2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

Tor Traffic Classification Based on Burst Features

Ding Li[1], Yi Pan[2], and Yinlong Xu[1]

[1] University of Science and Technology of China, No. 96 Jinzhai Road, Hefei City, Anhui Province 230026, China
[2] Anhui Property Rights Trading Center Co., Ltd, Binhu District, Hefei City, Anhui Province 230022, China

Abstract. The classification of Tor traffic is of crucial importance in the identification of anonymous web applications and the defense against cybercrime. Previous studies have focused on the automatic extraction of raw traffic features by means of deep learning algorithms. However, these methods have neglected the global intrinsic relationship between local features at different data locations, which has resulted in limited classification performance. In this regard, a dark net traffic classification method based on burst feature aggregation, called burst matrix, is proposed. The proposed method involves the aggregation of temporal and length features of Tor traffic in terms of bursts, followed by the capture of local spatio-temporal features from the burst matrix using convolutional neural networks. The intrinsic relationships and hidden connections between the previously extracted spatio-temporal features are then mined using the self-attention mechanism. The efficacy of the burst matrix method is then evaluated using the ISCXTor2016 dataset. The experimental results demonstrate that the burst matrix significantly outperforms other contemporary methods, attaining an F1-score of over 95%.

Keywords: Network Security, Encrypted Traffic Identification, Dark Web, Onion Routing, Deep Learning.

1 Introduction

With the development of Internet services, anonymous communication has become an important need to protect users'privacy. Tor [1,2], as a mainstream privacy-enhancing tool, hides users'identities and activities by encrypting and tunneling traffic through distributed nodes. However, its anonymity is also widely used for illegal activities, so identifying Tor traffic and its application types is important for fighting crime. Despite the fact that Tor traffic is encrypted, user interaction traffic when accessing applications can still expose service characteristics [3,4].

Traditional traffic identification relies on packet load analysis, which makes it difficult to handle encrypted traffic effectively. Packet feature and machine learning-based approaches classify encrypted traffic by extracting features and using algorithms, but these methods rely on manual feature engineering or complex feature selection, and their performance is unstable in different network environments. In contrast, deep

learning methods automatically extract high-level features from traffic without manual design. However, current deep learning mainly extracts local spatial or temporal features without sufficiently considering the intrinsic dependencies of global features, resulting in limited classification performance.

In this regard, this paper focuses on the feature representation of Tor traffic, i.e., how to transform raw traffic features into effective features to fully express its global intrinsic connections. Through analysis, it is found that there are significant differences in burst level features among different Tor traffic types. Based on this, a Tor traffic characterisation method based on burst features, burst matrix, is proposed. The method is divided into three steps: firstly, the Tor traffic feature sequence containing time and length information is extracted; secondly, the feature sequence is sliced into burst sequences according to the burst threshold; finally, each burst is converted into a burst matrix using the time and length features of the burst sequence.

To identify different types of Tor traffic, a deep learning model combining CNN and self-attention mechanism is designed in this paper. CNN is used to extract local spatio-temporal features from burst matrices, and the self-attention layer further mines the intrinsic relationships and hidden connections among these features. Experiments show that the method outperforms existing methods by nearly 6% on the ISCXTor-2016 dataset.

This paper is structured as follows: section II introduces the related research on darknet traffic identification; section III details the burst matrix generation method and model framework; section IV evaluates the performance of the proposed method; section V concludes the full paper.

2 Related Work

This section reviews related studies that focus on the categorization of encrypted traffic and the classification of Tor traffic.

2.1 Encrypted traffic analysis

Many researchers [5-10] have used machine learning approaches to handle encrypted traffic classification tasks, such as web page fingerprinting, IoT device identification, malicious communication detection [11], and encrypted application classification. Shen [12] et al. performed web page fingerprinting by extracting block features, sequence features, and statistical features, and generated fine-grained classifiers using a traditional machine learning model. Pinheiro [13] et al. achieved IoT and non-IoT device classification based on packet length statistics and transmission rates. Fang et al. proposed aggregating network packets with the same destination IP and port as a communication channel, extracting distribution, consistency, and statistical features, and detecting malicious HTTPS traffic by combining a genetic algorithm with random forest classification. Existing research tends to leverage the powerful nonlinear learning capabilities of deep learning to automatically obtain effective feature representations. For example, models such as 2D-CNN, 1D-CNN with Bi-GRU, LSTM [14] with a hierarchical attention mechanism, and graph neural networks (GNNs) have demonstrated good performance.

2.2 Tor traffic analysis

In the last decade, there has been extensive progress in Tor traffic analysis research [14]. ETC-PS [15], uses a sequence of session packet lengths to construct traffic paths to represent the interaction between the client and the server, then performs path transformations to show its structure and obtain different information, and finally computes multi-scale path signatures as a distinguishing feature to train traditional machine learning classifiers. Iliadis and Kaifas compared the effectiveness of different feature sets and classification models for darknet traffic prediction. Rao [16] et al. proposed an unsupervised method based on gravitational clustering to identify Tor traffic from non-anonymized network data. Sarwar et al. utilized modified CNN-LSTM and CNN-GRU models for dark web traffic classification and application identification. Additionally, Sirinam [17] et al. proposed combining triple networks and KNN classifiers to perform website fingerprinting for Tor systems. Habibi Lashkari [18] et al. introduced the Deep-Image method, which converts optimized statistical features into grayscale images and combines them with 2D-CNNs to classify dark web traffic. Singh [19] et al. designed a deep transfer learning architecture that integrates pre-trained models and classifiers to distinguish between malicious and benign traffic. Lin et al. combined 1D-CNNs with stacked Bi-LSTM networks to automatically extract advanced features for the fine-grained classification of Tor traffic.

In summary, most current studies leverage deep learning to recognize Tor traffic, with inputs typically consisting of raw traffic features (e.g., length sequences, time sequences) or their simple preprocessing. Although deep learning can automatically extract deep discriminative features, classification accuracy is often limited because these methods fail to fully capture global dependencies and hidden connections among local features, which impacts the classification performance.

Burst features, as localized features of traffic, are commonly utilized in statistical methods. Existing approaches to encrypted traffic identification based on burst features primarily adopt traditional machine learning methods. In this paper, significant differences in the burst features of various Tor traffic types are identified, and a method is proposed that combines burst features with deep learning to enhance Tor traffic recognition.

3 Tor Traffic Identification Based on Burst Characterization

This section begins by analyzing the differences in burst-level features of Tor traffic, then introduces the preprocessing procedure for the burst feature fusion matrix, and finally presents the overall framework of the deep learning model.

3.1 Burst Differences

Since a packet sequence usually consists of multiple streams, the sequence needs to be first divided into individual streams based on the 5-tuple (source/destination IP address, source/destination port, and protocol). Subsequently, each stream is sliced into bursts. A burst is a group of packets that satisfies the following condition: the time interval between each packet and the previous packet is within a set bursting threshold T . If T

is exceeded, a new burst group is generated. As shown in Fig. 1, the streams are categorized into bursts B1 and B2 by exceeding the threshold T. The difference in timestamps of the two groups of packets is greater than T.

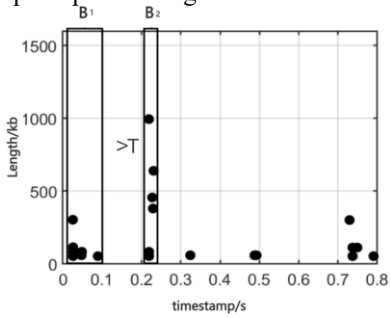


Fig. 1. Burst generation

After segmenting the Tor streams into bursts based on the burst threshold, the <timestamp, length> records of packets within each burst can be extracted. Additionally, the number of in-burst packets and the packet length characteristics for different types of flows can be analyzed. Fig. 2 illustrates the distribution of the number and length characteristics of intra-burst packets for CHAT and FILE types, clearly showing distinct differences in their burst characteristics. To represent these differences more comprehensively, a burst matrix is employed to fuse the extracted features, enabling the model to learn and utilize burst characteristics more effectively.

3.2 Burst Matrix

The burst matrix is defined as follows：

$$M = \left[M^1_{k\times k} M^2_{k\times k} M^3_{k\times k} \cdots M^m_{k\times k}\right] \tag{1}$$

Here, $k\times k$ represents the sub-matrix dimensions, m is the number of bursts, and each sub-matrix is represented as follows.

$$\begin{bmatrix} M_{11} \cdots M_{1k} \\ \vdots \ M_{ij} \ \vdots \\ M_{k1} \cdots M_{kk} \end{bmatrix} \tag{2}$$

The algorithm for generating the sub-matrix is as follows:

For any burst b=[⟨t_1, l_1⟩, . . . , ⟨t_p, l_p⟩], find the maximum packet length Lmax and the normalized maximum packet time interval Tmax, where Tmax=tp−t1.

Define the minimum matrix time unit as t0=Lmax/k and the minimum matrix length unit as l0=Tmax / k.

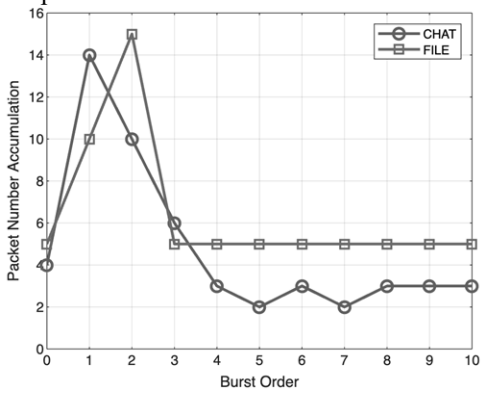
For each message pair in b, there are the following operations.

$$i = \lfloor (t - t_1)/t_0 \rfloor \tag{3}$$

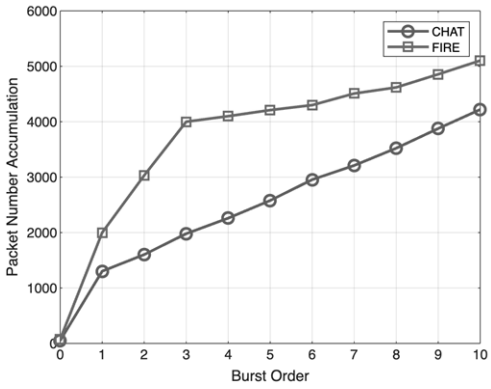
$$j = \lfloor l/l_0 \rfloor \tag{4}$$

$$M_{ij} = M_{ij} + 1 \tag{5}$$

As shown in figure 3. The operation results in each element of the sub-matrix representing the number of packets in the corresponding time interval and length range. Where the timestamp features are normalised, i. e. each timestamp is subtracted from the timestamp of the first packet in the burst.



(a) Distribution of packet numbers



(b) Packet length distribution

Fig. 2. Distribution of the number of packets and the length of packets in a burst

The minimum matrix time unit and the minimum matrix length unit of each sub-matrix are calculated based on their corresponding burst characteristics and are not fixed. The sum of the element values in each sub-matrix equals the total number of packets in that burst. The first n burst sub-matrices of a flow are used as inputs to the model.

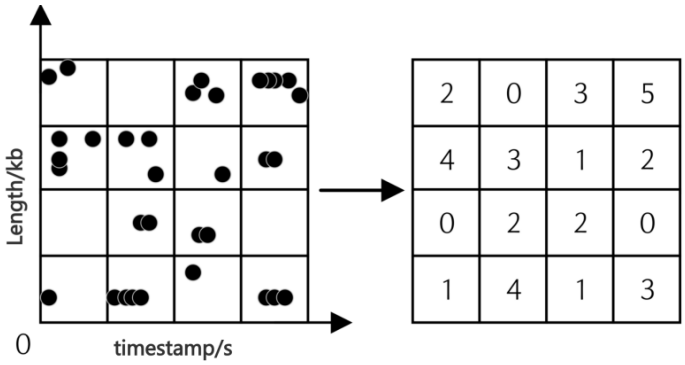


Fig. 3. Burst Matrix Generation

3.3 Modelling Design

This subsection identifies the features in the burst matrix using deep learning models and specifies three neural network layer structures. The overall framework is illustrated in Fig. 4.

Fig. 4. Modelling Framework

The first layer is the convolutional layer, designed to capture the local features of the bursts. The output of the convolutional layer is referred to as the feature map, where each unit in the feature map is connected to a local region of the matrix via a set of convolutional kernels. To preserve the distinct length-distributed features of different matrices and maintain their temporal characteristics, different feature matrices are fed into separate convolutional neural networks. The burst matrix Mk of the k-th sample is processed through the convolutional layer to generate a new set of burst matrices Zk, which have a fixed number and reduced dimensions. These are then passed to the next layer for further processing.

The second layer is the self-attention layer, which is used to learn bursty global features and provide the basis of global contextual information for each feature point, in order to make up for the defect that convolutional neural networks can only perform local feature learning. The structure of the self-attention layer is represented as follows.

Firstly, feature spreading is performed to generate the query vector Q, key vector K and value vector V as shown in equation (6).

$$Q = Z^k W_Q, \quad K = Z^k W_K, \quad V = Z^k W_V \tag{6}$$

Next, calculate the attention scores to determine the importance of each feature point relative to the other feature points, as shown in equation (7).

$$A = \mathrm{Softmax}\left(\frac{QK^T}{\sqrt{D}}\right) \tag{7}$$

Here, the similarity between each pair of feature points is calculated. The factor of QK^T is used for numerical stability. Using function $\sqrt{D}$, the attention scores for each feature location can be normalized.

Then, generate the self-attention features, which, when fused with the aforementioned convolutional features, can preserve the spatial structure of the features, as shown in equation (8).

$$Z_{attention}^{k} = AV \tag{8}$$

Finally, the self-attention features are fused with the convolutional features, as shown in equation (9).

$$Z_{final}^{k} = \alpha Z^{k} + (1-\alpha) Z_{attention}^{k} \tag{9}$$

Here, αrepresents a learnable parameter used to balance the contributions of the two features, Z^k and $Z_{attention}^k$.

After the self-attention layer, a fully connected layer is used to integrate all the category-discriminative information from the self-attention layer. The features output by the fully connected layer are represented by vector F^k, which are then input into a softmax classifier to obtain the predicted probability vector $\widehat{A^k}$. This probability vector represents the likelihood that M^kbelongs to a certain traffic class, as shown in equation (10).

$$\hat{A}^{k} = \mathrm{softmax}(\hat{F}^{k}) \tag{10}$$

The loss between the predicted label and the true label is calculated using the cross entropy function as a loss function as shown in Equation (11).

$$Loss = -\frac{1}{N}\sum_{i=0}^{N}\sum_{a=0}^{A-1}\left\{I\left(\hat{A}^{k}=a\right)\log\hat{A}^{k}\right\} \tag{11}$$

Here, N represents the total number of samples, and I indicates whether the predicted label matches the true label.

4 Experimental evaluatione

This section is used to evaluate the validity of bursting matrices, describing the dataset and experimental setup, and evaluating bursting matrices experimentally on the same dataset with other methods.

4.1 Experimental Setup

In order to test the methodology proposed in this paper, a packet pacp file from UNB capture: ISCX Tor-nonTor dataset (ISCX-Tor) was used. ISCX-Tor contains seven

types of captured traffic. and experimented on a service with an intel core i9-9900k CPU and NVIDIA GeForce RTX 2080 Ti GPU.

In order to fully understand the effectiveness of the proposed method, it is compared with the following three methods. DIDarknet[18], which uses CICFlowMeter to extract 80 statistical features from each network flow of darknet traffic, and then employs an RF-based feature selection method to select the most important features, which are then converted to grey-scale images and fed into a 2D-CNN for classification. Flow-Pic[4], which slices the flow sequences to generate grey scale images which are fed into a CNN model for classification. ETC-PS [15], uses a sequence of session packet lengths to construct traffic paths to represent the interaction between the client and the server, then performs path transformations to show its structure and obtain different information, and finally computes multi-scale path signatures as a distinguishing feature to train traditional machine learning classifiers.

The goal of the classifier is to accurately identify more encrypted streams and avoid misclassification. The criteria used to measure the classifiers include Precision, Recall and F1. The mathematical metrics for the above metrics are as follows.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{12}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{13}$$

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{14}$$

where TP refers to the number of samples that tested positive and were actually positive; FP refers to the number of samples that tested negative and were actually positive FN refers to the number of samples that tested positive and were actually negative. Precision indicates the proportion of correctly predicted positive cases to the proportion of data predicted to be positive, Recall indicates the proportion of correctly predicted positive cases to the proportion of data that were actually positive, and F1 is the F1 is a combination of Precision and Recall.

4.2 Performance Evaluations

This subsection investigates the accuracy of different classifiers on the same dataset. Then the parameter settings of the burst matrix method are analysed in detail.

Table 1 shows the performance metrics of burst matrix and other compared methods respectively. It can be seen that burst matrix outperforms the other methods and has the highest accuracy with an average F1 of 0.9570.7 Tor types are classified with an accuracy of more than 90%, 5 of which reach 95%.

Table 1. Indicators of comparative methods

Tor	DIDarknet		ETC-PS		Flow-Pic		Burst Matrix	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Audio	0.8615	0.7183	0.8629	0.8421	0.9955	0.9152	0.9618	0.9767
Browsing	0.9702	0.8906	0.8806	0.8381	0.7440	0.4624	0.9733	0.9932
Chat	0.7653	0.6902	0.8771	0.8647	0.9788	0.9762	0.9467	0.9930
File	0.9498	0.9435	0.8850	0.8991	0.8398	0.8958	0.9267	0.9789
Email	0.6811	0.8039	0.8912	0.8865	0.8150	0.9757	0.9521	0.8934
P2P	0.6145	0.7682	0.9013	0.9268	0.5854	0.3590	0.9267	0.9205
Video	0.9809	0.9778	0.9011	0.8714	0.9063	0.4205	0.9933	0.9241
F1 score	0.8001		0.8802		0.8981		0.9570	

Among the compared methods, DIDarknet has the worst performance, with an average F1 of only 0.8001, due to the fact that most packets in Tor are transmitted with a fixed maximum length, making it difficult for packet statistical features to distinguish between different types of Tor traffic. While DIDarknet is 0.1179 more effective than ETC-PS, this is due to the difference in classifiers, deep learning methods can automatically extract deeper discriminative features, which may include features that are not counted manually. Flow-Pic's average F1 reaches 0.8981, but it is poorly discriminative for some classes, such as Browsing's Precision and Recall are 0.7440 and 0.4624, respectively, while P2P is 0.5854 and 0.3590.The results show the superiority of this method in the Tor flow classification problem.

The reason why burst matrices work better is that by aggregating temporal and length features of Tor traffic and extracting them over local spatio-temporal features, the dynamic changes and patterns of traffic can be captured more effectively, can be adapted to the characteristics of fixed-length packets in Tor traffic, and can be used to find recognisable features in a wider range of traffic data.

4.3 Parameter evaluation

This subsection determines the number of bursts, matrix dimensions, and burst thresholds for the burst matrix and evaluates the trade-off between classification accuracy and efficiency.

Number of Bursts. Firstly, the matrix dimension is fixed to 30×30 and the number of bursts is varied at the same time. The model training results are shown in Table 2. The feature matrix generated using only the first 5 bursts achieves 88. 09% accuracy. As the number of bursts increases, both FET and CTT become larger accordingly and the accuracy increases. The number of bursts is determined as 15 to achieve higher accuracy with appropriate time overhead.

Matrix Dimension. Fixing the number of bursts and setting different matrix dimensions, the training results are shown in Table 3. With the increase of matrix dimension, the growth rate of CTT is basically unchanged, the growth rate of FET is getting larger,

and the accuracy is improving. It can be found that the optimal value of matrix dimension is 30, when the accuracy basically reaches the highest value, and the cost of FET and CTT is low.

Burst Thresholds. Different burst thresholds were used to train the model (from 0.01 to 0.06 with an interval of 0.01). The results are shown in Fig. 5, where F1, Precision and Recall increase with the increase of burst threshold and reach the maximum value at 0.04s and then start to decrease. Therefore, 0.04s is selected as the burst threshold.

Training Rounds. Figure 6 shows the F1, Precision and Recall of the classifier for different rounds. The model achieves F1 of 0.8993 in only one round. Generally, increasing the number of training rounds helps to improve the classification accuracy. When the number of training rounds is greater than 10, the increment of the three evaluation metrics becomes slow. Precision decreases after 15 training rounds. Therefore, epochs are determined to be 15.

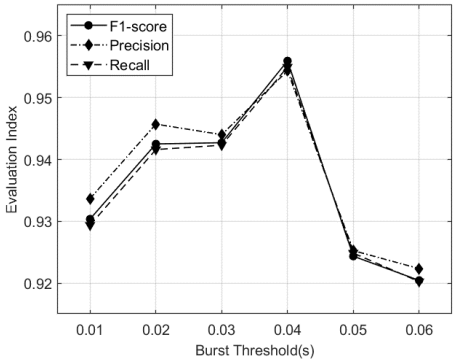


Fig. 5 Effect of burst threshold on F1, Precision and Recall

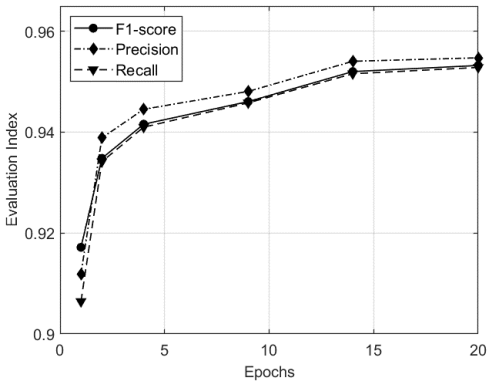


Fig. 6 Effect of training rounds on F1, Precision and Recall

Table 2. Effect of number of bursts n on FET (Feature Extraction Time), CTT (Classifier Training Time), and Accuracy

Bursts	5	10	15	20	25	30
FET(s)	77. 44	123. 11	182. 07	284. 12	333. 58	440.28
CTT(s)	277. 71	400.73	490.87	576. 41	668, 90	800.26
Accuracy	0.8809	0.9323	0.9570	0.9541	0.9501	0.9579

Table 3. Effect of feature matrix dimension m on FET (Feature Extraction Time), CTT (Classifier Training Time), and Accuracy

Dimension	10×10	15×15	20×20	25×25	30×30	35×35
FET(s)	31. 23	44. 21	68. 77	123. 27	182. 07	339. 14
CTT(s)	456. 64	458. 51	467. 29	473. 04	490.87	507. 04
Accuracy	0.9162	0.9308	0.9357	0.9405	0.9570	0.9579

5 Conclusions

This paper proposes a burst feature-based method for Tor traffic identification, leveraging the distinct burst-level characteristics of various Tor traffic types. The approach integrates a deep learning model composed of a Convolutional Neural Network (CNN) and a self-attention layer to achieve accurate classification of Tor traffic. By addressing the limitations of previous models in capturing global features effectively, the method requires only temporal and length features of Tor traffic, along with a simple preprocessing step. When evaluated on the ISCXTor2016 dataset, the method achieves an F1-score exceeding 95%, outperforming existing Tor traffic identification technique.

6 Acknowledgment

This work was financially supported by the New Generation Property Rights Trading System Project of Anhui Provincial Property Rights Trading Centre. We would like to thank the technical team of Anhui Property Rights Trading Centre for their in-depth discussion and provision of the experimental environment, Professor Xu Yinlong of the University of Science and Technology of China for his guidance on this work, and the reviewers for their professional comments.

References

1. Adewopo, V., Gonen, B., Varlioglu, S., Ozer, M.: Plunge into the underworld: A survey on emergence of darknet. In: Editor, F., Editor, S. (eds.) 2019 International Conference on Computational Science and Computational Intelligence (CSCI), pp. 155–159. IEEE (2019).
2. Al-Nabki, M.W., Fidalgo, E., Alegre, E., Fernández-Robles, L.: Torank: Identifying the most influential suspicious domains in the tor network. Expert Systems with Applications 123, 212–226 (2019).
3. Lin, K., Xu, X., Gao, H.: TSCRNN: A novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of IIoT. Computer Networks 190, 107974 (2021).
4. Shapira, T., Shavitt, Y.: FlowPic: A generic representation for encrypted traffic classification and applications identification. IEEE Transactions on Network and Service Management 18(2), 1218–1232 (2021).
5. Xie, G., Li, Q., Jiang, Y.: Self-attentive deep learning method for online traffic classification and its interpretability. Computer Networks 196, 108267 (2021).
6. Pacheco, F., Exposito, E., Gineste, M., Baudoin, C., Aguilar, J.: Towards the deployment of machine learning solutions in network traffic classification: A systematic survey. IEEE Communications Surveys & Tutorials 21(2), 1988–2014 (2018).
7. Kumar, S., Vranken, H., van Dijk, J., Hamalainen, T.: Deep in the dark: A novel threat detection system using darknet traffic. In: Editor, F., Editor, S. (eds.) 2019 IEEE International Conference on Big Data (Big Data), pp. 4273–4279. IEEE (2019).
8. Aggarwal, A., Grunwald, D.: Effects of network bandwidth on performance in software DSM systems. In: Editor, F., Editor, S. (eds.) High-Performance Computing and Networking: International Conference and Exhibition Amsterdam, The Netherlands, April 21–23, 1998, pp. 638–647. Springer Berlin Heidelberg (1998).
9. Pour, M.S., Mangino, A., Friday, K., Rathbun, M., Bou-Harb, E., Iqbal, F., Ghani, N.: On data-driven curation, learning, and analysis for inferring evolving internet-of-things (IoT) botnets in the wild. Computers & Security 91, 101707 (2020).
10. Shen, M., Zhang, J., Zhu, L., Xu, K., Du, X.: Accurate decentralized application identification via encrypted traffic analysis using graph neural networks. IEEE Transactions on Information Forensics and Security 16, 2367–2380 (2021).
11. Dong, C., Zhang, C., Lu, Z., Liu, B., Jiang, B.: CETAnalytics: Comprehensive effective traffic information analytics for encrypted traffic classification. Computer Networks 176, 107258 (2020).
12. Shen, M., Liu, Y., Zhu, L., Du, X., Hu, J.: Fine-grained webpage fingerprinting using only packet length information of encrypted traffic. IEEE Transactions on Information Forensics and Security 16, 2046–2059 (2020).
13. Pinheiro, A.J., Bezerra, J.D.M., Burgardt, C.A., Campelo, D.R.: Identifying IoT devices and events based on packet length from encrypted traffic. Computer Communications 144, 8–17 (2019).
14. Iliadis, L.A., Kaifas, T.: Darknet traffic classification using machine learning techniques. In: Editor, F., Editor, S. (eds.) 2021 10th International Conference on Modern Circuits and Systems Technologies (MOCAST), pp. 1–4. IEEE (2021).
15. Xu, S.J., Geng, G.G., Jin, X.B., Liu, D.J., Weng, J.: Seeing traffic paths: Encrypted traffic classification with path signature features. IEEE Transactions on Information Forensics and Security 17, 2166–2181 (2022).
16. Rao, Z., Niu, W., Zhang, X., Li, H.: Tor anonymous traffic identification based on gravitational clustering. Peer-to-Peer Networking and Applications 11, 592–601 (2018).

17. Sirinam, P., Mathews, N., Rahman, M.S., Wright, M.: Triplet fingerprinting: More practical and portable website fingerprinting with n-shot learning. In: Editor, F., Editor, S. (eds.) Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp. 1131–1148 (2019).
18. Habibi Lashkari, A., Kaur, G., Rahali, A.: Didarknet: A contemporary approach to detect and characterize the darknet traffic using deep image learning. In: Editor, F., Editor, S. (eds.) Proceedings of the 2020 10th International Conference on Communication and Network Security, pp. 1–13 (2020).
19. Singh, D., Shukla, A., Sajwan, M.: Deep transfer learning framework for the identification of malicious activities to combat cyberattack. Future Generation Computer Systems 125, 687–697 (2021).