Turkish Journal of Engineering
https://dergipark.org.tr/en/pub/tuje
e-ISSN 2587-1366

Encrypted malware detection methodology without decryption using deep learning-based approaches

Abhay Pratap Singh *1, **Mahendra Singh** 1, **Karamjit Bhatia** 1, **Heman Pathak**1

1 *Gurukula Kangri University, Department of Computer Science, India, rs.abhaypratapsingh@gkv.ac.in, msa@gkv.ac.in, kbhatia@gkv.ac.in, hpathak@gkv.ac.in*

Cite this study: Singh, A. P., Singh, M., Bhatia, K., & Pathak, H. (2024). Encrypted malware detection methodology without decryption using deep learning-based approaches. Turkish Journal of Engineering, 8 (3), 498-509

https://doi.org/10.31127/tuje.1416933

Keywords
Malware
Encrypted traffic
Deep learning
Network security

Research Article

Received: 09.01.2024
Revised: 07.03.2024
Accepted: 08.03.2024
Published: 05.07.2024

Abstract
The encrypted or https traffic on Internet accounts for the safe and secure communication between users and servers. However, cyber attackers are also exploiting https traffic to disguise their malignant activities. Detection of network threats in https traffic is a tiresome task for security experts owing to the convoluted nature of encrypted traffic on the web. Conventional detection techniques decrypt the network content, check it for threats, re-encrypt the network content, and then send it to the server. But this approach jeopardizes the secrecy of data and user. In recent time, deep learning (DL) has emerged as one of the most fruitful AI methods that diminishes the manual resolution of features to enhance classification accuracy. A DL based strategy is suggested for recognition of threat in encrypted communication without using decryption. The three DL algorithms, as used by the proposed approach are, multilayer perceptron (MLP), long short-term memory (LSTM) and 1-D convolutional neural network (1-D CNN), which are experimented on the CTU-13 malware dataset containing flow-based attributes of network traffic. The outcome of the experiment exhibits that MLP based approach performs better in comparison to 1-D CNN and LSTM based ones and other existing approaches. Thus, the secrecy of the data is maintained and the capability of identifying threats in encrypted communication is augmented.

1. Introduction

In the era of digital security, an ever-increasing number of web applications are utilizing security protocols, such as Hypertext Transfer Protocol Secure (HTTPs), Secure Shell (SSH), and Secure Sockets Layer/Transport Layer Security (SSL/TLS), to encipher the content of Internet traffic to protect user's privacy [1]. The most widely utilized protocol for encrypting Internet traffic is the HTTPs. HTTP over SSL/TLS provides a secure and safe communication through a computer network using encryption algorithms [2]. SSL/TLS is the most commonly used security protocols to encipher the content of HTTP. The primary purpose of using encryption across the Internet is to mitigate security threats so that the attacker or third party is unable intercept the information. Alternatively, encryption degraded the ability of network administrators to monitor their infrastructure for malicious traffic identification. The encryption provides a greater advantage to benign users for protecting their privacy, whereas attackers use it to hide their malevolent activities. As the deployment of encryption is on the rise, the diversity of enciphered malware is also evolving simultaneously. Therefore, cyber attackers also started using encryption as a weaponized tool to perform their malicious activity. Since most of the malignant actions are exposed on online platforms [3], cyber attackers are now utilizing other resources such as click jacking, phishing emails, and malvertising campaigns, some of which are exploiting vulnerabilities in genuine web applications or plug-ins to inject a malefic script that switches you to a malefic website [4]. Currently, threat detection techniques like Intrusion Detection Systems (IDSs) or firewalls provide defense against these types of safety risks [5]. The prime objective of using such devices is to distinguish between malignant and benignant traffic by inspecting the running network traffic over the Internet. However, traditional approaches have failed when the network traffic is enciphered. The other possible solution for detecting malware in HTTPs traffic is to install HTTPs interceptor proxy between the client and the server. This interceptor inspects the HTTPs traffic of the network by installing a special certificate in their systems. Then the enciphered traffic is de-ciphered and checked for malefic content. If the inspection is passed, the network traffic is re-encrypted and sent to the destination server [6], as depicted in Figure 1. But this approach leads to several other problems. The first problem is that it violates the privacy of HTTPs protocol. The second one is that deploying an interceptor proxy is more expensive and computationally slow while dealing with encryption and decryption of the network traffic.

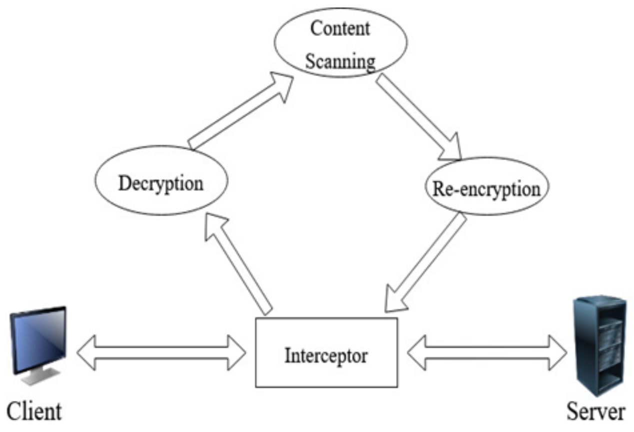


Figure 1. HTTPs Interception.

Therefore, the detection and analysis of enciphered malware remains a continuing challenge for current network security researchers. The application of Machine Learning (ML) based methods to investigate the network traffic measurements have rapidly increased in the recent years. Furthermore, the ML based techniques have shown great permissibility in detecting threats within encrypted traffic. However, the performance of ML based approach is highly reliable on the human-engineered traits and some private traffic information, which can dramatically limit the accuracy and generalizability [7]. The application of DL algorithms in network safety is also growing rapidly. The DL based methodologies facilitate the classification of network traffic by allowing spontaneous extrication and selection of traits by training [8]. The salient point of a DL based modal is that learning ability of it is more than conventional ML models like Support Vector Machine (SVM), Artificial Neural Network (ANN), and k-NN [9]. Hence, these are supposed to learn extremely complex features to achieve greater accuracy with high functionality. DL technology provides a new viewpoint for the applications in which network traffic does not need decryption, but the traits of the network traffic are investigated to handle the malicious traffic [10].

The key contributions of proposed research can be stated as follows:

- Only flow-based features have been extracted from raw encrypted traffic to facilitate traffic classification because the recent adoption of TLS v 1.3 makes classifying malicious traffic challenging.
- The proposed methodology explores different DL models including multi-layer perceptron (MLP), 1-D convolutional neural network (1-D CNN), and long short-term memory (LSTM) neural networks for malware detection in encrypted traffic.
- DL architectures have been tuned and tested while benchmarking with publicly available dataset and performance have been compared with several DL classifiers such as MLP, 1-D CNN, and LSTM.
- A malware detection approach is proposed utilizing DL techniques to achieve higher detection accuracy with a drop in false positive and false negative.

The remaining part of the paper is structured as follows: Section 2 reviews the existing work on malware detection in encrypted traffic and emphasizes the lack of related research. Section 3 explains the phases of the proposed methodology. Architectures of DL techniques are explored in section 4. Section 5 unfolds the experimental details and evaluation indicators. Section 6 contains result analysis. Finally, section 7 concludes the paper.

2. Related work

This section reviews the existing methods being employed for enciphered network traffic classification. We provided a comprehensive review of the ongoing research on malware detection in enciphered traffic with two prime artificial intelligence-based techniques.

2.1 Machine learning based techniques

In [11-12] authors utilized TLS flows, HTTP headers, DNS flow, and SSL/TLS unencrypted metadata information for identifying attacks in enciphered traffic without using decryption. The experimental outcomes displayed a high value for detection accuracy with various ML algorithms. In a similar fashion, Blake Anderson et al. [13] employed many ML algorithms to carry out study and observation of noisy labels and unsteady data. Researchers gathered a number of TLS enciphered flows for one year via a professional malware virtual box and two physically separable large enterprise networks. However, the methodology is dependent largely on human skills to describe the most significant traits. The method exhibited an accuracy of 99% with 0.01 FDR.

An intra-flow data mechanism was proposed in [14], which provided flow-based information to detect and inspect the network's threat. Researchers have also launched a Joy monitoring tool (software) for inspecting the network flows. They gathered an enormous number of malefic flows from threat mesh and normal flows from DMZ of a company network. They studied many features of network traffic, such as series of packet lengths and time, byte organization, and SSL/TLS handshake metadata and then employed these traits for the formation of ML model and classifying enciphered network flow. The detection accuracy of 95.68% was achieved while using flow-based features.

Anish Singh et al. [15] conducted a study on feature analysis of encrypted traffic and divided it into two classes, malefic and benign. The study highlighted feature investigation on the basis of ML models in place of using human skills to explore the appropriate features in enciphered traffic. In addition, authors developed the models using three ML algorithms, namely SVM, XGBoost, and RF, and then carried out feature examination using RFE (recursive feature elimination) method for each one. Among them, the performance of XGBoost was slightly better than RF with the accuracy reaching close to 99%, while SVM yielded a comparatively low accuracy.

A methodology was proposed in [16] based on the intra-flow metadata mechanism. The training and testing data were used from log files created by the Bro IDS tool. The log files used in their work were conn.org, ssl.log, and x509.log. However, in our research work, we used raw PCAP files. They performed with several machine learning algorithms, and experimental results exhibit that XGBoost has the highest detection accuracy of 98.5% among other classifiers. In a similar way, Rui dai et al. [17] proposed a machine learning based model that detected malicious traffic in HTTPs traffic using multi-view features of the network traffic. Authors utilized CTU-13 dataset (log files) and extracted several features such as SSL handshake fields, flow-based statistics and certificates. They investigated and compared four ML algorithms: RF, SVM, DT, and XGBoost. Among these, XGBoost is found to have the highest detection accuracy of 97.71%.

Bryan Scarbrough [18] used only unencrypted portion of the SSL/TLS handshake combined with Open-Source Intelligence (OSINT) data pertaining to IP addresses and domain names. The metadata is then analyzed utilizing three different ML based algorithms: SVM, One-Class SVM (OC-SVM), and an Autoencoder Neural Network. The proposed methodology uses an imbalanced dataset with OC-SVM model achieving high accuracy among all other algorithms for malware detection.

A novel method was proposed in [19] for identifying malignant TLS traffic by using the traits of communication channel. A unique set of attributes for the communication channel was planned with the placement attributes, consistency, and statistical attributes of the TLS network traffic. Then, researchers used a RF algorithm to train and test the detection model. The proposed model achieved an accuracy of 97.44%, which is more conducive for detecting highly disguised malicious traffic.

Luo Ziming et al. [20] introduced a distributed automation mechanism for malware detection in encrypted traffic based on machine learning. They utilized TLS flow of metadata and contextual flow data from network traffic. They applied three ML based algorithms, namely SVM, RF, and XGBoost, for testing the performance of ML model for identification of threats. From experimental results, the performance of RF was found the best among all classifiers. Moreover, this research conducted experiments using the multi-classification model. In addition, Wei Wang et al. [21] implemented an efficient feature extraction methodology based on structural correlation for the identification of threats in TLS malicious enciphered traffic. The extracted features are fed into a RF based ML model. The results generated out of experiment confirmed the accuracy of the model to 99.38%.

Apart from these, a novel unsupervised methodology was presented in [22] for detecting and clustering malicious TLS flows. Researchers built an unsupervised detector that measures the distance to the cluster to determine whether a given flow is malicious or not. They have also evaluated their approach using 972k traces and 35M TLS flows from a commercial sandbox. The proposed unsupervised detector achieves a F1 score of 0.91 and an FDR of 0.032% over the network traffic.

The above ML-based techniques needed the manual design of network traffic features and could not handle the end-to-end enciphered traffic classification. Therefore, the researchers are now utilizing DL networks to design the detection model for enciphered traffic classification to mitigate this issue.

2.2 Deep Learning based techniques

DL methods have been extensively utilized in Image Processing and Natural Language Processing (NLP) but are novel to the field of malicious enciphered traffic identification. Currently, the research on malicious enciphered traffic detection using DL methods has limited studies. This section reviews existing DL-based techniques to deal with threats in enciphered network traffic. Tangda Yu et al. [23] proposed an enciphered malignant network traffic detection method utilizing neural network-based techniques. The authors utilized a self-generated dataset and marked the benign and malicious flow with labels. They used multilayer networks of Autoencoder for feature extraction. Experimental results revealed that the proposed system had high detection accuracy and low loss rate.

Zhihong Zhou et al. [24] suggested a malignant SSL/TLS traffic detection strategy by employing the feature adaptive learning. The proposed detection system consists of three phases; first phase pre-processes the data, during second phase, an unsupervised neural network automatically extracts essential features from the encrypted traffic and optimizes the input data, and in third phase logistic regression classifier is applied which produces classification accuracy of 89.25%. However, the model in unable to predict future malicious traffic pattern and values of FPR and FNR are also not included in the classification result.

Wei Jihong et al. [25] introduced the Hybrid Neural Network Identification Model (HNNIM) for the classification of malicious threats in TLS-based network traffic. The proposed model comprises two layers, the first layer is used for feature extraction, and the second layer uses these features as input for learning. The HNNIM model combined the plain text of the handshake phase of the TLS protocol information and TCP protocol header field information, which integrates a fully connected deep neural network for an effective identification and classification of enciphered traffic. The experimental results showed that the proposed HNNIM model achieves the average accuracy of 89.28% on the multi-classification task, which was higher than other classifiers. Similar work was carried out by researchers in [26] by proposing a hybrid DL model to classify and detect enciphered network traffic. The CNN and gated recurrent unit (GRU) were used together for rapid feature extraction and learning. This hybrid model provides high accuracy and can be implemented in different networking environments to uniquely classify malicious traffic. Furthermore, the effectiveness of the proposed hybrid DL model was evaluated and tested using three different datasets NSL-KDD, UNSW-NB15, and CICIDS 2017. The proposed work developed a DL based model for identification of malicious activities in enciphered traffic.

A novel method was proposed by Wajdi Bazuhair et al. in [27] where connection features of network traffic are converted into images using the augmentation method. Authors utilized *Perlin Noise* as a carrier function to augment images of network flow features to be used with CNN for training and testing the DL model and successfully classified malignant and benign traffic. The proposed DL model performs better than the traditional machine learning based models by achieving a high detection accuracy of 97%, a low false negative rate of 0.4%, and a relatively higher false positive rate of 5.6%. Another DL based light-weight framework named as Deep-Full Range (DFR) was proposed in [28] for enciphered traffic classification and malicious traffic detection. Three DL algorithms, namely, CNN, LSTM, SAE (stacked auto encoder) are utilized for understanding the raw network traffic. The proposed framework DFR was evaluated on two public datasets and provided a much more robust and precise performance for both enciphered and malicious traffic classification with a minimum storage resource requirement.

3. Proposed methodology

The proposed methodology, which utilizes DL architectures to detect malware in enciphered network traffic, is elaborated in this section. The various phases of the methodology are depicted in Figure 2. Initially, raw network traffic comprising both benign and malicious packet capture files (PCAPs) is collected. Flow based features are then extracted from these PCAPs by the flow generator tool and saved as .CSV file. Next, the preprocessing phase is applied to the .CSV file to clean the data. Subsequently, different DL classifiers are applied to the dataset to train, validate, and test the model to classify the benign and malicious traffic. Finally, several evaluation indicators are used to evaluate the performance of the DL techniques.

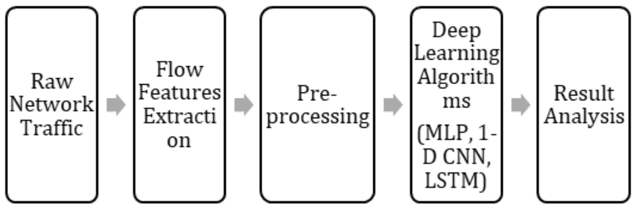


Figure 2. Overview of proposed methodology.

3.1. Dataset description

In our research, we have used the CTU-13 dataset collected from CTU University, Czech Republic available in public domain [29]. CTU-13 dataset contains 13 malicious traffic captures comprising benign and malicious packets. The malicious and benign part of the dataset was collected using a Windows 7 systems running on the virtual machine. The benign traffic utilizes unknown web browsers, and the malware perhaps utilizes its own libraries to communicate with the internet. These network traffic captures are available as PCAP files. We have used these PCAP files for dataset generation. A PCAP file contains the information of network packets and includes the header and payload of every packet. These files are utilized primarily in network analysis and security related tasks.

3.2. Flow features extraction

In this step, various features from benign and malicious PCAPs are extracted through the CICFlowmeter tool [30] and saved into a CSV format file. This tool can pull out more than 80 statistical network flow attributes. It is a network traffic stream creator tool, coded in Java and gives more elasticity when extracting relevant information from PCAP files. It also produces the bidirectional flows (source to destination and destination to source) of captured network flux. A flow in the network is represented by a sequence of packets from a source node to a destination node with similar values for five attributes: Source IP, Destination IP, Source Port, Destination Port, and Protocol. Flow base features can facilitate in-depth network traffic inspection. Table 1 displays a list of flow-based features except Source IP, Destination IP, Source Port, and Protocol because these features are either intrinsically non informative or set up within a simulated environment [31-32].

3.3 Data preprocessing

The original data may have redundancies, errors, and imbalance issues; therefore, it is necessary to remove all such pitfalls from the data before further processing. The features such as Flow Byte/s, Flow Packet/s, and Forward Packet/s that contain infinite and empty values are removed from the dataset using the Pandas library of Python, which relies on a numerical based technique. Dropna() method is utilized to remove the rows which contain infinite and empty values.

3.4 Data normalization

Normalization is an essential step when training a DL model because, without normalization, the model may give false-prone outcomes. For normalization, several techniques are used, literature survey reveals that min max approach, which exhibits good result in terms of scaling and solving outliers. Therefore, a feature scaling technique (min-max scalar) was applied to normalize the values of the dataset for predicting better results and fast

Table 1. The list of flow-based statistical features.

SN	Feature Name	SN	Feature Name	SN	Feature Name
1	Destination Port	27	Bwd IAT Mean	53	Avg Packet Size
2	Flow Duration	28	Bwd IAT Std	54	Avg Fwd Segment Size
3	Total Fwd Packets	29	Bwd IAT Max	55	Avg Bwd Segment Size
4	Total Bwd Packets	30	Bwd IAT Min	56	Fwd Header Length
5	Total Length of Fwd Pkts	31	Fwd PSH Flags	57	Fwd Avg Bytes/bulk
6	Total Length of Bwd Pkts	32	Bwd PSH Flags	58	Fwd Avg Packets /bulk
7	Fwd Packet Length Max	33	Fwd URG Flags	59	Fwd Avg Bulk Rate
8	Fwd Packet Length Min	34	Bwd URG Flags	60	Bwd Avg Bytes/Bulk
9	Fwd Packet Length Mean	35	Fwd Header Length	61	Bwd Avg Packets/Bulk
10	Fwd Packet Length Std	36	Bwd Header Length	62	Bwd Avg Bulk Rate
11	Bwd Packet Length Max	37	Fwd Packets/s	63	Sub flow Fwd Packets
12	Bwd Packet Length Min	38	Bwd Packets/s	64	Sub flow Fwd Bytes
13	Bwd Packet Length Mean	39	Min Packet Length	65	Sub flow Bwd Packets
14	Bwd Packet Length Std	40	Max Packet Length	66	Sub flow Bwd Bytes
15	Flow Bytes/s	41	Packet Length Mean	67	Init-Win-bytes-forward
16	Flow Packets/s	42	Packet Length Std	68	Init-Win-bytes-backward
17	Flow IAT Mean	43	Packet Length Variance	69	act-data-pkt-fwd
18	Flow IAT Std	44	Fin Flag Count	70	min-seg-size-forward
19	Flow IAT Max	45	Syn Flag Count	71	Active Mean
20	Flow IAT Min	46	RST Flag Count	72	Active Std
21	Fwd IAT Total	47	PSH Flag Count	73	Active Max
22	Fwd IAT Mean	48	ACK Flag Count	74	Active Min
23	Fwd IAT Std	49	URG Flag Count	75	Idle Mean
24	Fwd IAT Max	50	CWE Flag Count	76	Idle Std
25	Fwd IAT Min	51	ECE Flag Count	77	Idle Max
26	Bwd IAT Total	52	Down/Up Ratio	78	Idle Min

convergence of the model [33]. Min-Max scaling feature values in the range of [0, 1] using Equation 1. Now, all the features have the same weights and are in one scope.

$$X_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (1)$$

where, $\min(x)$ and $\max(x)$ indicate the minimum and maximum values of feature x, respectively. X_{scaled} is the normalized feature.

4. Deep Learning Classifiers

Our approach aims to utilize the power and efficiency of the DL models to detect malware in HTTPs traffic. A scalable DL model provides an optimal solution with incredible accuracy. The proposed methodology applies three DL classifiers MLP, 1-D CNN, and LSTM for malicious traffic detection. Our dataset consists of time series data and high dimensional input features; where, MLP, 1-D CNN, and LSTM are reported to be more accurate [34]. An overview of these three classifiers along with the summary of used hyper-parameters is outlined hereunder.

4.1 MLP (Multi-layer Perceptron Classifier)

The MLP classifier is primarily used for regression and classification problems [35]. It is a feed-forward neural network that consists of three layers: an input layer, various hidden layers, and an output layer. Every layer has various neurons which are closely connected to the adjacent layers. A neuron uses a weighted sum of its input feature and produces an output that passes through a non-linear activation function. MLP also utilizes back propagation for training the neural network. The input layer contains the feature of a dataset that feeds into the hidden layer, which acts as a computational engine between the input layer and output layer. Finally, the output layer exhibits the result of the given input feature. In our proposed study, the selection of hyper-parameters for the MLP classifier is illustrated in Table 2.

Table 2. Hyper-parameters summary for MLP.

Hyper-parameters	Value
Hidden layer size	(100,100)
Learning rate	0.1
Loss function	Cross-entropy
Optimizer	Stochastic gradient descent (sgd)
Activation function	Logistic
Max iterations	200

4.2 1-D CNN (1-Dimensional convolutional neural network)

The CNN model is primarily utilized in image processing, computer vision, and object detection. This type of architecture consists of an input layer, various hidden layers (convolution layers, pooling, and fully connected), and an output layer. For most image processing applications, 2-D CNN is applied to image data. It is also known as 2 dimensional CNN because the kernel moves the image from left to right and top to bottom, whereas the 1-D CNN kernel slides along one dimension. The only major difference between 1-D CNN and 2-D CNN is the filter and input dimensions of the data [36]. We selected 1D-CNN over 2D-CNN for our study because the structure of dataset format is text or sequential. The architecture of the 1D-CNN classifier is shown in Figure 3.

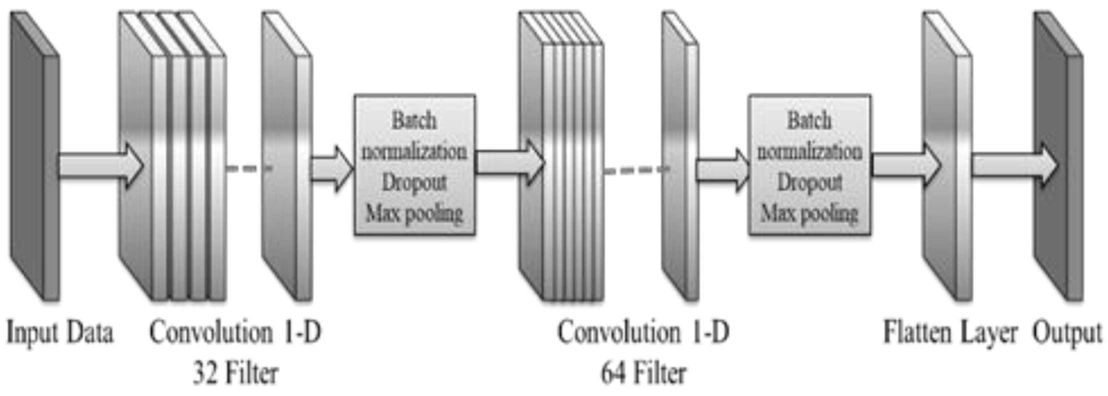


Figure 3. Architecture of 1-D CNN.

The architecture of 1D-CNN consists of two convolution layers, batch normalization, dropout, max pooling, flatten layer, and a dense connected layer with a sigmoid activation function.

The first convolution layer processes the input data with 32 filters, where kernel size is 2 with stride 1 [37]. Each filter moves 1 step after one convolution operation. The results of the convolution layer are forwarded to Rectified Linear Unit (ReLU) activation function. The purpose of the ReLU function is to remove all negative values in the filtered layer and replace them with zeros. The equation for ReLU is defined in Equation 2.

$$f(x) = \max(0, x) \quad (2)$$

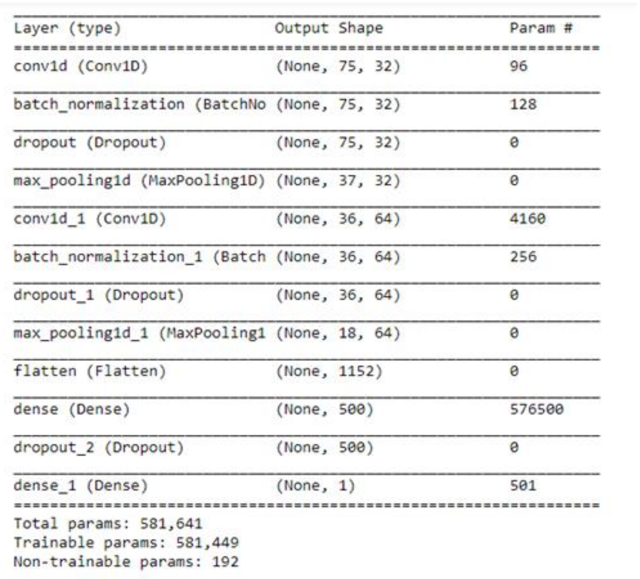
Afterward, the results are processed with batch normalization, dropout, and max-pooling layer. Batch normalization is used for improving the speed and performance of neural networks [38]. It makes the training process of the neural networks easy. The key idea is to use the dropout technique to randomly delete units from the neural networks during the training process, keeping the neural network away from being dependent on some specific features leading to better classification results in any circumstances [38]. The main objective of the dropout layer is to prevent over fitting issues. Max-pooling layer is used to reduce the size of convoluted features and computation in the network. The result goes through a second convolution layer, which is similar to the first one. The only difference between these two convolutional layers is that the second convolution layer has 64 filters and different dropout rates. The flatten layer involves in transforming the entire pooled feature map matrix into a single column vector. In the end, data will move to a dense connected layer with 500 neurons, followed by a ReLU activation function and a 50% dropout rate. Finally, the output label is attained by the sigmoid activation function (Equation 3). In the context of the proposed research, the selected hyper-parameters for 1-D CNN are illustrated in Table 3.

Figure 4 shows the model summary, describing each layer and the number of parameters. The total numbers of parameters used in this model are 581,641, where 581,449 are trainable, and 192 are non-trainable.

$$f(x) = \frac{1}{1 + e^{-x}} \quad (3)$$

Table 3. Hyper-parameters Summary for 1-D CNN.

Hyper-parameters	Value
Number of filters (1D-CNN1)	32
Number of filters (1D- CNN2)	64
Kernel size	2
Max-pooling	Pool size (2,2)
Dropout (1D-CNN1)	0.2
Dropout (1D-CNN2)	0.5
Dense layer	500 = units
Dropout (dense layer)	0.5
Loss function	Binarycross-entropy
Optimizer	Adam
Learning rate	0.001
Batch size	32
Number of epochs	200



```
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 75, 32)            96
_________________________________________________________________
batch_normalization (BatchNo (None, 75, 32)            128
_________________________________________________________________
dropout (Dropout)            (None, 75, 32)            0
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 37, 32)            0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 36, 64)            4160
_________________________________________________________________
batch_normalization_1 (Batch (None, 36, 64)            256
_________________________________________________________________
dropout_1 (Dropout)          (None, 36, 64)            0
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 18, 64)            0
_________________________________________________________________
flatten (Flatten)            (None, 1152)              0
_________________________________________________________________
dense (Dense)                (None, 500)               576500
_________________________________________________________________
dropout_2 (Dropout)          (None, 500)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 501
=================================================================
Total params: 581,641
Trainable params: 581,449
Non-trainable params: 192
```

Figure 4. 1-D CNN Model Summary.

4.3. Long Short Term Memory (LSTM)

Hochreiter et al. [39] presented a model architecture known as the LSTM network. This is a type of neural network which is especially designed to prevent the long term dependency problem that cannot be resolved in a Recurrent Neural Network (RNN). It enables the efficient learning process of both long and short term dependencies by modifying the fundamental processing unit. The LSTM based network also resolves the issues related to gradient vanishing and gradient exploding when the network is too large [40]. Typically, LSTM is a type of network which is designed to deal with time-related information or sequential data. Figure 5 shows the layered architecture LSTM classifier.

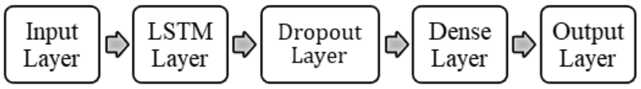


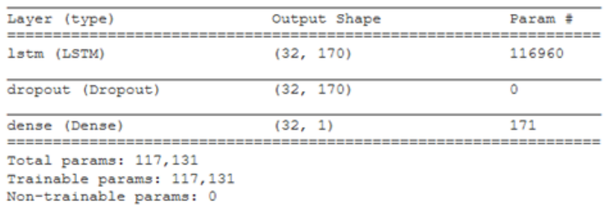
Figure 5. Architecture of LSTM.

LSTM has multi-layered architecture. First, the input data will pass through to the LSTM layer with 170 neurons. In order to prevent overfitting and to achieve better generalization of the model, the dropout layer is used. At last, data will enter to a densely connected layer followed by a sigmoid activation function to predict the output. In the proposed research work, the list of selected hyper-parameters for the LSTM classifier is depicted in Table 4.

Figure 6 all are shows the model summary; the total number of parameters used in this model is 117,131 where 117,131 are trainable.

Table 4. Hyper-parameters Summary for LSTM.

Hyper-parameters	Value
Hidden layer	170 Neurons
Dropout	0.1
Loss function	Binary-cross entropy
Optimizer	Adam
Activation function	Sigmoid
Learning rate	0.01
Max epoch	200
Batch size	32



```
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (32, 170)                 116960

dropout (Dropout)            (32, 170)                 0

dense (Dense)                (32, 1)                   171
=================================================================
Total params: 117,131
Trainable params: 117,131
Non-trainable params: 0
```

Figure 6. LSTM Model Summary.

5. Experimental Details

We have implemented three DL architectures with all the layers and activation function built on the flow-based features extracted from raw PCAPs. The dataset is divided into 7:2:1, i.e., 70 % of data is used for training the model, 20 % data is used for validation, and 10 % data is used as test set to evaluate the effectiveness of the proposed methodology. It contains 288,000 packets for training, 72000 packets for validation, and 40000 packets for testing. All the evaluations are performed on the test set data, which the model has never seen during the training phase. For the proposed research study, we acquired the Google Colab Pro platform for efficiently performing all experiments. It enables us faster GPUs, more memory, and faster executions.

5.1 Evaluation Indicators

The performance of the classifiers can be measured using a number of evaluation indicators. The DL classifiers used in the proposed research are evaluated based on indicators of accuracy, precision, recall, F1-score, FPR, and FNR. Achieving the highest accuracy of the classifier is not the only relevant factor in evaluating the classifier's reliability. Therefore, we utilize several evaluation indicators to assess the reliability of the proposed methodology. These evaluation indicators are defined the follows:

TP (True Positive): Encrypted malicious traffic identified as malicious traffic.
FP (False Positive): Encrypted benign traffic identified as malicious traffic.
TN (True Negative): Encrypted benign traffic identified as benign traffic.
FN (False Negative): Encrypted malicious traffic identified as benign traffic.

The values of the indicators are determined as shown in Equation 4-9.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F1 - Score = 2.\frac{precison \times recall}{precison + recall} \tag{7}$$

$$FPR = \frac{FP}{TN + FP} \tag{8}$$

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

6. Results and Analysis

The proposed methodology uses DL classifiers to classify malicious and benign traffic in enciphered network traffic. The following sub-sections elaborate and analyze the results obtained by each classifier used.

6.1 MLP Classifier Results

In order to retrieve the best hyper parameter for MLP based classification technique, we performed a series of experiments with different number of learning rate and the number of iterations (Figure 7).

Table 5. Results of MLP classifier.

Iterations	Training accuracy (%)	Validation accuracy (%)	Testing accuracy (%)
50	97.17	97.20	97.08
100	97.52	97.57	97.60
150	99.06	99.03	99.01
200	99.12	99.13	99.10

This phenomenon is known as hyper-parameter tuning. Table 2 shows the list of hyper parameters used in MLP classification. Initially, the neural network is trained for a small number of iterations, and it is found that as we increase the number of iterations, the accuracy also improves. Finally, the maximum testing accuracy of 99.10% is achieved at 200 iterations. The accuracy is not changing after 200 iterations because the error rate is significantly decreased. The results obtained from the MLP classifier based on different number of iterations are shown in Table 5.

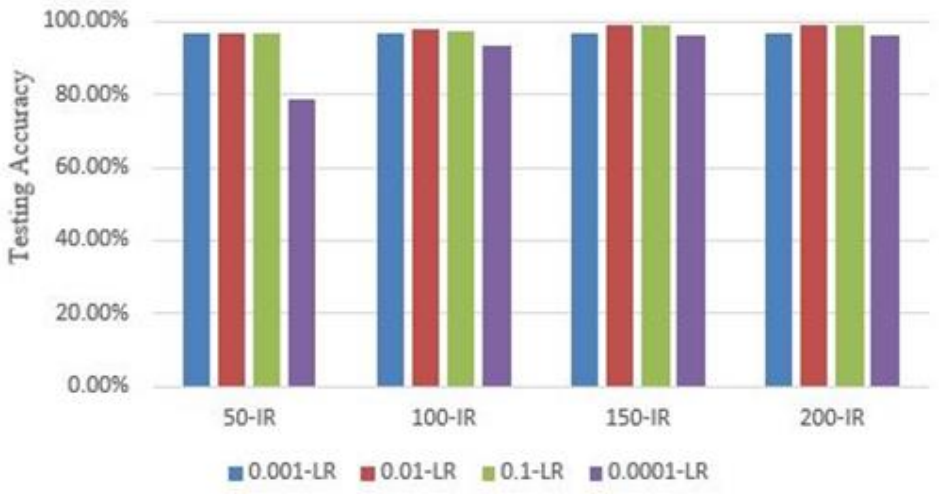


Figure 7. Accuracy changes with different learning rate and iterations.

6.2 1-D CNN Classifier Results

In the 1-D CNN classifier, we also tune the hyper parameter for the classification technique to find a suitable list of hyper parameters for the classification process (Table 3). Learning rate and batch size are important hyper parameters for the DL process. Figure 8 shows the accuracy generated with different numbers of learning rate and batch size for the classification model.

The learning rate determines how fast the parameters are updated, and batch size refers to the number of samples that will be propagated through the neural network before updating the model hyper parameter. The maximum testing accuracy achieved for the 1-D CNN classifier is 98.68% at 200 epochs as shown in Table 6.

The training and validation accuracy, along with training and validation loss for the 1-D CNN model, is shown in Figure 9 and 10 respectively. The purpose of using these curves is to diagnose the overfitting and under fitting problems in the model. It can be observed from Figure 9 and 10 that the gap between training and validation accuracy and that for training and validation loss is relatively small. Thus, it shows no sign of overfitting and underfitting in the model.

Table 6. Results of 1-D CNN classifier.

Epochs	Training accuracy (%)	Validation accuracy (%)	Testing accuracy (%)
50	98.36	98.34	98.21
100	98.66	85.64	98.50
150	98.67	99.14	98.69
200	98.67	98.53	98.68

6.3 LSTM classifier results

Similarly, the hyper parameter is also tuned for this classification approach in order to find a suitable list of hyper parameters for the classifier (Table 4). Figure 11 shows the accuracy regarding the different numbers of learning rate and batch size for the classification technique.

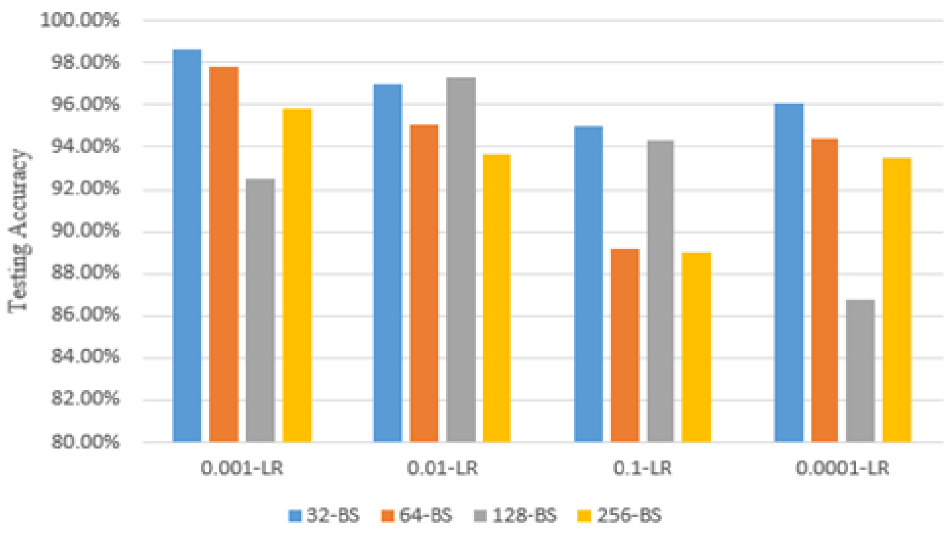


Figure 8. Accuracy changes with different learning rate and batch size.

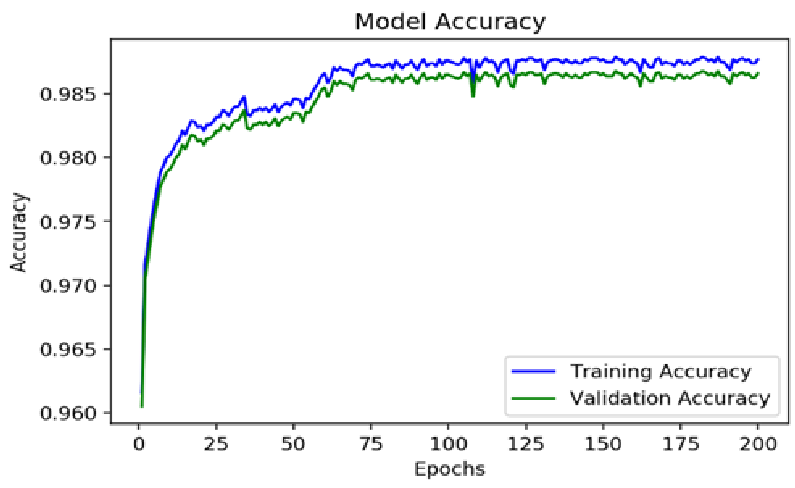


Figure 9. 1-D CNN training and validation accuracy.

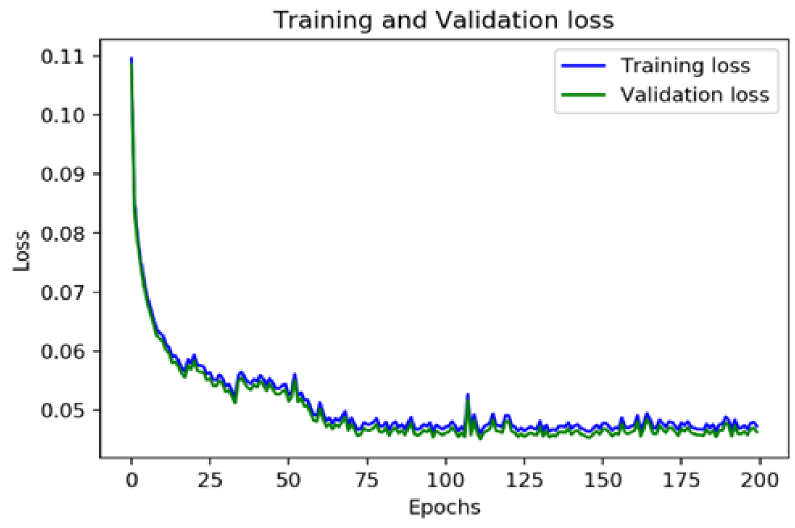


Figure 10. 1-D CNN training and validation loss.

It can be observed from the Table 7 that with each round of epoch, accuracy also increases. The testing accuracy stabilizes at 97.33 % when epoch gets close to 200. The accuracy of the classifier is high, and the technique is relatively reliable, as verified by statistical tests.

Table 7. Results of LSTM classifier.

Epochs	Training accuracy (%)	Validation accuracy (%)	Testing accuracy (%)
50	96.01	94.78	94.92
100	94.94	95.12	95.87
150	97.38	97.43	97.21
200	97.36	97.39	97.33

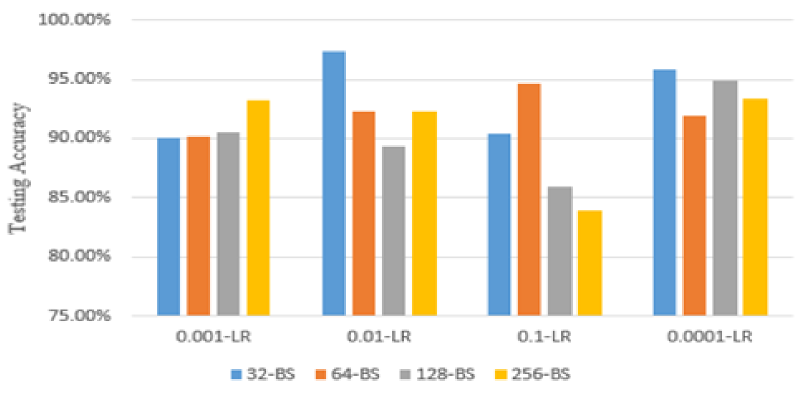


Figure 11. Accuracy changes with different learning rate and batch size.

6.4 Comparative analysis

Results of the proposed classification strategies (MLP, 1-D CNN, and LSTM based), and that of other approaches, are compared and analyzed in this section. The numerical values of evaluation indicators for the proposed strategies and that of other state of arts are shown in Table 8. It is obvious that the accuracy of MLP based technique is significantly higher than 1-D CNN and LSTM based ones for the classification of encrypted malicious data set containing flow-based features of the traffic. Moreover, it is observed that the proposed DL based methodologies (MLP and 1-D CNN based) perform better than the existing studies [17] and [24] in terms of accuracy, precision, recall, and F1-score. This significant improvement of accuracy, precision, recall, and F1-score can be attributed to the use of selective statistical features based on network flow, in our proposed research study. Another contribution of our proposed research study is to generate the optimum low values of FPR and FNR, which are essential indicators in cyber security in order to assess the performance of a detection strategy. The high values of FPR and FNR may lead to misclassification in any detection model. The existing studies [17] and [24] have not considered FPR and FNR to evaluate their classification strategies.

The experimental results are not only evaluated in terms of evaluation indicators, but also as a confusion matrix. A confusion matrix is a simple table layout used to analyze or to understand the performance of the classifier. It contains the information regarding the actual and predicted classification on the test data. The confusion matrices of all three classifiers are shown in Figure 12, 13, and 14 respectively.

Table 8. Comparison of the proposed techniques with existing works.

Techniques		Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	FPR (%)	FNR (%)
Proposed	MLP	99.10	1.0	98.20	99.09	0.0	0.017
	1-D CNN	98.68	97.40	99.99	98.68	0.026	0.00005
	LSTM	97.33	95.22	99.60	97.36	0.049	0.003
Rui Dai et.al [17]		97.71	98.11	97.49	97.80	-	-
Zhihong et.al [24]		89.25	-	-	-	-	-

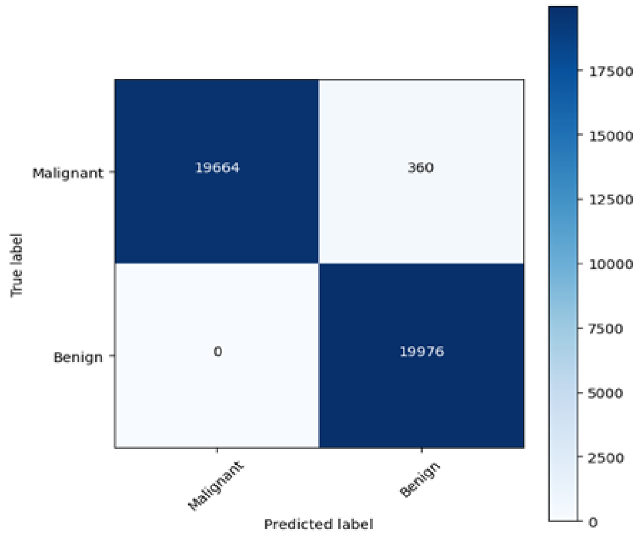


Figure 12. Confusion matrix of MLP model.

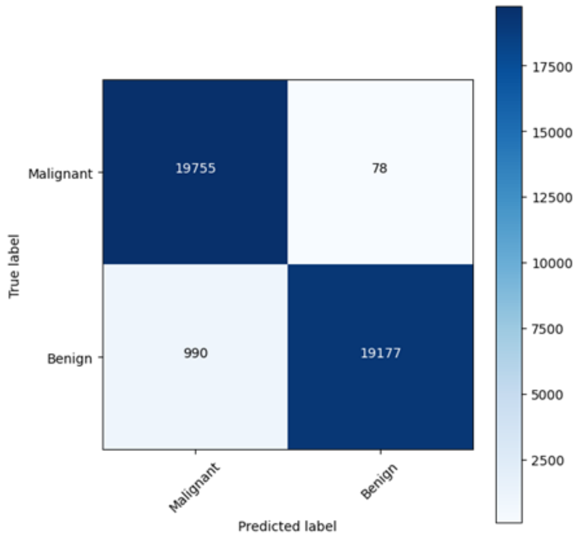


Figure 14. Confusion matrix of LSTM model.

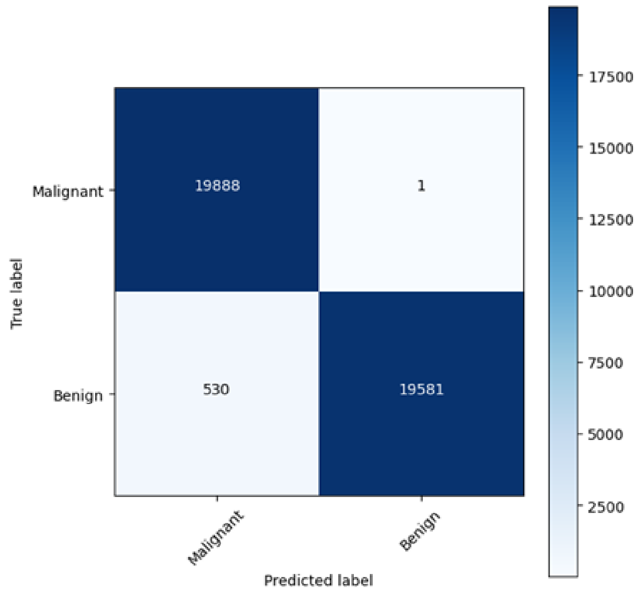


Figure 13. Confusion matrix of 1-D CNN model.

7. Conclusion

As encrypted traffic increases, attackers are also increasingly using encryption to cover up the intention of the attack, which brings considerable value to detection difficulty. Therefore, the importance of encrypted malicious traffic detection has become an increasingly prominent in the prevailing scenario. The researchers are considering the deep learning-based strategies to deal with this critical problem. In this paper, deep learning-based methodology is proposed, which is capable of classifying benign and malicious enciphered traffic with a high accuracy and precision without intercepting the network traffic. The proposed methodology utilizes three DL classifiers MLP, 1-D CNN and LSTM to facilitate classification of network traffic using the CTU-13 dataset. In contrast to other state of arts, the proposed DL-Based approach does not require human intervention for selecting features and private featured details about SSL/TLS metadata and at the same time also maintains the confidentiality of the data. The results of the experiment reveal that the proposed MLP based approach shows a feasible and scalable solution for malware detection in enciphered network traffic using flow-based features. Furthermore, the proposed approach can be tested on other data sets comprising different types of features. For future investigations, deep learning or hybrid models such as resnet and Bi-LSTM ANN can also be explored for detecting the encrypted malware.

Author contributions

Abhay Pratap Singh: Conceptualization, Proposed Methodology, Writing-original draft preparation.
Mahendra Singh: Methodology, Validation, Writing-original draft preparation.
Karamjit Bhatia: Validation, Writing-Reviewing and Editing.
Heman Pathak: Investigation, Writing-Reviewing and Editing.

Conflicts of interest

The authors declare no conflicts of interest.

References

1. Papadogiannaki, E., & Ioannidis, S. (2021). A survey on encrypted network traffic analysis applications, techniques, and countermeasures. ACM Computing Surveys (CSUR), 54(6), 1-35. https://doi.org/10.1145/3457904
2. Singh, A. P., & Singh, M. (2021). A comparative review of malware analysis and detection in HTTPs traffic. International Journal of Computing and Digital Systems, 10(1), 111-123. http://dx.doi.org/10.12785/ijcds/100111
3. Ayas, M. Ş. (2021). A brief review on attack design and detection strategies for networked cyber-physical systems. Turkish Journal of Engineering, 5(1), 1-7. https://doi.org/10.31127/tuje.640282
4. Grier, C., Ballard, L., Caballero, J., Chachra, N., Dietrich, C. J., Levchenko, K., ... & Voelker, G. M. (2012, October). Manufacturing compromise: the emergence of exploit-as-a-service. In Proceedings of the 2012 ACM Conference on Computer and Communications Security, 821-832. https://doi.org/10.1145/2382196.238228
5. Mishra, N., & Pandya, S. (2021). Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review. IEEE Access, 9, 59353-59377. https://doi.org/10.1109/ACCESS.2021.3073408
6. Zhu, T., Weng, Z., Fu, L., & Ruan, L. (2020). A web shell detection method based on multiview feature fusion. Applied Sciences, 10(18), 6274. https://doi.org/10.3390/app10186274
7. Zhao, Y., Yang, Y., Tian, B., Yang, J., Zhang, T., & Hu, N. (2021). Edge intelligence based identification and classification of encrypted traffic of Internet of Things. IEEE Access, 9, 21895-21903. https://doi.org/10.1109/ACCESS.2021.3056216
8. Wang, P., Ye, F., Chen, X., & Qian, Y. (2018). Datanet: Deep learning based encrypted network traffic classification in sdn home gateway. IEEE Access, 6, 55380-55391. https://doi.org/10.1109/ACCESS.2018.2872430
9. Atli, B. G., Miche, Y., Kalliola, A., Oliver, I., Holtmanns, S., & Lendasse, A. (2018). Anomaly-based intrusion detection using extreme learning machine and aggregation of network traffic statistics in probability space. Cognitive Computation, 10(5), 848-863. https://doi.org/10.1007/s12559-018-9564-y
10. Guo, L., Wu, Q., Liu, S., Duan, M., Li, H., & Sun, J. (2020). Deep learning-based real-time VPN encrypted traffic identification methods. Journal of Real-Time Image Processing, 17(1), 103-114. https://doi.org/10.1007/s11554-019-00930-6
11. Anderson, B., & McGrew, D. (2016, October). Identifying encrypted malware traffic with contextual flow data. In Proceedings of the 2016 ACM workshop on Artificial Intelligence and Security, 35-46. https://doi.org/10.1145/2996758.2996768
12. Anderson, B., Paul, S., & McGrew, D. (2018). Deciphering malware's use of TLS (without decryption). Journal of Computer Virology and Hacking Techniques, 14, 195-211. https://doi.org/10.1007/s11416-017-0306-6
13. Anderson, B., & McGrew, D. (2017, August). Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1723-1732. https://doi.org/10.1145/3097983.3098163
14. McGrew, D., & Anderson, B. (2016). Enhanced telemetry for encrypted threat analytics. In 2016 IEEE 24th International Conference on Network Protocols (ICNP), 1-6. https://doi.org/10.1109/ICNP.2016.7785325
15. Shekhawat, A. S., Di Troia, F., & Stamp, M. (2019). Feature analysis of encrypted malicious traffic. Expert Systems with Applications, 125, 130-141. https://doi.org/10.1016/j.eswa.2019.01.064
16. Hamad, M., Durad, M. H., & Yousaf, M. (2018). Mitigation of the effect of standard networks attacks in SSL encrypted traffic by encrypted traffic analysis. VFAST Transactions on Mathematics, 6(1), 15-22. https://doi.org/10.21015/vtm.v8i1.578
17. Dai, R., Gao, C., Lang, B., Yang, L., Liu, H., & Chen, S. (2019, November). SSL malicious traffic detection based on multi-view features. In Proceedings of the 2019 9th International Conference on Communication and Network Security, 40-46. https://doi.org/10.1145/3371676.3371697
18. Scarbrough, B. (2021). Malware Detection in Encrypted TLS Traffic Through Machine Learning. Global Information Assurance Certification Paper.
19. Zheng, R., Liu, J., Li, K., Liao, S., & Liu, L. (2020, August). Detecting malicious tls network traffic based on communication channel features. In 2020 IEEE 8th International Conference on Information, Communication and Networks (ICICN), 14-19. https://doi.org/10.1109/ICICN51133.2020.9205087

20. Luo, Z. M., & Xu, S. B. (2020). Scheme for identifying malware traffic with TLS data based on machine learning. Chinese Journal of Network and Information Security, 6(1), 77-83.
21. Wang, W., Sun, C. S., & Ye, J. N. (2021). A method for TLS malicious traffic identification based on machine learning. Advances in Science and Technology, 105, 291-301. https://doi.org/10.4028/www.scientific.net/AST.105.291
22. Gomez, G., Kotzias, P., Dell'Amico, M., Bilge, L., & Caballero, J. (2023). Unsupervised detection and clustering of malicious tls flows. Security and Communication Networks, 2023(1), 3676692. https://doi.org/10.1155/2023/3676692
23. Yu, T., Zou, F., Li, L., & Yi, P. (2019). An encrypted malicious traffic detection system based on neural network. In 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 62-70. https://doi.org/10.1109/CyberC.2019.00020
24. Zhou, Z., Bin, H., Li, J., Yin, Y., Chen, X., Ma, J., & Yao, L. (2022). Malicious encrypted traffic features extraction model based on unsupervised feature adaptive learning. Journal of Computer Virology and Hacking Techniques, 18(4), 453-463. https://doi.org/10.1007/s11416-022-00429-y
25. Jie, F. (2020, September). Research on malicious TLS traffic identification based on hybrid neural network. In 2020 International Conference on Advance in Ambient Computing and Intelligence (ICAACI), 42-46. https://doi.org/10.1109/ICAACI50733.2020.00013
26. Bakhshi, T., & Ghita, B. (2021). Anomaly detection in encrypted internet traffic using hybrid deep learning. Security and Communication Networks, 2021(1), 5363750. https://doi.org/10.1155/2021/5363750
27. Bazuhair, W., & Lee, W. (2020, January). Detecting malign encrypted network traffic using perlin noise and convolutional neural network. In 2020 10th Annual Computing and Communication Workshop and Conference (CCWC), 0200-0206. https://doi.org/10.1109/CCWC47524.2020.9031116
28. Zeng, Y., Gu, H., Wei, W., & Guo, Y. (2019). Deep-Full-Range: A deep learning based network encrypted traffic classification and intrusion detection framework. IEEE Access, 7, 45182-45190. https://doi.org/10.1109/ACCESS.2019.2908225
29. Ctu-13 dataset. https://www.stratosphereips.org/datasets-ctu13
30. CICFlowmeter. https://www.unb.ca/cic/reserach/applications.html
31. Zhang, C., Chen, Y., Meng, Y., Ruan, F., Chen, R., Li, Y., & Yang, Y. (2021). A novel framework design of network intrusion detection based on machine learning techniques. Security and Communication Networks, 2021(1), 6610675. https://doi.org/10.1155/2021/6610675
32. Pontes, C. F., De Souza, M. M., Gondim, J. J., Bishop, M., & Marotta, M. A. (2021). A new method for flow-based network intrusion detection using the inverse Potts model. IEEE Transactions on Network and Service Management, 18(2), 1125-1136. https://doi.org/10.1109/TNSM.2021.3075503
33. Başarslan, M. S., & Kayaalp, F. (2023). Sentiment analysis with ensemble and machine learning methods in multi-domain datasets. Turkish Journal of Engineering, 7(2), 141-148. https://doi.org/10.31127/tuje.1079698
34. Rezaei, S., & Liu, X. (2019). Deep learning for encrypted traffic classification: An overview. IEEE Communications Magazine, 57(5), 76-81. https://doi.org/10.1109/MCOM.2019.1800819
35. Dirik, M. (2023). Machine learning-based lung cancer diagnosis. Turkish Journal of Engineering, 7(4), 322-330. https://doi.org/10.31127/tuje.1180931
36. Sharma, A., Malacaria, P., & Khouzani, M. H. R. (2019, June). Malware detection using 1-dimensional convolutional neural networks. In 2019 IEEE European symposium on security and privacy workshops (EuroS&PW), 247-256. https://doi.org/10.1109/EuroSPW.2019.00034
37. Azizjon, M., Jumabek, A., & Kim, W. (2020, February). 1D CNN based network intrusion detection with normalization on imbalanced data. In 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 218-224. https://doi.org/10.1109/ICAIIC48513.2020.9064976
38. Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., ... & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. Journal of Big Data, 8, 1-74. https://doi.org/10.1186/s40537-021-00444-8
39. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735
40. Dang, D., Di Troia, F., & Stamp, M. (2021). Malware classification using long short-term memory models. Cryptography and Security,1-16. https://doi.org/10.48550/arXiv.2103.02746