Network Traffic Data Collection for Machine Learning Analysis

James Chao[a] and Ramiro Rodriguez[a]

[a]Naval Information Warfare Center Pacific, San Diego, USA

ABSTRACT

Network traffic has increased substantially due to the introduction of advanced network-enabled applications and devices. The introduction of software defined networks (SDNs) and machine learning (ML) has empowered optimizing network operations and network traffic monitoring, resulting in improved complex traffic operations and security with faster malicious intention detections. This paper focuses on network traffic data collection systems, and the data is evaluated using a survey of ML algorithms, depending on the data type (tabular or image). Adhering to system architecture best practices including a decoupled design to integrate with existing network monitoring infrastructures and cybersecurity standards; and online and offline data collection via packet capture (PCAP) standards. For packet based network traffic data analysis, we convert captured data into images and feed into a convolutional neural network to classify the data based on requirements. For statistical based network traffic data analysis, we apply feature engineering on tabular data and feed into various ML systems to classify based on requirements. Finally, We show that the same ML algorithm outperforms publicly available datasets using our collection method.

Keywords: network traffic classification, machine learning, data collection

1. INTRODUCTION

With the advent of 5G networks, internet of things (IoT), and emerging applications, demands for agile and resilient network management capabilities have been emphasized with the growing complexity of network traffic data. As a result, efficient network resource management has become a key performance requirement to sustain operations over high usage and bandwidth demand. In addition, novel ML methods have been cleverly applied for network operations that require little to no network operator input allowing complex decision making.

Generally, network traffic engineering is applied to address resource allocation challenges. In particular, when faced with constrained network resources, resilient communication protocols require basic network quality of service (QoS) rules to prioritize traffic and dictate what type of data must be transmitted real-time and what type of data can accommodate latency. However, limitations shown in traditional network management techniques face complex challenges due to traffic hidden patterns, security controls, and encryption that affects prediction's accuracy.[1] Network traffic awareness comes relevant to fields related to information security with applications such as intrusion and anomaly detection, as well as encryption and cryptographic applications.[2]

From,[2] five traffic classification methods were identified, and are summarized as follows:

- Statistics-based — Statistical features of traffic.
- Correlation-based — Packet aggregation into flows to classify by correlation between flows.
- Behavior-based — Classification achieved by checking and counting behaviors of a host.
- Payload-based — Check packet payload by Deep Packet Inspection (DPI) and Stochastic Packet Inspection (SPI) for encrypted data.

Further author information: (Send correspondence to J.C.)
J.C.: E-mail: james.chao.civ@us.navy.mil
R.R.: E-mail: ramiro.rodriguez86.civ@us.navy.mil

- Port-based — Compare ports against the Internet Assigned Numbers Authority (IANA).

Payload-based methods rely on packet data, where it is generally a long string of 0 and 1s, although long data strings can be used to apply machine learning (ML) algorithms, it has been shown that converting the payload packet to images and classifying using a convolutional neural network (CNN) can result in improved performance.[3] In this paper, a novel method to capture data from an enterprise-grade cybersecure machine is utilized to capture, preprocess, and convert to images in order to apply network traffic analysis. To generalize and validate our methods, a public dataset[4] of traffic data converted into images is also used, it consists of images of malicious malware and normal network traffic.

Statistics-based methods generally rely on performing a variety of ML algorithms on the features of the network traffic such as ports and IP addresses. In this paper, the dataset[5] was used for statistics-based experiments. The dataset consists of 3.5 million entries that contains 87 features. Each instance holds the information of an IP flow generated by a network device including source and destination IP addresses, ports, inter-arrival times, layer 7 protocol that indicates the application used on that flow as the class. Most of the attributes are numeric type but there are also nominal types and a date type due to the Timestamp. The protocol can be used as the prediction outcome of the ML models to identify the application. Statistics-based data is simple to capture, as it can be simply exported from tools such as Wireshark.[6] Therefore, no novel method was required to perform data collection.

2. RELATED WORK

Sun et al.[7] have demonstrated statistics-based network traffic classification using support vector machines (SVM). This method can update the classifier according to the newly arrived traffic in time. The granularity of its classification is unknown. Wang et al.[8] showed promising results using random forest; they used 20 flow features to aggregate traffic into classes at a specific protocol level. Each tree is generated by iteratively splitting the nodes based on m variables randomly selected from input variables. Al-Obaidy et al.[9] showed using decision trees, also in statistics-based network traffic classification.

Lim et al. and Lopez-Martin et al.[310] have demonstrated payload-based deep packet inspection (DPI) network traffic classification using CNNs and ResNets. Payload-based classification methods are generally one of two types according to the methods used for packet inspection, one is DPI, and the other is Stochastic Packet Inspection (SPI). DPI is a network technology that detects network traffic and packet contents. Because of its high classification accuracy, DPI technology is very popular in traffic management, security analysis and attack prevention. SPI was proposed to deal with the problem that DPI cannot classify encrypted data. This method does not directly use the content of the packet but uses the statistical information of the payload to automatically generate the protocol signature to identify different protocols. The classification method is also highly accurate. Although SPI can classify encrypted data, this method still faces the problem of excessive computational cost.

3. STATISTICAL BASED NETWORK TRAFFIC ANALYSIS

Statistical based can be summarized as typically using the number of packets, the statistical value of the packet size, the packet inter-arrival time, and the total bytes transmitted. This is mainly because these features are simple, and easy to be obtained and handled. In addition, these features also have good stability and robustness.[2]

In this section, data is exported directly from data capture tools, and is preprocessed and applied ML algorithms including linear and Radial Basis Function (RBF) SVMs, decision tree, random forest, and artificial neural networks.

3.1 Data Preprocess

Out of the 85 features, a combination of: user behavior based - idle time, active time. Standard flow based - IP, ports based. data based - packet size, length, flow duration, total bytes were experimented with the ML algorithms. The goal is to predict the protocol, which indicates the application, as seen in the confusion matrix 1, the applications can be determined.

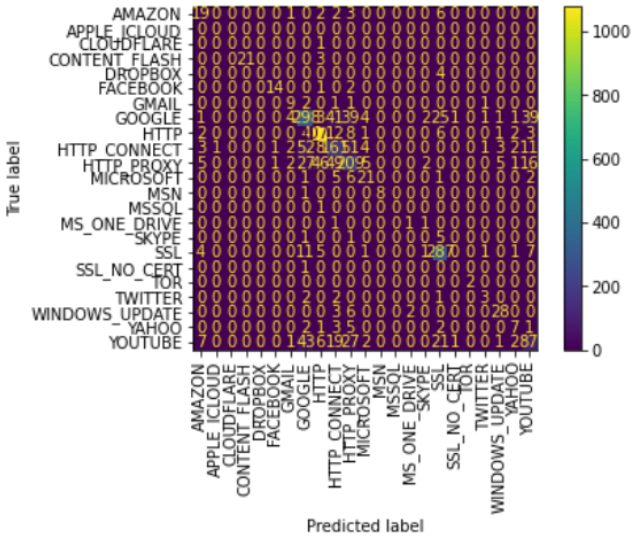

Figure 1. Random Forest Confusion Matrix

Regularization parameter	1
Kernel	Linear & RBF
Probability estimates	False
Tolerance	1e-3
Decision function	one-vs-rest
Tie break	first encountered
Random data shuffling	0

Table 1. SVM Parameters

Of the features, the destination IP and destination port were one hot encoded since they are category type data, the other features were scaled between [0, 1] as it was found to be necessary for SVMs and beneficial to random forests.

The artificial neural network (ANN) requires the labels to be encoded into integers, and feature scaling of the non-categorical data between [-1, 1] improved performance.

3.2 Support Vector Machine

A SVM is a ML model that comprises a kernel function and a decision hyperplane. Generally faster training than ANNs and are grounded in their capacity to converge to the solution for a problem. For the statistics-based dataset, to deal with the scaling issue,[11] a min max scalar is required to stabilize the training process for both linear and RBF SVMs. The hyperparameters are detailed in Table 1.

3.3 Decision Tree

A decision tree is a ML model that maps outcomes in a branching structure, explicitly fitting parameters to direct the information flow. Generally yields very fast training due to model simplicity. We found that scaling had minimal effect on the decision tree. The hyperparameters are detailed in Table 2.

3.4 Random Forest

Random Forest is an ensemble of decision trees where the majority class is used for classification, also very fast training due to its simplicity. Although scaling was not required, it was performed to speed up training time. The hyperparameters are detailed in Table 3.

Criterion	gini
Splitter	best
Max depth	none
Min samples to split	2
Min samples leaf	1
Randomness of estimator	0
Max leaf nodes	none
Min impurity decrease	0
Min Cost-Complexity Pruning	0

Table 2. Decision Tree Parameters

Number of trees in forest	100
Criterion	entropy
Splitter	best
Max depth	none
Min samples to split	2
Min samples leaf	1
Max leaf nodes	none
Min impurity decrease	0
Min Cost-Complexity Pruning	0
Bootstrap samples	true
Number of features for split:	sqrt(n_features)
Randomness of bootstrapping	0
Randomness of sampling features	0

Table 3. Random Forest Parameters

3.5 Artificial Neural Network

ANNs present a more complex architecture and generally cost more to train, but perform well with larger numbers of features and data size. ANNs generally result in faster inference time. The ANN model architecture contains three deep fully connected layers containing 256, 128, and 128 nodes. With relu activation functions and a softmax output, sparse categorical cross entropy loss function, and adam optimizer.

4. PAYLOAD-BASED NETWORK DATA COLLECTION AND ANALYSIS

Payload-based classification algorithms such as DPI, due to its high classification accuracy, are very popular in traffic management, security analysis, and attack prevention. Although classifying encrypted traffic is still ongoing research, for example, SPI generally decreases the accuracy compared to non-encrypted network traffic.[2] As mentioned before, payload packets can be converted into images, and classified using CNNs with good performance. In this section, we set up the network collection, convert to images, perform data reprocess, and classify the data.

Typically, to enable deep learning of packet data and traffic flow, tabular data is converted into images[12131 14]. The data format presented by PCAP data is not usually presented as a form of an image. There are multiple techniques one can apply to represent network statistics, flow, and payload with a computationally easy process, a subroutine negligible to the overall integrated effort.

4.1 Data Collection and Conversion

The setup considers a software defined network (SDN) topology with Virtual Machines (VMs) that accommodate for both real-time and offline network data collection pipeline scheme, which consists of data collection with network to image processing, network traffic classifier and a network controller that interfaces the network classifier and SDN router for network optimization purposes. A high-level diagram of the architecture is shown in Fig. 2.

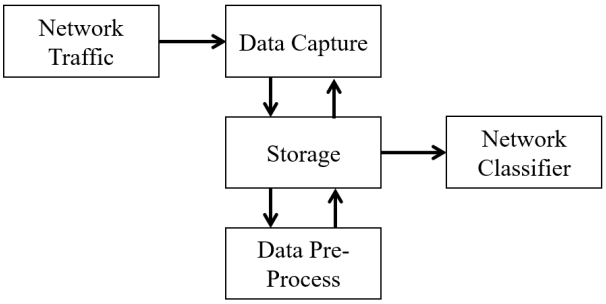

Figure 2. High Level Data Collection Process

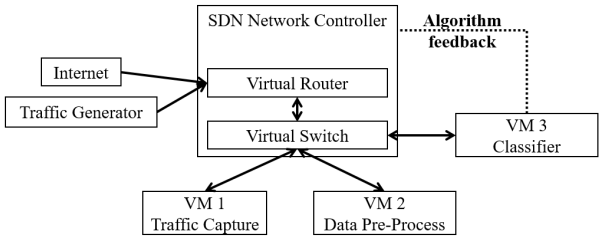

Figure 3. SDN Topology for Data Collection

The data collection component takes care of acquiring and storing network traffic data and the preprocess of network data into images, residing in separate virtual hosts within the SDN, as shown in Fig. 3. As mentioned, this topology allows for the creation of two pipelines: the real-time traffic and the offline traffic. The offline dataset is composed from recorded network traffic, academic datasets, and generated traffic. Both pipelines feed into a separate VM host that houses the data, where raw data is converted into images and fed into the network classifier VM host.

The integration of the network classifier and the SDN network controller is possible via a network bus connecting both domains in a feedback loop. The architecture allows for the classifier algorithm to learn from both real-time and offline pipelines, and feed relevant network information to the SDN controller (see Fig. 3). Any new data protocol or format will be sent to the offline pipeline and stored for learning purposes. This feedback loop, corresponds to the adaptive aspect of our project which helps with the continual online learning required for current complex networked systems.

The separation of modules and virtual machines (VM) in the architecture allows for decoupled micro systems to handle network traffic data sensitivity levels for cybersecurity purposes, improve load balancing and performance optimization by allocating resources appropriate according to packet capture performances, and system segregation to isolate issues or sub-optimal performances.

The construction of the data collection for the offline pipeline is centered at the network resource controller which serves as the collection point between the network router and the storage bin. The storage bin which can be hosted in a virtual machine instance or container, will digest the network data flow extracting the payload and packet statistics which can be transformed to images by a network to image subroutine defined below. Similarly, at the real-time pipeline, the network resource controller can direct the router network information to a processing virtual instance hosted at a container which will digest the network data transforming it to images.

Finally, these images are accessed via a network switch to a virtual host that hosts the ML classifier that will perform either training or testing with its outcomes fed to the SDN network controller for optimization purposes.

A command line instance of Wireshark called TShark[15] can read, process, and digest the network packet flow and information as seen in Fig. 4. That feeds into a custom python script using Pillow (PIL)[16] which converts the data into gray scale images representing the data packets. PCAP network data is converted from the original hex format to its binary expression, a black pixel is generated for a 0, and a white pixel for a 1 from the binary data stream.

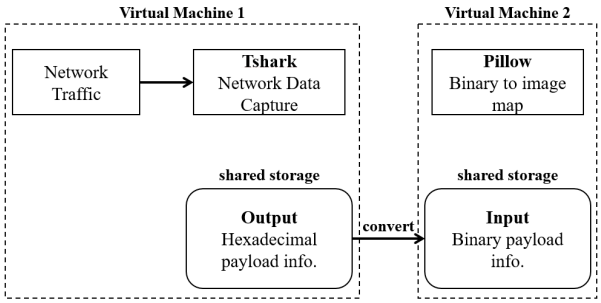

Figure 4. Data Preprocess: Network PCAP to Image

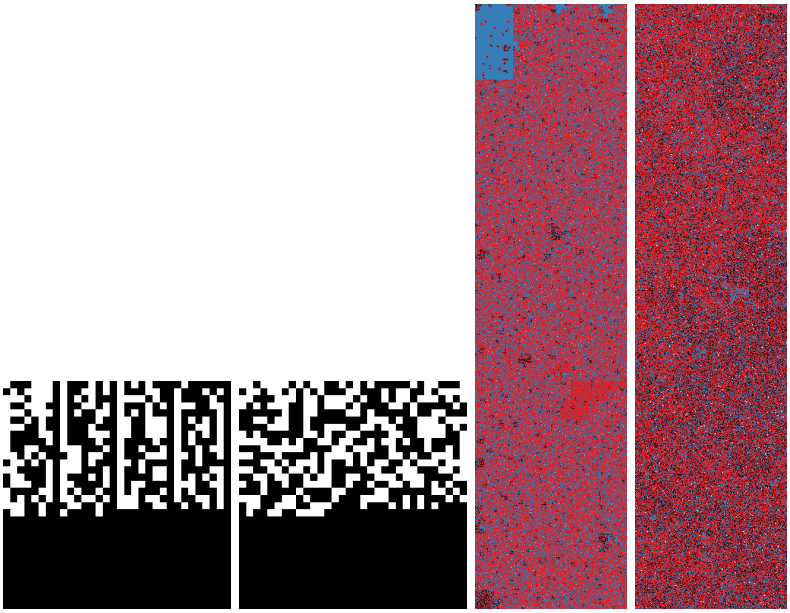

Figure 5. (a) HTTP Traffic Image (b) TLS Traffic Image (c) Malicious Traffic Image (d) Normal Traffic Image

An example of this image can be seen in Fig. 5 and Fig. 5, which shows a dataset that can be categorized into two network protocols; one being transmitted over HTTP and another being transmitted over TCP.

The final data preprocess step is to split the dataset captured from the SDN architecture into training images and testing images.

4.2 Public Dataset

The dataset[4] is utilized to further validate our ML classification algorithms. The dataset consists of malicious and normal network traffic, and is already converted into images, an example of a normal and malicious traffic image is shown in Fig. 5 and Fig. 5.

The data will also be split into training and testing images, and classified using ML algorithms to compare with the online collected dataset classification results.

4.3 Convolution Neural Network

As mentioned, CNNs are a good way to analyze payload packet data after converting to images, Fig. 6 shows the architecture of the CNN, and Table 4 shows the hyperparameters.

Image rescale	1/255
Shear range	0.2
Zoom range	0.2
horizontal flip	True
Target size	64 x 64
Kernel size	3 x 3
Pool size	2 x 2
Stride	2

Table 4. CNN Parameters

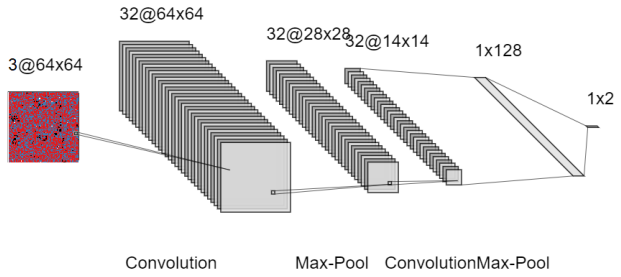

Figure 6. CNN Architecture

4.4 ResNet and VGG

As suggested, a ResNet and VGG was also applied to the image dataset, the results were poor, which is consistent with the findings in,[3] this is due to the size of the datasets, with a smaller dataset like the one we captured and the public dataset, a CNN will generally perform better. This will be revisited with larger datasets in future work.

5. RESULTS

5.1 Statistics-Based

Key metric results are recorded in Table 5, and confusion matrices such as Fig. 1 can be supplemented for each algorithm result.

The feature set of Destination IP, destination Port, and average packet size, with the algorithm of random forest performed the best with highest accuracy of 0.75, reasonable training time of 4.4s, and reasonable dataset size requirements of 10k entries. SVMs performed relatively decently with similar accuracy but longer training time. Both results encourage the future work of quantum enhancements to reduce dataset dimensionality, improve training and inference time, and improve accuracy. Feature set two performed well only with RBF SVM. Likely since linear SVM is a parametric model, and the complexity grows with the size of the training set with more features. Note random forest performed well regardless of which feature set was selected.

5.2 Payload-Based

5.2.1 Real Time Collected Dataset Results

Results from the real time captured dataset shows a high accuracy for classifying network protocol as shown in Table 6 after eight epochs of training.

Not only does the model achieve a high accuracy of 0.9998, the stability is good as the loss and validation loss are decreasing in a steady manner, as shown in Fig. 7 and Fig. 7. Until it reaches a 0.1031 loss and 0.021 validation loss. While the model converged after eight epochs and 808.33s, the model continued to train until 20 epochs and maintained a steady accuracy and loss.

Feature Set	Algorithm	Accuracy	Training Time	Data Size
Destination IP, Destination Port, Average Packet Size	SVM	0.742	91.3s	10k
Destination IP, Destination Port, Average Packet Size	RBF SVM	0.726	101.1s	10k
Destination IP, Destination Port, Average Packet Size	DT	0.748	2.2s	10k
Destination IP, Destination Port, Average Packet Size	RF	0.75	4.4s	10k
Destination IP, Destination Port, Average Packet Size	ANN	0.613	150.41s	100k
Average Packet Size, Forward Packet Length Mean, Backward Packet Length Mean, Total Forward Packets, Total Backward Packets, Flow Bytes	SVM	0.42	1.65s	10k
Average Packet Size, Forward Packet Length Mean, Backward Packet Length Mean, Total Forward Packets, Total Backward Packets, Flow Bytes	RBF SVM	0.726	101.1s	10k
Average Packet Size, Forward Packet Length Mean, Backward Packet Length Mean, Total Forward Packets, Total Backward Packets, Flow Bytes	DT	0.68	0.08s	10k
Average Packet Size, Forward Packet Length Mean, Backward Packet Length Mean, Total Forward Packets, Total Backward Packets, Flow Bytes	RF	0.72	2.4s	10k
Average Packet Size, Forward Packet Length Mean, Backward Packet Length Mean, Total Forward Packets, Total Backward Packets, Flow Bytes	ANN	0.40	33.4s	100k
Destination IP, Destination Port, Idle Mean, Active Mean	SVM	0.742	281.2s	10k
Destination IP, Destination Port, Idle Mean, Active Mean	RBF SVM	0.726	267.3s	10k
Destination IP, Destination Port, Idle Mean, Active Mean	DT	0.745	8.7s	10k
Destination IP, Destination Port, Idle Mean, Active Mean	RF	0.747	14.1s	10k
Destination IP, Destination Port, Idle Mean, Active Mean	ANN	0.607	153.2s	100k

Table 5. Statistics-Based Dataset Results

	Accuracy	Training Time
CNN	0.9998	808.33s

Table 6. Payload-Based Collected Dataset Results

5.2.2 Public Dataset

Table 7 shows the result of the CNN, VGG, and ResNet performance on the payload-based public dataset.

Although a CNN provided good accuracy, the validation loss is unstable as shown in Fig. 7 and Fig. 7. Furthermore, the model accuracy is improving in a general direction, but occasionally the accuracy will deteriorate, creating a non-monotonic training improvement. This validates our data capture and conversion process, as the same ML algorithm outperformed using new collected data. Both in terms of accuracy, loss, and the training stability of the model is achieved faster, with a stable loss after each training epoch, and the accuracy is consistently improving after each training epoch.

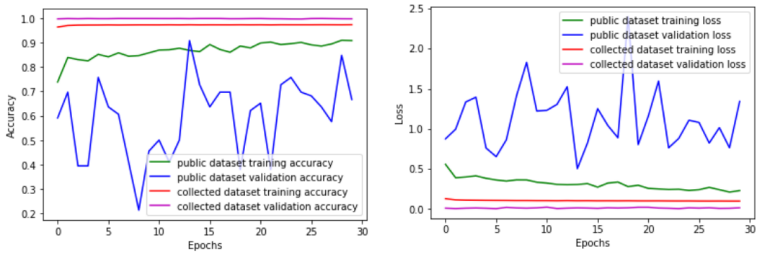

Figure 7. (a) Accuracy (b) Loss

	Accuracy	Training Time
CNN	0.849	144.1s
VGG16	0.394	23s
ResNet	0.269	25s

Table 7. Payload-Based Public Dataset Results

6. CONCLUSION AND FUTURE WORK

In this paper, we set up an architecture to collect network traffic data, and perform analysis using a survey of data preprocessing and ML algorithms on different types of datasets. Specifically, statistic-based and payload-based data are examined. For statistics-based methods, the focus was on applying random forest, decision tree, SVMs, and ANNs on PCAP datasets, while payload-based methods were focused on classifying using CNNs, VGGs, and ResNets on PCAP converted image datasets.

We validated the network traffic data collection architecture by comparing ML analysis results with public datasets, and showed our results outperformed the public dataset. While adding the benefits of a module decoupled based architecture, including resource optimization, data segmentation, and load balancing. Given the improved results, the approach of using ML classification algorithms to analyze the application and its characteristics is also validated, such as ensuring TLS protocol to increase cybersecurity.

For future work, we intend to improve the analysis performance by using quantum machine learning (QML) and quantum inspired machine learning. Specifically, we see benefits of adding quantum convolutional layers to the CNN in payload-based ML neural network architectures, as well as employing quantum SVMs, quantum random forests, and quantum neural networks (QNN) over classical SVMs, random forests, and artificial neural networks in statistics-based ML analysis. Our results encourage the future work of quantum enhancements to reducing dataset dimensionality, improve training and inference time, and improve accuracy. In particular, payload-based and statistics-based methods are interesting to us due to their suitability for QML, our planned future work. Therefore, it is the focus of this work, even though the quantum results are not discussed in this paper.

REFERENCES

[1] Abbasi, M., Shahraki, A., and Taherkordi, A., "Deep learning for network traffic monitoring and analysis (ntma): A survey," *Computer Communications* **170**, 19–41 (2021).

[2] Zhao, J., Jing, X., Yan, Z., and Pedrycz, W., "Network traffic classification for data fusion: A survey," *Information Fusion* **72**, 22–47 (2021).

[3] Lim, H.-K., Kim, J.-B., Heo, J.-S., Kim, K., Hong, Y.-G., and Han, Y.-H., "Packet-based network traffic classification using deep learning," in [*2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*], 046–051 (2019).

[4] Rose, J., "Malicious network traffic pcaps-2021," (March 2021).

[5] Rojas, J. S., Rendón, , and Corrales, J. C., "Consumption behavior analysis of over the top services: Incremental learning or traditional methods?," *IEEE Access* **7**, 136581–136591 (2019).

[6] Lamping, U. and Warnicke, E., "Wireshark user's guide," *Interface* **4**(6), 1 (2004).

[7] Sun, G., Chen, T., Su, Y., and Li, C., "Internet traffic classification based on incremental support vector machines," *Mobile Networks and Applications* **23**, 789–796 (Aug 2018).

[8] Wang, B., Zhang, J., Zhang, Z., Pan, L., Xiang, Y., and Xia, D., "Noise-resistant statistical traffic classification," *IEEE Transactions on Big Data* **5**(4), 454–466 (2019).

[9] Al-Obaidy, F., Momtahen, S., Hossain, M., and Mohammadi, F., "Encrypted traffic classification based ml for identifying different social media applications," in [*2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*], 1–5 (2019).

[10] Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., and Lloret, J., "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access* **5**, 18042–18050 (2017).

[11] Hsu, C.-W., Chang, C.-C., and Lin, C.-J., "A practical guide to support vector classification."

[12] Zhu, Y., Brettin, T., Xia, F., Partin, A., Shukla, M., Yoo, H., Evrard, Y. A., Doroshow, J. H., and Stevens, R. L., "Converting tabular data into images for deep learning with convolutional neural networks," *Scientific Reports* **11**(1), 11325 (2021).

[13] Moreira, R., Rodrigues, L. F., Rosa, P. F., Aguiar, R. L., and Silva, F. d. O., "Packet vision: a convolutional neural network approach for network traffic classification," in [*2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*], 256–263 (2020).

[14] Kim, S. and Reddy, A., "Modeling network traffic as images," in [*IEEE International Conference on Communications, 2005. ICC 2005. 2005*], **1**, 168–172 Vol. 1 (2005).

[15] Merino, B., [*Instant traffic analysis with Tshark how-to*], Packt Publishing Ltd (2013).

[16] Clark, A., "Pillow (pil fork) documentation," *readthedocs* (2015).