CPU), and DL models can be launched on CPU and Graphics processing unit (GPU), but GPU will be preferable to keep performance and efficiency. However, it should be considered that the model's processing time depends on the given model's complexity. For example, *k*-NN is more time-consuming than others. On the other hand, the processing time of ML/DL models depends on the used hardware. This way, the ML and DL models can be comparable in time.

On the other hand,ML methods require more attention to data preprocessing, as the algorithm's success hinges on this step. In contrast,DL models offer a more flexible approach. Feature selection, for instance, is optional for DL models, as they can process the input data and extract

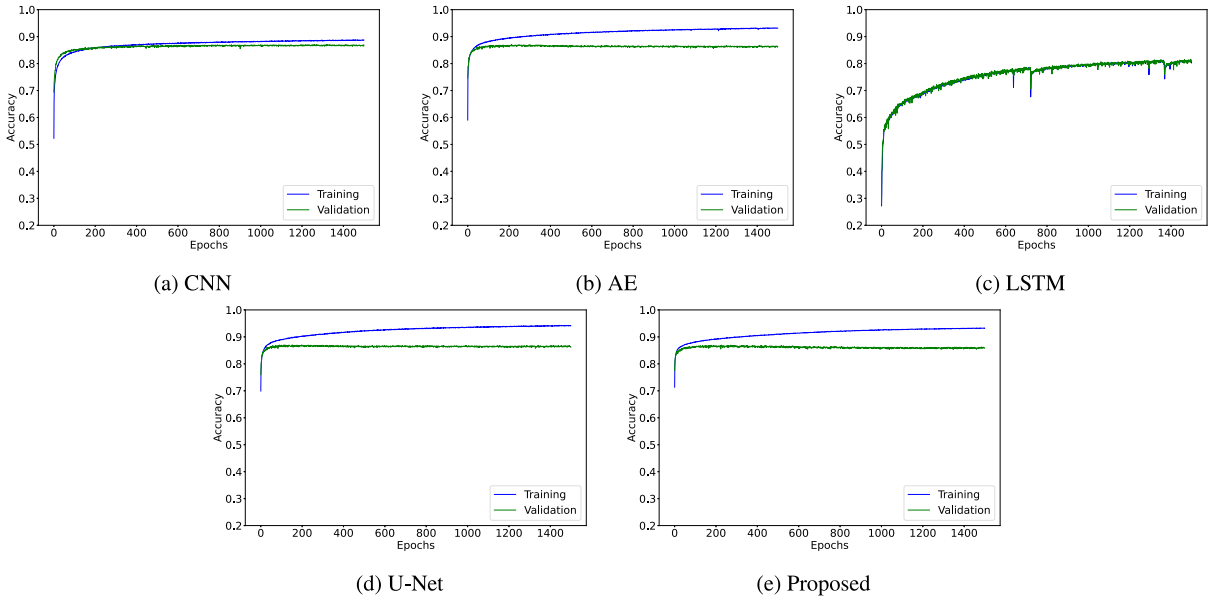


Fig. 9. Training and validation accuracy for each algorithm (Network traffic categorization). The best possible result is 1.

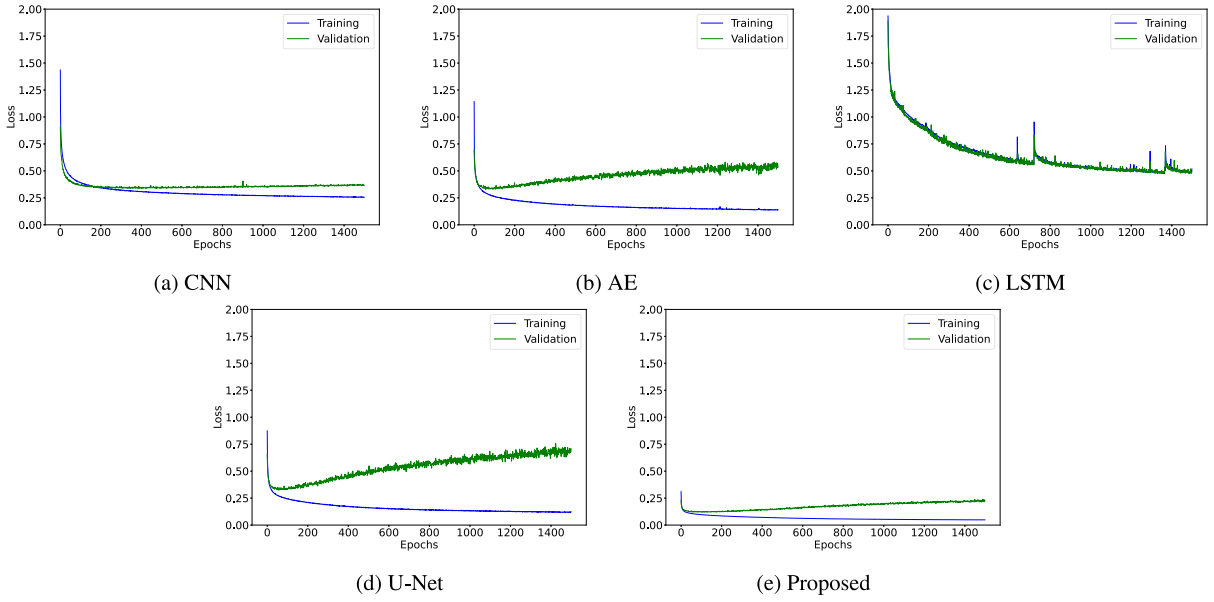


Fig. 10. Training and validation loss for each algorithm (Network traffic categorization). The best possible result is 0.

the necessary information. This adaptability is one of the intriguing advantages of the DL approach.

However, explaining the decision of 'black-box' models is a challenge that needs to be addressed. It is crucial to develop approaches that can explain the output of these models. This understanding is not only beneficial for security experts to evaluate the correctness of the algorithm's work but also for the broader research community to advance our knowledge in this field.

The proposed model can still be improved. Since the experiments show the effectiveness of skip connections between different layers, it is also possible to transfer extracted information not only from the bottom layers to the top but also in opposite directions. Another improvement is modifying ConvBlock in the model. For example, extra layers should be added for regularization or dropouts to prevent overfitting problems. It can also be applied here since the new Lion optimizer shows its efficiency in this field of research.

Here, how to work with Darknet traffic efficiently can also be investigated. One of the ways is to apply two-level classification, as was done in [53]. Firstly, the normal traffic can be filtered accurately, and the rest, labeled as Darknet, can be investigated with a model trained for traffic categorization. It can decrease the number of evaluated samples and make it more applicable in the real world, resulting in less computation demand and decreased processing time.

The summary of detected challenges and possible solutions is introduced in Table 5.

9. Future directions

This work has provided a comparison of ML and DL models via testing models under similar conditions and dataset split. On the other hand, several challenges and possible improvements of the proposed

Table 5
Detected challenges and possible solutions.

Challenge	Description	Potential solution	Ref.
Feature preprocessing	Traditional ML methods require extensive feature engineering to achieve high performance.	DL models are able to extract features during the training phase from complex data.	[11]
ML overfitting	Small number of neighbors for *k*-NN can lead to overfitting of the model	Application of optimization approaches, such as smoothing, minimum-cost *k*-value selection, feature selection, and ensemble selection	[50]
Explanation of DL's decision	"Black-box" nature of NNs makes it difficult to understand the reasons for the algorithm's decision.	Utilization of advanced techniques for interpretation, for example, Deep SHAP.	[51]
Regularization in DL	The model's performance can be significantly improved with the application of different regularization techniques.	Find optimal combination regularization techniques using a joint optimization over the decision on which regularizers to apply and their subsidiary hyper-parameters	[52]
Optimization of Darknet traffic categorization	It is required to categorize only Darknet traffic.	Use two-level classification, which is firstly filter Darknet traffic and after that categorize the traffic according to the application	[53]

Table 6
Summary of approaches for encrypted traffic detection (sorted by the publication year). See Appendix A.

Ref.	Year	Main idea	Used dataset	Used technique
[19]	2019	Analysis of encrypted network traffic features using ML	CTU-13 dataset, Malware Capture Facility Project dataset	SVM, RF, XGB
[20]	2021	The method combines ML, DL and natural language processing for detection of malicious traffic encryption methods	Private dataset	TF-IDF, ML ensemble model, CNN
[21]	2021	An encrypted malicious traffic is detected by ResNet. A network threat generation is based on *Q*-learning and GAN.	CTU-Mixed-Capture datasets	ResNet, *Q*-learning and GAN
[22]	2021	A cost matrix is utilized to deal with unbalanced dataset and to assign the weight to each class based on the distribution	ISCX VPN-nonVPN dataset	Cost-Sensitive CNN
[23]	2021	The method is aimed to identify encrypted traffic. Inception NN extracts local information. LSTM extracts temporal features.	ISCX 2016 dataset	Inception-LSTM
[24]	2021	Method is based on transformation of flow data into a picture with following classification using CNN	ISCX-VPN, ISCX Tor-nonTor dataset	CNN
[25]	2021	The lightweight approach is proposed and utilizes CNN and multi-head-attention	Open HTTPS Dataset, CTU-13, private dataset	CNN, multi-head-attention
[26]	2021	Method consists of preprocessing and classification phases. CNN extracts spatial features and bidirectional LSTM extracts temporal characteristics.	ISCXTor2016	CNN, LSTM
[27]	2021	The novel feature engineering approach is proposed for encrypted web protocols. The CNN-LSTM based NN is proposed.	Orange'20 Dataset, UC Davis QUIC Dataset	CNN, LSTM
[28]	2022	Feature extraction is performed with CNN. Feature selection with ant-lion metaheuristic algorithm. Traffic is classified using a fuzzy-SOM based-clustering.	ISCX VPN-non-VPN	CNN, ant-lion metaheuristic algorithm, self-organizing map
[29]	2023	The proposed model combines CNN and RL. The input format of data is an image.	ISCXVPN2016, ISCXTor2017	CNN, RL
[30]	2023	The approach is basedon ViT and augmentation with Bidirectional GAN.	ISCXTor2016	ViT, BiGAN

model are also detected (see Table 5). As a consequence, several future directions are defined for our following work.

Firstly, creating an updated dataset with Darknet traffic corresponding to the actual state is necessary. Since the number of cybercrimes is rising rapidly, this point is critical for the cyber security field.

Secondly, this work shows that traditional ML algorithms can achieve good results, but with a high probability, the overfitting problem appears. In this case, the future direction is to focus on DL models, which can also outperform results with good generalization. As the first step, the DL model was proposed. Unlike previous related works, this work considers semantic segmentation and gives promising results. However, there is a space for further improvements, for example, testing different optimization techniques (optimizers, loss functions, etc.) or extending the model with attention modules, which would better capture spatial and temporal features of given traffic.

Concerning application for real-world applications, it is required to develop the methodology for the explanation of NN's decision. Despite existing techniques, they can be inefficient due to time-consuming analysis caused by the model's complexity. This is why this field needs to be studied in more detail.

Lastly, it is important to make the developed methodology available for real-world application. Because of that, the possible future direction is integrating it into the existing system.

Table 7
p-value test for Darknet traffic detection. See Appendix B.

Feature	*p*-value	Feature	*p*-value	Feature	*p*-value
Fwd PSH Flags	0.00E+00	Subflow Fwd Bytes	2.42E−57	Down/Up Ratio	1.27E−08
Packet Length Mean	0.00E+00	ACK Flag Count	1.11E−56	PSH Flag Count	1.35E−08
FIN Flag Count	0.00E+00	FWD Init Win Bytes	3.23E−43	Bwd IAT Min	4.46E−08
RST Flag Count	0.00E+00	Bwd Packet Length Max	2.91E−37	Bwd Packet Length Mean	1.46E−07
Average Packet Size	0.00E+00	Fwd Packet Length Max	1.10E−35	Bwd Segment Size Avg	1.46E−07
Subflow Fwd Packets	0.00E+00	Packet Length Max	3.36E−32	Idle Mean	1.33E−06
Fwd Seg Size Min	0.00E+00	Fwd IAT Min	7.73E−29	Bwd Header Length	3.70E−06
Src Port	1.30E−235	Fwd IAT Std	2.09E−27	Idle Std	1.51E−05
Bwd Packet Length Min	9.72E−214	Fwd Packets/s	3.84E−27	Fwd IAT Max	2.41E−05
Flow Packets/s	1.82E−204	Total Bwd packets	1.38E−24	Packet Length Min	1.68E−04
SYN Flag Count	4.32E−189	Flow IAT Std	1.08E−23	Flow IAT Max	2.82E−04
Bwd Init Win Bytes	1.28E−150	Fwd Packet Length Min	2.27E−22	Flow IAT Mean	3.27E−04
Fwd Segment Size Avg	8.71E−133	Total Fwd Packet	1.47E−17	Flow Duration	1.13E−03
Fwd Packet Length Mean	8.71E−133	Total Length of Bwd Packet	2.09E−17	Fwd Act Data Pkts	8.05E−03
Subflow Bwd Bytes	2.40E−132	Fwd IAT Mean	6.92E−16	Bwd Bulk Rate Avg	9.63E−03
Dst Port	1.69E−120	Idle Min	4.77E−14	Bwd IAT Mean	1.17E−02
Bwd Packet Length Std	4.01E−105	Flow IAT Min	3.66E−13	Packet Length Std	1.19E−02
Total Length of Fwd Packet	5.38E−92	Bwd IAT Total	1.56E−11	Flow Bytes/s	1.54E−02
Bwd Packets/s	1.03E−86	Bwd IAT Std	2.71E−11	Packet Length Variance	4.46E−02
Fwd Packet Length Std	1.88E−69	Bwd IAT Max	1.66E−09	Bwd Packet/Bulk Avg	1.08E−01
Fwd Header Length	8.13E−58	Idle Max	2.25E−09	Fwd IAT Total	2.73E−01

Table 8
p-value test for network traffic categorization. See Appendix B.

Feature	*p*-value	Feature	*p*-value	Feature	*p*-value
Src Port	0.00E+00	Bwd Packets/s	9.76E−19	Bwd IAT Std	1.16E−04
Dst Port	0.00E+00	Bwd Header Length	1.67E−18	Down/Up Ratio	5.28E−04
Fwd PSH Flags	0.00E+00	Total Length of Fwd Packet	1.88E−17	Bwd Packet Length Mean	6.96E−04
FIN Flag Count	0.00E+00	Total Fwd Packet	1.32E−14	Bwd Segment Size Avg	6.96E−04
RST Flag Count	0.00E+00	Idle Min	1.60E−13	Fwd Header Length	1.42E−03
Subflow Fwd Packets	0.00E+00	FWD Init Win Bytes	1.79E−13	Packet Length Variance	1.62E−03
Fwd Seg Size Min	0.00E+00	Flow IAT Std	2.07E−13	Bwd Packet Length Std	1.98E−03
Packet Length Min	6.14E−148	Fwd Packet Length Mean	2.17E−13	Bwd Init Win Bytes	2.44E−03
Flow Packets/s	2.74E−118	Fwd Segment Size Avg	2.17E−13	Bwd Packet Length Max	5.58E−03
Bwd IAT Total	5.79E−72	Bwd Packet Length Min	2.42E−13	Flow IAT Max	8.95E−03
Fwd Packet Length Min	8.48E−60	ACK Flag Count	5.08E−12	Bwd IAT Max	1.03E−02
PSH Flag Count	5.24E−55	Packet Length Std	5.43E−12	Fwd IAT Max	1.25E−02
SYN Flag Count	5.54E−51	Flow Duration	3.57E−08	Average Packet Size	1.29E−02
Flow IAT Mean	1.40E−43	Fwd Packet Length Max	4.43E−08	Fwd IAT Mean	2.44E−02
Idle Mean	1.57E−43	Packet Length Mean	3.47E−07	Fwd IAT Std	2.09E−01
Fwd Packet Length Std	2.10E−43	Bwd IAT Mean	8.46E−07	Fwd Act Data Pkts	2.24E−01
Idle Max	5.16E−38	Bwd Bulk Rate Avg	3.54E−06	Bwd Packet/Bulk Avg	3.42E−01
Subflow Fwd Bytes	3.99E−34	Fwd IAT Total	3.79E−06	Fwd Packets/s	4.46E−01
Subflow Bwd Bytes	5.25E−30	Flow Bytes/s	1.52E−05	Fwd IAT Min	5.93E−01
Flow IAT Min	1.73E−21	Bwd IAT Min	5.02E−05	Packet Length Max	7.52E−01
Idle Std	2.49E−19	Total Bwd packets	6.03E−05	Total Length of Bwd Packet	8.19E−01

10. Conclusion

With technological advancement, cybersecurity remains a very active and critical field of research. With the growing amount of information transferred, the risk of malicious behavior is also increasing. Consequently, controlling and monitoring traffic is becoming more difficult, especially in the darker areas of the World Wide Web.

This work proposes the DL semantic segmentation-based method, which aims to improve the systems' capabilities to detect Darknet traffic and categorize for mitigating illegal activities and preventing cyber security incidents. Our approach is based on utilizing UNet++ architecture, modified the processing of 1D data.

This work has proved that a more complex structure and optimized network traffic processing are required in this research field. We have trained and tested several ML and DL models here. According to the results, the proposed model more accurately classifies the traffic. Darknet detection has an accuracy of 0.9819, and traffic categorization has an accuracy of 0.8727, outperforming other evaluated models.

To conclude the work, it is worth mentioning that developing DL models in this field of information technologies is an essential part of the research. It is capable of better feature extraction and processing large amounts of data. As the extension of this work, analysis of the feature's importance can be performed not just for ML methods but also for DL models.

Acronyms

AE Autoencoder
AI Artificial intelligence
BIE Binary Image Encoding
Bi-LSTM Bidirectional Long Short-Term Memory
CNN Convolutional Neural Network
CPU Central processing unit
DL Deep Learning
DT Decision Tree
GAN Generative Adversarial Network
GELU Gaussian Error Linear Unit
GPU Graphics processing unit
IDS Intrusion Detection System
IoT Internet of Things
***k*-NN** *k*-Nearest Neighbours
LR Logistic Regression
LSTM Long Short-Term Memory

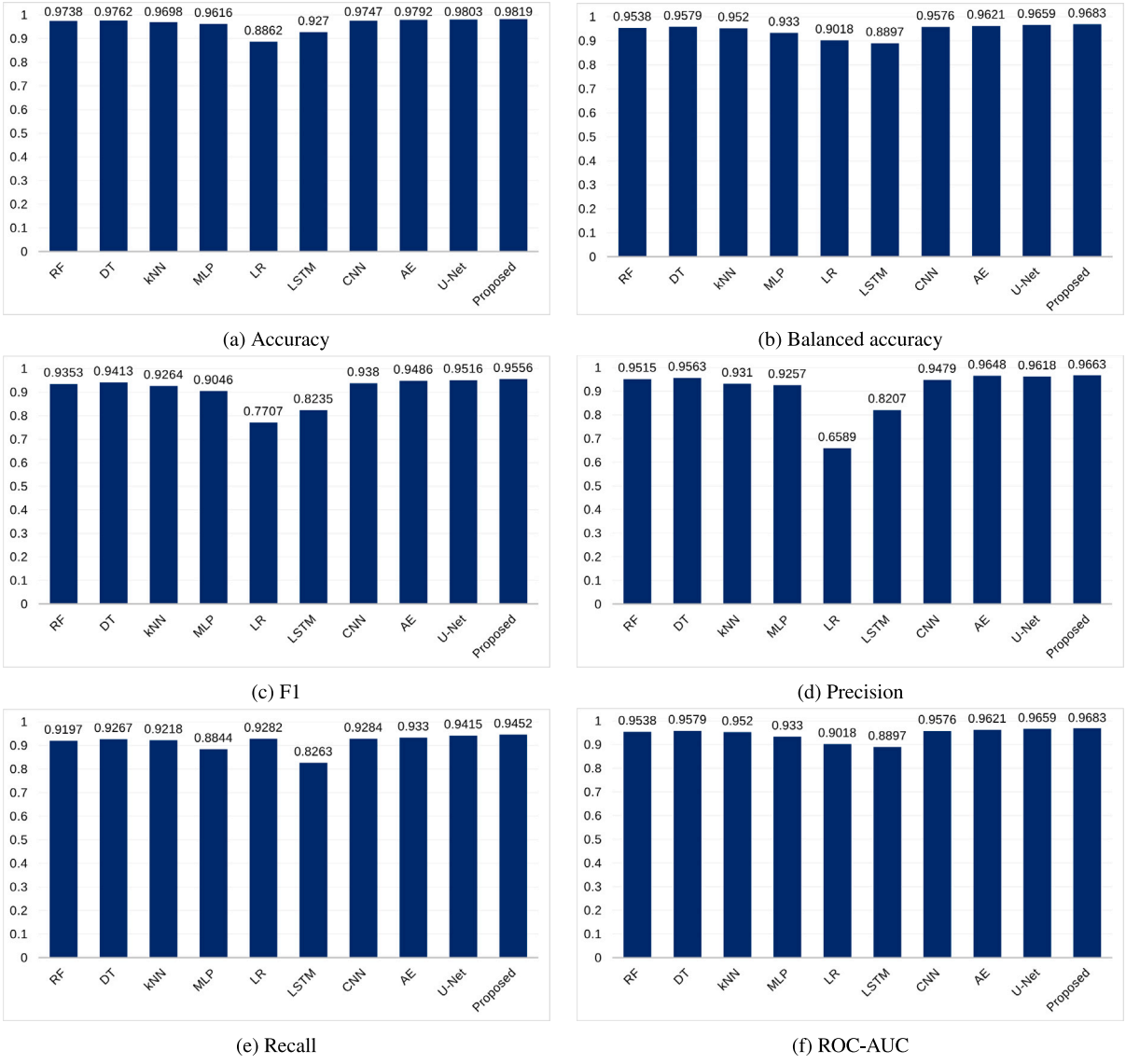


Fig. 11. Barplot with results for Darknet traffic detection. See Appendix C.

ML Machine Learning
MLP Multilayer Perceptron
NIDS Network Intrusion Detection System
NN Neural Network
P2P Peer-to-Peer
PCA Principal Component Analysis
RF Random Forest
RL Reinforcement Learning
RNN Recurrent Neural Network
SMOTE Synthetic Minority Over-sampling Technique
SVM Support Vector Machines
TF-IDF Term Frequency - Inverse Document Frequency
Tor The Onion Router
ViT Vision Transformer
VoIP Voice over Internet Protocol
VPN Virtual Private Network
XGB XGBoost

Funding

This work was supported by the Technology Agency of the Czech Republic (TACR) under the grant with a number CK04000027.

CRediT authorship contribution statement

Anzhelika Mezina: Writing – original draft, Visualization, Validation, Methodology, Conceptualization. **Radim Burget:** Writing – review & editing, Formal analysis. **Aleksandr Ometov:** Writing – review & editing, Validation, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

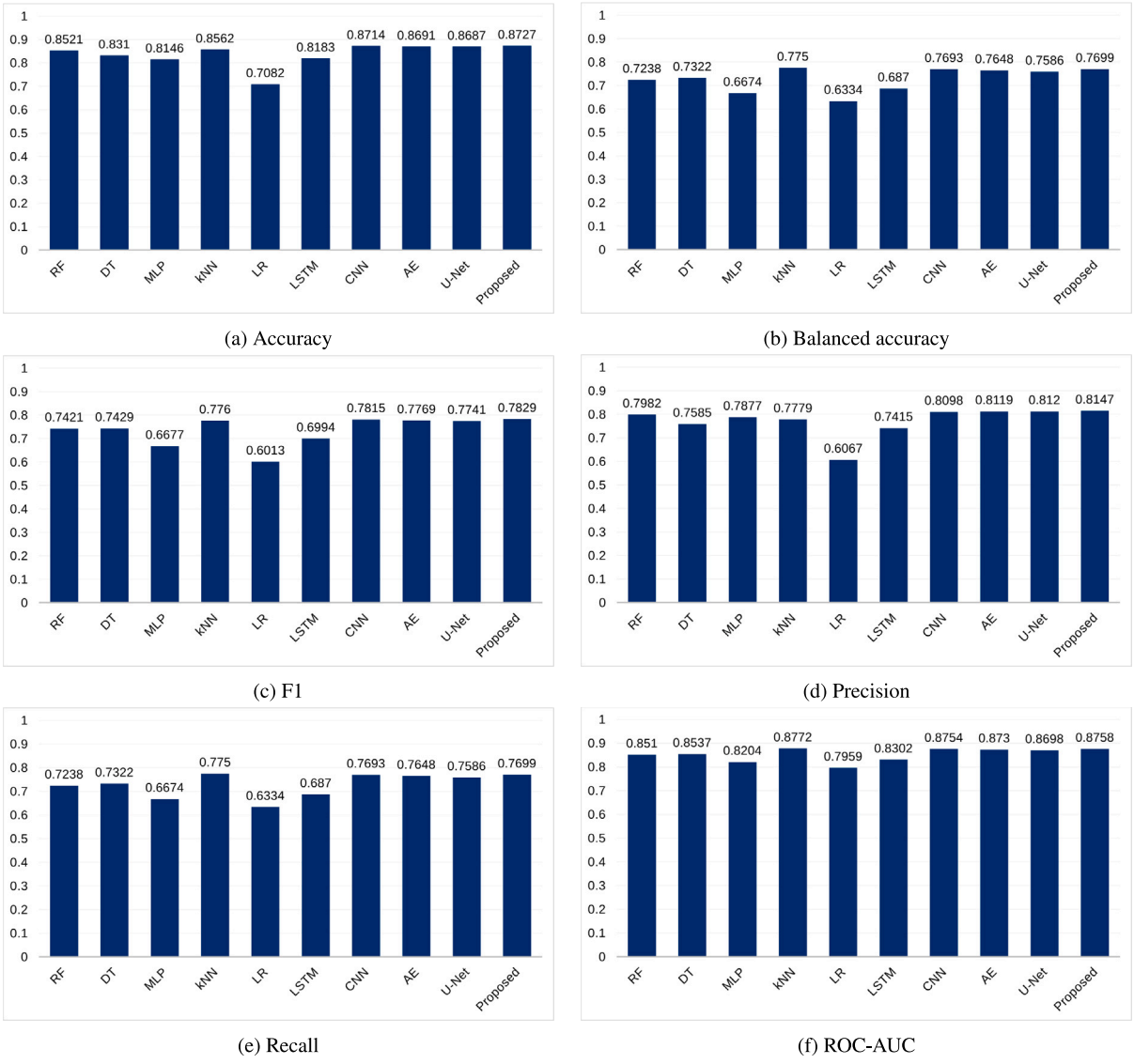


Fig. 12. Barplot with results for traffic categorization. See Appendix C.

Appendix A. Summary of related works related to encrypted traffic detection

Table 6 represents the summary of works introduced in Section 2.1.

Appendix B. Statistically important features

Tables 8 and 8 show detailed results for the *p*-value test for each feature from the CIC-Darknet2020 dataset for both tasks, i.e., Darknet traffic detection and traffic categorization. More detailed information is in Section 3.

Appendix C. Additional graphical results representation

Figs. 11 and 12 depict bars with the comparison of achieved results using different metrics for conducted experiments, i.e., Darknet traffic detection and traffic categorization. The introduced graphs correspond to Tables 2 and 3.

References

[1] D. Silhavy, D. Waring, D. Audsin, R. Bradbury, J. Mika, K. Kuehnhammer, K. Krauss, J.J. Gimenez, 3GPP rel-17 5G media streaming and 5G broadcast powered by 5G-MAG reference tools, in: Proceedings of the 2nd Mile-High Video Conference, 2023, pp. 85–90.

[2] J. Lan, X. Liu, B. Li, Y. Li, T. Geng, DarknetSec: A novel self-attentive deep learning method for Darknet traffic classification and application identification, Comput. Secur. 116 (2022) 102663.

[3] R.B. Zeid, J. Moubarak, C. Bassil, Investigating the Darknet, in: Proceedings of International Wireless Communications and Mobile Computing, IWCMC, IEEE, 2020, pp. 727–732.

[4] EMCDDA, Drugs and the Darknet: Perspectives for Enforcement, Research and Policy, 2017, https://www.emcdda.europa.eu/publications/joint-publications/drugs-and-the-darknet_en (Accessed 01 March 2024).

[5] N. Dutta, N. Jadav, S. Tanwar, H.K.D. Sarma, E. Pricop, N. Dutta, N. Jadav, S. Tanwar, H.K.D. Sarma, E. Pricop, DarkNet and hidden services, in: Cyber Security: Issues and Current Trends, Springer, 2022, pp. 57–69.

[6] chainalysis, The Chainalysis 2023 Crypto Crime Report, 2023, https://go.chainalysis.com/2023-crypto-crime-report.html (Accessed 19 November 2023).

[7] Q. Abu Al-Haija, M. Krichen, W. Abu Elhaija, Machine-learning-based Darknet traffic detection system for IoT applications, Electronics 11 (4) (2022) 556.

[8] M. Douiba, S. Benkirane, A. Guezzaz, M. Azrour, An improved anomaly detection model for IoT security using decision tree and gradient boosting, J. Supercomput. 79 (3) (2023) 3392–3411.
[9] M. Płachta, M. Krzemień, K. Szczypiorski, A. Janicki, Detection of image steganography using deep learning and ensemble classifiers, Electronics 11 (10) (2022) 1565.
[10] M. Dener, G. Ok, A. Orman, Malware detection using memory analysis data in big data environment, Appl. Sci. 12 (17) (2022) 8604.
[11] Y. Zeng, H. Gu, W. Wei, Y. Guo, *Deep – Full – Range*: A deep learning based network encrypted traffic classification and intrusion detection framework, IEEE Access 7 (2019) 45182–45190.
[12] Y.-C. Wang, Y.-C. Houng, H.-X. Chen, S.-M. Tseng, Network anomaly intrusion detection based on deep learning approach, Sensors 23 (4) (2023) 2171.
[13] A. Mezina, R. Burget, C.M. Travieso-González, Network anomaly detection with temporal convolutional network and U-Net model, IEEE Access 9 (2021) 143608–143622.
[14] Y. Mo, Y. Wu, X. Yang, F. Liu, Y. Liao, Review the state-of-the-art technologies of semantic segmentation based on deep learning, Neurocomputing 493 (2022) 626–646.
[15] B.D. Setiawan, M. Kovacs, U. Serdült, V. Kryssanov, Semantic segmentation on smartphone motion sensor data for road surface mmonitoring, Procedia Comput. Sci. 204 (2022) 346–353.
[16] K. Duraj, N. Piaseczna, P. Kostka, E. Tkacz, Semantic segmentation of 12-lead ECG using 1D residual U-Net with squeeze-excitation blocks, Appl. Sci. 12 (7) (2022) 3332.
[17] X. Hou, X. Wang, Y. Hu, Y. Chen, G. Huang, S. Nie, A one-dimensional U-net-based calibration-transfer method for low-field nuclear magnetic resonance signals, Anal. Chem. 93 (30) (2021) 10469–10476.
[18] Z. Guo, C. Ding, X. Hu, C. Rudin, A supervised machine learning semantic segmentation approach for detecting artifacts in plethysmography signals from wearables, Physiol. Meas. 42 (12) (2021) 125003.
[19] A.S. Shekhawat, F. Di Troia, M. Stamp, Feature analysis of encrypted malicious traffic, Expert Syst. Appl. 125 (2019) 130–141.
[20] H. Yang, Q. He, Z. Liu, Q. Zhang, Malicious encryption traffic detection based on NLP, Secur. Commun. Netw. 2021 (2021) 1–10.
[21] J. Yang, G. Liang, B. Li, G. Wen, T. Gao, A deep-learning-and reinforcement-learning-based system for encrypted network malicious traffic detection, Electron. Lett. 57 (9) (2021) 363–365.
[22] S. Soleymanpour, H. Sadr, M. Nazari Soleimandarabi, CSCNN: Cost-sensitive convolutional neural network for encrypted traffic classification, Neural Process. Lett. 53 (5) (2021) 3497–3523.
[23] B. Lu, N. Luktarhan, C. Ding, W. Zhang, ICLSTM: Encrypted traffic service identification based on inception-LSTM neural network, Symmetry 13 (6) (2021) 1080.
[24] T. Shapira, Y. Shavitt, FlowPic: A generic representation for encrypted traffic classification and applications identification, IEEE Trans. Netw. Serv. Manag. 18 (2) (2021) 1218–1232.
[25] J. Cheng, Y. Wu, E. Yuepeng, J. You, T. Li, H. Li, J. Ge, MATEC: A lightweight neural network for online encrypted traffic classification, Comput. Netw. 199 (2021) 108472.
[26] K. Lin, X. Xu, H. Gao, TSCRNN: A novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of IIoT, Comput. Netw. 190 (2021) 107974.
[27] I. Akbari, M.A. Salahuddin, L. Ven, N. Limam, R. Boutaba, B. Mathieu, S. Moteau, S. Tuffin, A look behind the curtain: Traffic classification in an increasingly encrypted web, Proceedings of the ACM on Measurement and Analysis of Computing Systems (POMACS) 5 (1) (2021) 1–26.
[28] S. Izadi, M. Ahmadi, R. Nikbazm, Network traffic classification using convolutional neural network and ant-lion optimization, Comput. Electr. Eng. 101 (2022) 108024.
[29] R. Moreira, L.F.R. Moreira, F. de Oliveira Silva, An intelligent network monitoring approach for online classification of Darknet traffic, Comput. Electr. Eng. 110 (2023) 108852.
[30] Y. Sanjalawe, S. Fraihat, et al., Detection of obfuscated tor traffic based on bidirectional generative adversarial networks and vision transform, Comput. Secur. (2023) 103512.
[31] L.A. Iliadis, T. Kaifas, Darknet traffic classification using machine learning techniques, in: Proceedings of 10th International Conference on Modern Circuits and Systems Technologies, MOCAST, IEEE, 2021, pp. 1–4.
[32] H. Karagöl, O. Erdem, B. Akbas, T. Soylu, Darknet traffic classification with machine learning algorithms and SMOTE method, in: 2022 7th International Conference on Computer Science and Engineering, UBMK, IEEE, 2022, pp. 374–378.
[33] H. Mohanty, A.H. Roudsari, A.H. Lashkari, Robust stacking ensemble model for Darknet traffic classification under adversarial settings, Comput. Secur. 120 (2022) 102830.
[34] M.C. Marim, P.V.B. Ramos, A.B. Vieira, A. Galletta, M. Villari, R.M. de Oliveira, E.F. Silva, Darknet traffic detection and characterization with models based on decision trees and neural networks, Intell. Syst. Appl. (2023) 200199.
[35] T. Dong, S. Li, H. Qiu, J. Lu, An interpretable federated learning-based network intrusion detection framework, 2022, arXiv preprint arXiv:2201.03134.
[36] A. Habibi Lashkari, G. Kaur, A. Rahali, DIDarknet: A contemporary approach to detect and characterize the Darknet traffic using deep image learning, in: Proceedings of the 10th International Conference on Communication and Network Security, 2020, pp. 1–13.
[37] M.B. Sarwar, M.K. Hanif, R. Talib, M. Younas, M.U. Sarwar, DarkDetect: Darknet traffic detection and categorization using modified convolution-long short-term memory, IEEE Access 9 (2021) 113705–113713.
[38] N. Rust-Nguyen, S. Sharma, M. Stamp, Darknet traffic classification and adversarial attacks using machine learning, Comput. Secur. (2023) 103098.
[39] D. Singh, A. Shukla, M. Sajwan, Deep transfer learning framework for the identification of malicious activities to combat cyberattack, Future Gener. Comput. Syst. 125 (2021) 687–697.
[40] N. Briner, D. Cullen, J. Halladay, D. Miller, R. Primeau, A. Avila, R. Basnet, T. Doleck, Tabular-to-image transformations for the classification of anonymous network traffic using deep residual networks, IEEE Access (2023).
[41] C. Bachechi, F. Rollo, L. Po, Detection and classification of sensor anomalies for simulating urban traffic scenarios, Cluster Comput. 25 (4) (2022) 2793–2817.
[42] L.B. de Amorim, G.D. Cavalcanti, R.M. Cruz, The choice of scaling technique matters for classification performance, Appl. Soft Comput. 133 (2023) 109924.
[43] H. Rhys, Machine Learning with R, the Tidyverse, and MLR, Simon and Schuster, 2020.
[44] Z. Zhou, M.M. Rahman Siddiquee, N. Tajbakhsh, J. Liang, UNet++: A nested U-Net architecture for medical image segmentation, in: Proceedings of Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, 4, Springer, 2018, pp. 3–11.
[45] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in: Proceedings of Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18, Springer, 2015, pp. 234–241.
[46] D. Hendrycks, K. Gimpel, Gaussian error linear units (GELUs), 2016, arXiv preprint arXiv:1606.08415.
[47] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proc. of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.
[48] X. Chen, C. Liang, D. Huang, E. Real, K. Wang, Y. Liu, H. Pham, X. Dong, T. Luong, C.-J. Hsieh, et al., Symbolic discovery of optimization algorithms, 2023, arXiv preprint arXiv:2302.06675.
[49] M. Grandini, E. Bagli, G. Visani, Metrics for multi-class classification: an overview, 2020, arXiv preprint arXiv:2008.05756.
[50] S. Zhang, Cost-sensitive KNN classification, Neurocomputing 391 (2020) 234–242.
[51] S.M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, Adv. Neural Inf. Process. Syst. 30 (2017).
[52] A. Kadra, M. Lindauer, F. Hutter, J. Grabocka, Regularization is all you need: Simple neural nets can excel on tabular data, 2021, arXiv preprint arXiv:2106.11189. 536.
[53] A. Mezina, A. Ometov, Detecting smart contract vulnerabilities with combined binary and multiclass classification, Cryptography 7 (3) (2023) 34.

Anzhelika Mezina received her Bachelor's degree and Master's degree in Information security at the Brno University of Technology (BUT), Czech Republic in 2018 and 2020, correspondingly. Currently, she is pursuing her Doctoral Degree in Information Security at BUT. The focus of her research is mainly leaning towards developing and applying Deep Learning methods for various real-world scenarios. Currently, she is involved in national-level funded research projects focusing on security and the medical fields (in cooperation with the Palacky University Olomouc and the Ministry of Interior of the Czech Republic). Her research interests are deep learning, information security, anomaly detection, computer vision, and image processing.

Radim Burget received the Ph.D. degree in teleinformatics from Brno University of Technology (BUT), Czech Republic, in 2010 and passed the habilitation, in 2013. He is currently an Associate Professor with the BUT, where he is heading the Signal Processing Program at the SIX Research Centre. His main research expertise lies in artificial intelligence, machine learning, genetic programming, e-health, and genetic algorithms.

Aleksandr Ometov (Senior Member, IEEE) received the M.Sc. degree in Information Technology and the D.Sc. (Tech.) degree in Telecommunications from the Tampere University of Technology (TUT), Finland, in 2016 and 2018, respectively. He also holds a Specialist degree in Information Security from the Saint Petersburg State University of Aerospace Instrumentation (SUAI) from 2013. He is a Senior Research Fellow at Tampere University (TAU), Finland, and the coordinator of the CONVERGENCE of Humans and Machines research field funded by the Jane and Aatos Erkko Foundation. He is a Project and Training Manager of EU H2020 MCSA A-WEAR and APROPOS ITN projects. His research interests include wireless communications, information security, computing paradigms, and wearable applications. He was recognized as a "Young Researcher of the Year in Finland" by The Finnish Foundation for Technology Promotion in 2023.