Adaptive Malware Detection in Network Traffic Using Few-Shot In-Context Learning

Abhinandan Dubey
abhinandandubey@live.com

Abstract

Traditional malware detection systems often rely on signature-based methods or static analysis techniques to identify known malware instances. However, these approaches struggle to keep pace with the rapid evolution of malware, as attackers continuously develop new variants and evasion techniques. This work explores various state-of-the-art transformers and large language models for malware detection based on network traffic captures. It also presents a novel approach for classifying network traffic records in a Few-Shot In-Context Learning scenario using GPT3 model with the 'DSPy' library. The results demonstrate the effectiveness of the few-shot learning model in classifying network traffic, highlighting its potential for real-world application in network security.

1 Introduction

Traditional methods may not effectively detect zero-day attacks or previously unseen malware. In contemporary malware detection systems, machine learning plays a pivotal role in enabling adaptability and efficacy. To address these limitations, there is a growing interest in developing adaptive malware detection systems that can effectively identify novel and evolving malware threats in network traffic. Feature engineering remains a cornerstone of malware detection, involving the extraction of relevant features from raw data to represent network traffic or malware samples effectively. Feature selection techniques further enhance detection accuracy by identifying the most discriminative features, thereby streamlining the learning process and improving model interpretability.

Deep learning models, notably convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have gained prominence for their ability to automatically extract intricate patterns from raw data, at a compromise for explainability. These models excel in capturing complex relationships within network traffic or malware binaries, thereby enhancing detection accuracy. Transfer learning has garnered attention for its ability to leverage pre-trained models on large-scale datasets to bootstrap learning on smaller, domain-specific datasets. By transferring knowledge from related tasks or domains, transfer learning enhances the generalization and adaptability of malware detection models, especially in scenarios where labeled malware samples are scarce, or modeling and defining what constitutes "normal" behaviour might be an intractable problem. This is where semi-supervised and unsupervised learning approaches are pivotal when labeled malware samples are limited. These techniques, such as self-training, co-training, and clustering-based methods, identify patterns and anomalies in network traffic or malware binaries without heavy reliance on labeled data, thus facilitating robust detection even in data-scarce environments.

These methods are often combined or adapted to suit specific malware detection scenarios, considering factors such as dataset characteristics, computational resources, and deployment constraints. As the field of machine learning continues to evolve, researchers and practitioners explore novel techniques and frameworks to enhance the effectiveness and adaptability of adaptive malware detection systems. Using few-shot in-context learning for detecting malware using network traffic involves leveraging the ability of few-shot learning models to generalize from limited examples and contextual understanding of the network traffic.

1.1 Task Definition

Our approach involves using network traffic to identify presence of malware using fine-tuned large

language models and exploring the possibility of few-shot in-context learning, which leverages deep learning techniques to learn from a small number of labeled examples within the context of a broader dataset.

Definition 1.1 *Given a dataset*

$$D = \{r_1, r_2, ..., r_N\}$$

where r_i represents a record in network traffic, the task is to learn a function $f : R_k \rightarrow Y$ that accurately predicts if a record r_t is malicious.

2 Prior Literature

Malicious traffic refers to network traffic that is carefully designed to attack a network, system, or application. Hackers usually use malicious traffic to carry out various network attacks, such as denial-of-service attacks, data theft, and malware distribution, thus causing severe threats and losses to the economic interests and information security of enterprises, governments, and users. In recent years, malicious software commonly employs traffic encryption techniques in order to circumvent the detection of firewalls and antivirus software. Almost all malware currently uses traffic encryption techniques, which pose a great challenge to malicious traffic detection. Therefore, the research and development of effective malicious traffic detection and defense techniques are of critical importance for cyberspace security.

Existing methods for traffic identification can be grouped into four types: port-based, constructive fingerprinting techniques, statistical methods, and deep learning models [9]. Port-based methods rely on fixed port numbers assigned to applications, but modern programs employ techniques such as random port policies and Network Address Translation protocols, making port-based classification increasingly inappropriate for modern network environments. Fingerprint construction methods, such as Deep Packet Inspection [6] and FlowPrint [7], analyze past attack traffic data and extract specific strings or unencrypted information as traffic fingerprints for identification. However, these approaches are highly reliant on plaintext information.

Statistical methods, such as AppScanner [12] and BIND [13], utilize distinctive traffic features and machine learning models for classification. These methods are lightweight and can effectively classify encrypted traffic, but they require specially designed features, which makes them time-consuming, costly, and laborious [4].

There are various previous works which focus on employing deep learning techniques to improve network traffic classification and malicious traffic detection. While they share the common goal of enhancing accuracy and efficiency in these tasks, each paper presents a unique approach and model architecture.

The paper "Identification of encrypted and malicious network traffic based on one-dimensional convolutional neural network" by Zhou et al. (2023) concentrates on identifying both encrypted and malicious network traffic. They propose the HexCNN-1D model, which combines normalized processing and attention mechanisms with a one-dimensional convolutional neural network (CNN). The authors introduce Global Attention Block (GAB) and Category Attention Block (CAB) to handle detailed data information of encrypted and malicious traffic categories. Experiments are conducted in four different network environments to identify conventional, encrypted, malicious, and mixed traffic.

Similarly, the work "Research on Anomaly Network Detection Based on Self-Attention Mechanism" by Hu et al. (2023) aims to improve the efficiency and accuracy of network traffic anomaly detection. However, their approach differs from Zhou et al. (2023) as they focus on feature engineering to construct a comprehensive dataset (DNTAD) by re-extracting and designing feature description sets from raw traffic data. They develop a detection algorithm model based on Long Short-Term Memory (LSTM) and a recurrent neural network self-attention mechanism, evaluating its performance on the constructed DNTAD dataset and comparing it with other advanced methods.

Deep learning-based methods, such as DF [14], FS-NET [15], Deeppacket [16], TSCRNN [17], and wang's approach [5], have shown excellent performance in traffic classification. These methods can automatically extract features from raw traffic and attain high classification accuracy. However, existing deep learning-based approaches rely on a large amount of labeled data, and the imbalance of the dataset can significantly impact the classifier's performance [4]. Our literature review demonstrates growing interest in applying deep learning techniques to network traffic analysis. By

proposing unique model architectures and data preprocessing methods, each paper contributes to the advancement of accurate and efficient network traffic classification and malicious traffic detection. However, none of the papers here have tested the ability of in-context learning in a few shot scenario, and that is one possibility for detecting anomalies which could be indicative of malicious behaviour in network traffic data.

In contrast to the aforementioned approaches, our research focuses on few-shot learning techniques for network traffic classification. We also explore the effectiveness of fine-tuned large language models such as BERT, XLNet, and explore GPT3 with learning from a limited number of labeled examples, aiming to address the challenges of relying on large labeled datasets and specially handcrafted features.

3 Data

We are using two primary datasets for this task -

1. **Stratosphere CTU-13**: The CTU-13 is a dataset of botnet traffic that was captured in the CTU University, Czech Republic, in 2011. The goal of the dataset was to have a large capture of real botnet traffic mixed with normal traffic and background traffic. The CTU-13 dataset consists in thirteen captures (called scenarios) of different botnet samples. On each scenario they have executed a specific malware, which used several protocols and performed different actions.

2. **USTC-TFC2016**: Wei et al. collected the USTC-TFC2016 dataset, which comprises both normal and malware traffic. The normal traffic portion consists of 10 types of application traffic generated using the IXIA BPS network traffic emulation device. These applications can be categorized into 8 groups based on the specific activities they involve. The malware traffic portion is partially derived from public websites and was collected by CTU researchers between 2011 and 2015. To ensure the dataset's uniformity, the authors truncated excessively large traffic samples and combined the smaller ones, resulting in 10 distinct types of malware traffic.

   Because of limited time and resources such as GPU credits, we could only use the above datasets. However, there is scope to use additional datasets such as **DEFCON CTF PCAPs** which includes PCAP files from capture-the-flag (CTF) competitions and challenges organized by DEFCON. One could also explore **NETRESEC PCAP Data** which contains several public packet capture (PCAP) repositories, which are freely available on the Internet. It includes network traffic from exercises and competitions, such as Cyber Defense Exercises (CDX) and red-team/blue-team competitions. It also includes PCAP files which capture malware traffic from honeypots, sandboxes or real world intrusions.

Sample Data


```
{
    "http_method": "GET",
    "user_agent": "Mozilla/5.0 (
        Macintosh; Intel Mac OS
        X 14_0) AppleWebKit
        /605.1.15 (KHTML, like
        Gecko) Version/16.5
        Safari/605.1.15",
    "host": "10.100.123.53",
    "http_referer": null,
    "url_requested": "http
        ://10.100.123.53/lib/
        upgrade.txt",
    "query_parameter": null,
    "cookie": null,
    "cookie_pair": null,
    "http_content_type": null,
    "http_data": null,
    "is_valid": true,
    "additional_info": "http.
        request_number: 1",
    "source_address":
        "10.128.0.209",
    "destination_address":
        "10.100.123.53",
    "timestamp": 1708188225,
    "is_malicious": 0,
    "tcp_src_port": "34775",
    "tcp_dst_port": "80"
}
```


4 Model

We have explored various models in our study. The extracted features from PCAP files, such as content type headers, protocols, URLs etc were converted into a JSON format which is fed into various models. This includes **BERT**, **RoBERTa**, **XLNET**, and **T5** in a fine-tuning setting, and the **GPT3** model in a few-shot learning scenario. For BERT, we leverage the bidirectional context captured the model so it is able learn to identify patterns and classify packets indicative of malicious traffic. The

Table 1: Evaluation Results for Different Models in Fine-Tuning Setting

Model	Accuracy	Precision	Recall	F1 Score
BERT	0.988	0.979	1.000	0.990
XLNet	0.992	0.986	1.000	0.993
RoBERTa	0.992	0.986	1.000	0.993
T5	1.000	1.000	1.000	1.000

Table 2: Evaluation Metrics for Different Values of k in Few-Shot ICL Setting

k	Accuracy	Precision	Recall	F1 Score
5	0.860	1.000	0.748	0.856
10	0.930	0.963	0.909	0.935
20	0.965	0.953	0.986	0.969
50	1.000	1.000	1.000	1.000

fine-tuned BERT model is used to classify new, unseen traffic as either malicious or benign based on the extracted features. We have also explored **XLNet**, in a similar fashion, with its permutation language modeling objective, which was adapted for malicious network traffic prediction. By fine-tuning XLNet on our labeled dataset, the model learns to identify malicious packets in network traffic. We have also explored **T5** model by framing the problem as a text-to-text task. T5 was similarly fine-tuned on a labeled dataset, where the input is the text representation of the extracted features, and the output is the corresponding label (`MALICIOUS` or `BENIGN`).

5 Methods

5.1 Data Preprocessing

Packets are parsed from PCAP files using our novel network traffic parsing framework which extracts information from Layer 4 and Layer 7. The framework yields a dataset, consisting of HTTP packet contents, which is saved as a JSON file. Each record in the dataset contains various fields such as HTTP method, user agent, host, URL, and other relevant information. The dataset is split into training and testing sets. To prepare the data for training and testing the malware detection models, we developed a pre-processing pipeline to extract HTTP information from network packet captures (PCAP files). The pre-processing code utilizes the `pyshark` library to capture packets and extract relevant fields such as HTTP method, user agent, host, URL requested, query parameters, cookies, content type, and more. Each packet is labeled as either malicious or benign based on the source PCAP file. We also evenly split the data into training and testing sets based on a 75% training and 25% testing ratio. The resulting JSON files serve as the pre-processed dataset for training and evaluating the malware detection models. This pre-processing pipeline ensures that the relevant HTTP information is extracted from the network traffic data and labeled appropriately for training and testing purposes.

5.2 Model Training

5.2.1 Fine-Tuning LLM

A custom dataset class, HTTPDataset, is defined to handle the JSON data and prepare it for training. The pre-trained BERT model and tokenizer are loaded, and the model is fine-tuned on the HTTP traffic dataset using the Hugging Face Transformers library. The fine-tuned BERT model is used for inference on new data. Similar to the BERT-based approach the pre-trained XLNet model and tokenizer are loaded, and the model is fine-tuned on the HTTP traffic dataset. This process is done for BERT, RoBERTa, XLNET and T5 models.

5.2.2 Few-Shot In-Context Learning (FS-ICL) Approach

We utilize the `DSPy` library for this approach. A class `PacketClassifierSignature` is defined to specify the input and output fields for the packet classification task. The class `PacketClassifierFewShot`, which inherits from `dspy.Module`, represents the few-shot learning model for packet classification. Training and testing examples are created using `dspy.Example`, containing the packet and its

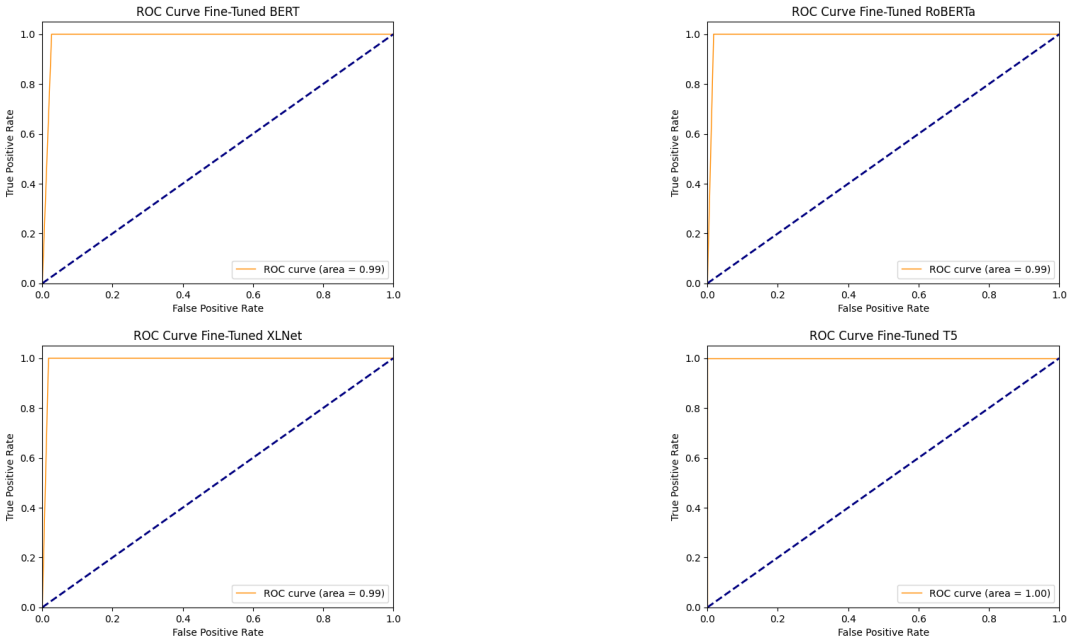

Figure 1: ROC Curves for various models on fine-tuning task

corresponding label. Predictions are generated using the compiled few-shot models on the test set, and the model's performance is evaluated for each value of k (number of examples provided)

6 Results

Our models are evaluated on a test set, which includes packets completely unseen by the model. The results have been summarized in Table 1 and 2. Several performance metrics are calculated to assess the model's effectiveness. Let TP, TN, FP, and FN denote the number of true positives, true negatives, false positives, and false negatives, respectively. The metrics are defined as follows:

- Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision: $\frac{TP}{TP+FP}$
- Recall: $\frac{TP}{TP+FN}$
- F1 Score: $2 \times \frac{Precision \times Recall}{Precision+Recall}$

7 Analysis

We can use the Receiver Operating Characteristic (ROC) curve to analyze and visualize the model's performance at different k values (where k is number of examples in the few-shot scenario). The ROC curve illustrates the trade-off between the true positive rate (TPR) and the false positive rate (FPR) as the classification threshold varies. The TPR and FPR are defined as follows:

- True Positive Rate (TPR): $\frac{TP}{TP+FN}$
- False Positive Rate (FPR): $\frac{FP}{FP+TN}$

The area under the ROC curve (AUC) is computed to quantify the model's discriminatory power. From 2 we can see that the model performance increases as it sees more examples.

8 Conclusion

The research presented in this paper demonstrates the effectiveness of the various NLU techniques and state of the art models such as BERT for the malware detection task, along with a GPT3 based model which can classify HTTP traffic as MALWARE or BENIGN in a few-shot scenario. However, there are several avenues for future work to further enhance the proposed approach:

1. Exploring the impact of including additional features or representations of the HTTP records, such as leveraging domain knowledge or feature engineering techniques, to improve classification performance.

2. Investigating the scalability and real-time applicability of a RAG (Retrieval-Augmented) model in handling large-scale network traffic data and adapting to evolving malware patterns.

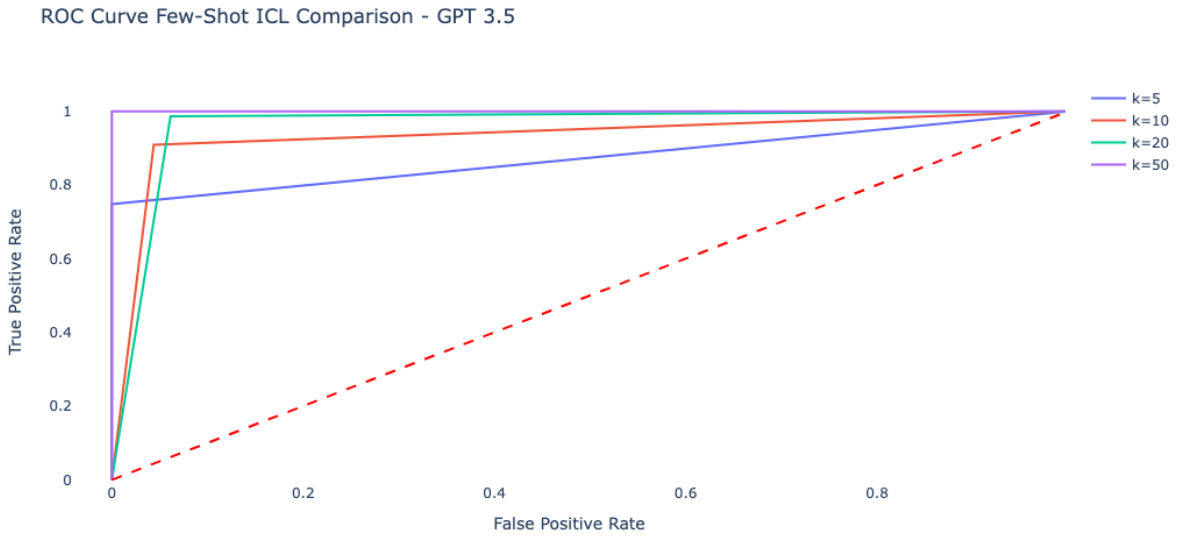

Figure 2: ROC Curves for GPT3 on Few Shot ICL task

3. Comparing the performance of the our models with other state-of-the-art classification algorithms to identify potential improvements and complementary approaches.

4. Conducting further analysis on interpretability and explainability to gain insights into the key factors contributing to the classification decisions.

Known Project Limitations

Network packets can often vary quite a lot in nature. For this reason, it is quite easy to overfit and har dto generalize. While this makes it hard to train task-specific models which can generalize, this pushes us further to explore this in a few-shot in context learning scenario which has shown to perform well. Our dataset can also be expanded significantly to fill this gap. The few-shot learning approaches rely on a relatively small number of labeled examples for training. While this is an advantage in scenarios with limited labeled data, it may impact the generalization ability of the models when applied to larger and more diverse datasets. The performance of the models may not scale well to real-world scenarios with a broader range of network traffic patterns.
Dataset imbalance in real world: Our study does not explicitly address the issue of class imbalance in the real world. If the proportion of malicious and benign traffic samples is significantly skewed, it can affect the model's performance. For our study, we have handpicked sample ratios to make sure we do not have imbalance. The high accuracy and precision scores reported in the results may be influenced by this carefully crafted class distribution, and the models may struggle to correctly classify the minority class (e.g., benign traffic) when the dataset is heavily imbalanced in real world.
Concept drift: Network traffic patterns and malware behaviors evolve over time, leading to concept drift. The models trained on a specific dataset may become less effective as new types of malware emerge or network traffic characteristics change. The study does not discuss strategies for adapting the models to handle concept drift and maintain their performance over time.
Computational resources : Deep learning-based approaches, such as BERT and XLNet, often require significant computational resources for training and inference. The study does not provide details on the computational requirements of the models, which can be a limiting factor in resource-constrained environments or real-time detection scenarios. Also, while the study compares the performance of different few-shot learning approaches, it does not provide a comprehensive comparison with traditional machine learning algorithms or rule-based systems commonly used in network traffic classification.
A note on real-world deployment challenges: The study focuses on the performance evaluation of the models using specific datasets and evaluation metrics. However, it does not address the chal-

lenges and considerations involved in deploying these models in real-world network environments, such as integration with existing security systems, scalability, and runtime performance.

Addressing these limitations in future work can help strengthen the proposed approaches and provide a more comprehensive evaluation of their effectiveness in detecting malware in network traffic using few-shot learning techniques.

Authorship Statement

This study was independently carried out by *Abhinandan Dubey*. The dataset exploration was done through various sources available on online forums. The direction of the project and exploration came from *XCS224U - Natural Language Understanding* course by *Prof. Christopher Potts*, *Stanford University*.

References

[1] Marın, G., Casas, P. & Capdehourat, G. DeepMAL - Deep Learning Models for Malware Traffic Detection and Classification. *CoRR*. **abs/2003.04079** (2020), https://arxiv.org/abs/2003.04079

[2] Srinivasan, S., Ravi, V., Alazab, M. & Kp, S. Network Flow based IoT Botnet Attack Detection using Deep Learning. (2020,7)

[3] Hu, F., Zhang, S., Lin, X., Wu, L., Liao, N. & Song, Y. Network Traffic Classification Model Based on Attention Mechanism and Spatiotemporal Features. (2021,3)

[4] Hu, W., Cao, L., Ruan, Q. & Wu, Q. Research on Anomaly Network Detection Based on Self-Attention Mechanism. *Sensors*. **23** (2023), https://www.mdpi.com/1424-8220/23/11/5059

[5] Zhou, Y., Shi, H., Zhao, Y., Ding, W., Han, J., Sun, H., Zhang, X., Tang, C. & Zhang, W. Identification of encrypted and malicious network traffic based on one-dimensional convolutional neural network. *Journal Of Cloud Computing*. **12** (2023,4)

[6] Liu, X. & Liu, J. Malicious traffic detection combined deep neural network with hierarchical attention mechanism. *Scientific Reports*. **11** (2021,6)

[7] Tayyab, U., Khan, F., Durad, M., Khan, A. & Lee, Y. A Survey of the Recent Trends in Deep Learning Based Malware Detection. *Journal Of Cybersecurity And Privacy*. **2**, 800-829 (2022), https://www.mdpi.com/2624-800X/2/4/41

[8] A. Dainotti, A. Pescape, and K. C. Claffy, "Issues and future directions in traffic classification," IEEE Network, vol. 26, no. 1, pp. 35–40, January 2012.

[9] G. Sun, Y. Xue, Y. Dong, D. Wang, and C. Li, "An novel hybrid method for effectively classifying encrypted traffic," in 2010 IEEE Global Telecommunications Conference GLOBECOM 2010, Dec 2010, pp. 1–5.

[10] P. Velan, M. Cerm ˇ ak, P. ´ Celeda, and M. Dra ˇ sar, "A survey of ˇ methods for encrypted traffic classification and analysis," Netw., vol. 25, no. 5, pp. 355–374, Sep. 2015. [Online]. Available: http://dx.doi.org/10.1002/nem.1901

[11] D. J. Arndt and A. N. Zincir-Heywood, "A comparison of three machine learning techniques for encrypted network traffic analysis," in 2011 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), April 2011, pp. 107–114.

[12] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescap, "Mobile encrypted traffic classification using deep learning," in 2018 Network Traffic Measurement and Analysis Conference (TMA), June 2018, pp. 1–8.

[13] P. Wang, F. Ye, X. Chen, and Y. Qian, "Datanet: Deep learning based encrypted network traffic classification in sdn home gateway," IEEE Access, vol. 6, pp. 55 380–55 391, 2018.

[14] M. J. S. M. S. Mohammad Lotfollahi, Ramin Shirali Hossein Zade, "Deep packet: A novel approach for encrypted traffic classification using deep learning," Available from http://www.arxiv.org, 2017.

[15] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), pp. 43–48, 2017.

[16] and, , and and, "Malware traffic classification using convolutional neural network for representation learning," in 2017 International Conference on Information Networking (ICOIN), Jan 2017, pp. 712–717.

[17] Z. Chen, K. He, J. Li, and Y. Geng, "Seq2img: A sequence-to-image based approach towards ip traffic classification using convolutional neural networks," in 2017 IEEE International Conference on Big Data (Big Data), Dec 2017, pp. 1271–1276.

[18] X. Chen, J. Yu, F. Ye, and P. Wang, "A hierarchical approach to encrypted data packet classification in smart home gateways," in 2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech), Aug 2018, pp. 41–45.

[19] Z.Wang, "The application of deep learning on traffic identification," Available from http://www.blackhat.com, 2015.

[20] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," IEEE Access, vol. 5, pp. 18 042–18 050, 2017.

[21] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu, "Hast-ids: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," IEEE Access, vol. 6, pp. 1792–1806, 2018.