Digital Communications and Networks 10 (2024) 676–692

Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.keaipublishing.com/dcan

Network traffic classification: Techniques, datasets, and challenges

Ahmad Azab [a,*], Mahmoud Khasawneh [b], Saed Alrabaee [c,**], Kim-Kwang Raymond Choo [d], Maysa Sarsour [e]

[a] College of Information Technology and Systems, Victorian Institute of Technology, Australia
[b] College of Engineering, Al Ain University, Abu Dhabi, United Arab Emirates
[c] Information Systems and Security, College of IT, United Arab Emirates University, Al Ain, 15551, United Arab Emirates
[d] Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX, 78260, USA
[e] School of Photovoltaic and Renewable Energy Engineering, University of New South Wales, Sydney, NSW, 2052, Australia

ARTICLE INFO

Keywords:
Network classification
Machine learning
Deep learning
Deep packet inspection
Traffic monitoring

ABSTRACT

In network traffic classification, it is important to understand the correlation between network traffic and its causal application, protocol, or service group, for example, in facilitating lawful interception, ensuring the quality of service, preventing application choke points, and facilitating malicious behavior identification. In this paper, we review existing network classification techniques, such as port-based identification and those based on deep packet inspection, statistical features in conjunction with machine learning, and deep learning algorithms. We also explain the implementations, advantages, and limitations associated with these techniques. Our review also extends to publicly available datasets used in the literature. Finally, we discuss existing and emerging challenges, as well as future research directions.

1. Introduction

Reliance on the Internet has significantly increased around the globe, making it an essential part of both individuals' and corporates' daily operations. This increase is due to the Internet technology revolution (fiber and 5G connections), ease of access via different devices, competitive offered prices, and various services provided by the Internet [1]. According to the latest report of the International Telecommunication Union (ITU) [2], Internet users worldwide reached 4.1 billion in 2019, an increase of over 53% compared to 2005 and 5.3% compared to 2018. This has posed a heavy burden over Internet Service Providers (ISPs) to find a solution for network traffic classification and identification of the causal application, protocol, or service to fulfill Quality of Service (QoS), content filtering, lawful interception, and malicious behavior identification objectives. Real-time applications, such as Voice over IP (VoIP) and video conferencing, necessitate low latency in delivering their network traffic, unlike other applications like web browsing. This is fulfilled by applying QoS, which ensures a higher priority to the real-time applications' traffic to be processed by the different network nodes. Various countries' regulations prohibit the usage of specific applications within their boundaries and filter the content accordingly. For example, China prohibited the usage of Skype in 2013 since its protocol did not comply with China's national regulations [3]. Lawful interception is a mandate in many countries where ISPs are obliged by the security authorities to provide the capability of lawful interception when required by the local law enforcement agencies [4–6]. As a result, ISPs must provide the proper infrastructure for lawful interception before operating. Malicious behavior identification has become a crucial goal since the economy shifted into the digital form, prompting the cybercriminals to turn their attention to conducting malicious attacks via the Internet to gain profit [7–9]. *Anti*-malware companies not only rely on host level identification of the malicious behavior, but also integrate the identification of malicious network traffic to improve the detection confidence. For example, the identification of botnet network traffic helps to identify the infected devices as well as track down the cybercriminals [10,11].

The research community proposed solutions and frameworks to fulfill the network traffic classification goal. The solutions addressed the effectiveness of traffic classification in terms of classification accuracy, minimal loss, high throughput, required computational resources, and classification speed. The earliest solution relied on packet port-based classification but became ineffective for applications with port randomization technique. Later, Deep Packet Inspection (DPI) of network traffic

* Corresponding author.
** Corresponding author.
E-mail addresses: ahmad.azab@vit.edu.au (A. Azab), salrabaee@uaeu.ac.ae (S. Alrabaee), raymond.choo@fulbrightmail.org (K.-K.R. Choo).

https://doi.org/10.1016/j.dcan.2022.09.009
Received 21 May 2021; Received in revised form 25 June 2022; Accepted 14 September 2022
Available online 18 September 2022
2352-8648/© 2024 Chongqing University of Posts and Telecommunications. Publishing Services by Elsevier B.V. on behalf of KeAi Communications Co. Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

content has filled the gap of the port-based solution. However, it showed its drawbacks against ciphered network traffic. Recently, the utilization of machine learning algorithms has increased in the literature to classify network traffic without the need to access packets' port numbers or content. It extracts statistical features that represent the behavior of a specific protocol or application flows, which are used for establishing the solution. Supervised, unsupervised, and semi-supervised machine learning frameworks have shown the effectiveness of the traffic classification process, where each one of them has its own strength and weakness. Deep learning algorithms, such as Convolution Neural Networks (CNN), have proven their efficiency through the unnecessity of extracting any statistical feature and through their reliance on the employment of the raw network traffic as their input.

Motivated by the importance of network traffic classification, this study provides a comprehensive survey of the most prevalent traffic classification techniques. Beside the discussion of the port-based and DPI techniques, we provide an in-depth analysis of both the machine learning with statistical features and deep learning solutions since they are considered the current trend of network traffic classification techniques. The literature contains a significant number of published studies regarding the techniques that this survey addresses, but they cannot all be included. Therefore, we have selected the following criteria to choose the published research for each technique in our work: novelty, clear description of the used datasets and algorithms, a high number of citations, explanation of the used and recently published research unless the research is highly cited, and novel. The contributions of this work are summarized as follows:

- Providing a comprehensive discussion of the most deployed techniques for network traffic classification in the literature.
- Explaining the advantages and disadvantages of the discussed techniques.
- Reviewing the relevant research of each technique.
- Highlighting the classification focuses of the proposed solutions in the literature.
- Describing the public datasets deployed by researchers in the literature.
- Discussing the challenges of the techniques that are deployed.
- Providing a systematic survey that fills the gaps of the other proposed surveys in the literature.
- Providing future directions of possible research in the area of network traffic classification.

The rest of the paper is organized as follows. Section 2 provides an overview of the surveys conducted by researchers in the literature and highlights how our study adds knowledge to the field. Sections 3 and 4 introduce the port-based and DPI classification techniques, respectively. They discuss the techniques' workflow coupled with their strengths and weaknesses. Section 5 provides an in-depth explanation of the statistical approach in conjunction with machine learning algorithms. It discusses in detail the usage of supervised, unsupervised, and semi-supervised algorithms in classifying network traffic, along with their current research progress, goals, advantages, and disadvantages. Section 6 describes the deployment of deep learning algorithms in establishing a network classifier without the need to conduct feature engineering processes. It discusses the recent solutions proposed by the research community in the literature. Section 7 summarizes the public datasets that have been used by researchers in the literature. A discussion of the strengths, weaknesses, and challenges of the classification methods are provided in section 8. Finally, our conclusion and future directions are marked in section 9.

2. Related works

During the last few years, researchers have conducted various surveys that discuss the network traffic classification problem from different viewpoints. Finsterbusch et al. [12] presented a survey that mainly aims to analyze the performance of the most prevalent open source DPI module in terms of the attained accuracy and needed computational requirements. The survey compared six DPI modules, namely OpenDPI, nDPI, libprotoident, IPP2P, Hi-Performance Protocol Identification Engine (HiPPIE) and L7-filter, to classify network traffic of various protocols such as SMTP, IMAP, and HTTP. Valenti et al. [13] reviewed the traffic classification solutions, specifically the supervised machine learning algorithms. They highlighted the advantages of the statistical machine learning classifiers in identifying the causal protocol or application. The work also developed two supervised learning classifiers with different feature sets to reflect the importance of the feature selection in building the classifier. The survey of Pacheco et al. [14] aimed to provide an overall overview and the needed procedure required to fulfill the traffic classification goal through the use of machine learning algorithms. The study provided a detailed explanation of each step of the process and highlighted its relevant research. Besides, the study discussed the most common trends and challenges of the discussed reviewed papers. Salman et al. [15] reviewed the traffic classification techniques, particularly the machine learning approaches including supervised, unsupervised, and semi-supervised classifiers. Data collection and presentation methods required for machine learning classifiers have also been discussed. The authors concluded their work with recommendations for the current research. Tahaei et al. [16] addressed IoT device network traffic classification techniques and their characteristics. An overview of the current literature solutions has been conducted, highlighting their strengths and weaknesses. In addition, the research discussed the various datasets used in the literature as well as the challenges faced by the current classification techniques. Wang and Chen [17] highlighted the emerging research on the application of deep learning algorithms to network classification, especially the mobile network traffic. The study provided an overview of the overall general framework of deep learning solution and discussed the recent related work in the literature in terms of data preparation, data pre-processing, input design, and model architecture. Zhao et al. [18] provided a review of the current traffic classification techniques and categorized them according to the deployed representative features. The five categories discussed include port-based, payload-based, correlation-based, behavior-based and statistical-based classifications. For each category, the paper provided an analysis of its workflow, advantages, disadvantages and deployed features.

Although discussing different network traffic classification techniques, the conducted surveys suffer from multiple shortcomings. First, each survey focused on specific classification techniques in its study. Second, the surveys lack discussion about the available public datasets that researchers in their experiential evaluation could utilize. Third, each survey discussed the challenges for the only addressed techniques in their study. Our research differs from the previous conducted surveys as it discusses the most deployed techniques in network traffic classification, including DPI, supervised learning, unsupervised learning, semi-supervised learning, and deep learning algorithms. Furthermore, this survey introduces the widely used datasets generated by computers, mobiles, and IoT. It also discusses the challenges of each solution that degrade their performance. Table 1 compares the surveys in the literature and our work in terms of the reviewed techniques and dataset discussion. This survey covers the gaps in using a single survey as a general reference that discusses all the techniques, reviews the latest relevant literature on each technique, explains their challenges, and highlights the recent deployed public datasets.

3. Port-based identification

The earliest network traffic classification solutions utilized packets' port numbers to classify the network traffic to their correspondence protocols. Internet Assigned Numbers Authority (IANA) has allocated standard port numbers to distinguish among different services or protocols network traffic [19]. Port numbers are classified into three ranges;

Table 1
Survey comparison.

Survey	DPI	Supervised Learning	Unsupervised Learning	Semi-supervised Learning	Deep Learning	Dataset	Year
[12]	✓	×	×	×	×	×	2104
[13]	✓	✓	×	×	×	×	2013
[14]	✓	✓	✓	✓	×	×	2019
[15]	✓	✓	✓	✓	×	×	2020
[16]	✓	✓	✓	✓	×	✓	2020
[17]	×	×	×	×	✓	×	2019
[18]	✓	✓	✓	✓	×	✓	2021
Our Study	✓	✓	✓	✓	✓	✓	2021

system ports (0–1023), user ports (1024–49,151), and the dynamic ports (49,152–65,535).The standard protocol deployment port range is 0–1023. The classification process usually takes place by inspecting the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) packets' port numbers which are mapped to the predefined ports by IANA for classification purposes. For example, if the inspected port number is 80, it corresponds to the HTTP protocol.

Moore and Papagiannaki [20] have assessed the port-based identification approach to classify network traffic into their correspondent protocols, such as HTTP and FTP. Their empirical evaluation provided results with an accuracy of no more than 70%, regardless of the number of packets observed. Madhukar and Williamson [21] evaluated the port-based classification against their private collected Internet traffic in their university's labs. The authors emphasized that the applied solution did not identify 30%–70% of the collected data. Sen et al. [22] addressed the effectiveness of the port-based method to classify P2P network traffic. Their private collected data (including VPN and non -VPN network traffic) indicated that the standard ports accounted for only 30% of the whole dataset and the solution provided low performance.

Port-based classification solution's advantages are reflected in its simplicity to implement the low computing resources requirements and the high-speed classification process. On the other hand, it has two main disadvantages. First, various applications nowadays deploy masquerading, that is, using standard ports to deliver other protocol traffic, such as malware traffic over HTTP. Second, many applications are deployed randomly, which refers to the utilization of non-standard/dynamic ports to deliver their network traffic, such as VoIP applications [23].

4. DPI

DPI approach, sometimes referred to as signature-based identification, overcomes the shortcomings of the port-based classification approach since it does not require accessing port numbers. Therefore, it avoids masquerading and randomization [24]. DPI classifies and analyzes the payload of the network traffic to identify the causal applications. A signature is extracted from the packets' content, including characters, strings, bit pattern, and symbols that classify the applications. A signature library comprises of signature records, each of which is related to an application. This technique allows the classifier to examine the content of a single packet or aggregate packets, and checks them according to a signature library; if a match occurs, then an alert is raised, and the traffic is correlated to an application. This technique enhances the accuracy compared to the port-based identification, even if the application uses non-standard ports.

Wang et al. [25] proposed a framework named Length-Based Matching (LBM) that contains a novel accelerating scheme for RegEx matching. The solution has its own Dual-Finite Automata (DFA) matcher called Stride-DFA (StriD2FA). It differs from traditional matching methods in two aspects. First, LBM works by converting the original byte stream into a shorter stride-length integer stream before forwarding it to StriD2FA. Second, it is not built directly from the traditional RegEx. The evaluation experiments showed an increase in the matching speed and a decrease in the memory consumption compared to the traditional DFA.

Fernandes et al. [26] proposed a framework named lightweight DPI (LW-DPI). The solution's goal was to reduce the overhead of the detection process while preserving acceptable accuracy results. To fulfill this, the solution classifies network traffic by examining the content of a limited number of packets or a fraction of the payload of a given packet. The evaluation is based on network traffic collected from commercial ISPs and local university laboratories, and is marked as multiple network protocols. The accuracy results were up to 99% for P2P and mail network traffic.

Hubballi and Swarnkar [27] proposed a Bitcoding framework, a bit-level DPI signature generation for traffic classification. It analyses the first n bits of the network flow, and the signatures are extracted and converted into a state transition machine for further comparison. Hamming distance has been integrated in their system to reduce collisions and increase the number of targeted applications. Later, the same authors [28] introduced another solution called BitProb that utilizes the concept of the probabilistic bit signatures which are an n-bit binary string length extracted from the network flows to classify network traffic. The solution represents raw network traffic as network flows, and the n-bit signature is generated for each flow by monitoring the first few packets. Afterward, the n-bit is fed to a state transition machine named Probabilistic Counting Deterministic Automata (PCDA) to calculate the probability of the bit signatures and classify them into temporary protocols. For evaluation purpose, three datasets covering 20 protocols have been utilized, achieving low misclassification results.

Although DPI overcomes port-based identification drawbacks, it suffers from other disadvantages. The first disadvantage is the high demand on computing resources since a single packet's or aggregate packets' content must be accessed, processed, stored, and compared to match a signature. Second, DPI does not classify encrypted network traffic. This drops the solution's performance since numerous applications generate encrypted traffic, such as HTTPS and malware. Third, the access of network traffic content represents a breach of the privacy policies or a violation of the privacy legislation in different countries. Table 2 summarizes the DPI solutions discussed in terms of the granularity, algorithm, dataset, and aim. The table shows that the addressed granularity tested to classify packets into their causal protocols or applications, focusing more on protocols.

Table 2
Summary of DPI solutions.

Study	Granularity	Algorithm	Dataset	Aim
[25]	Application	LBM	DARPA, DEFCON, TSINGHUA	Space and time efficiency
[26]	Protocol	LW-DPI	Private	CPU processing time
[27]	Application/ Protocol	Bit-Level DPI	NTRESEC, D-CORPORA, FOI	Reliability
[28]	Protocol	Probabilistic Bit-Level DPI	NTRESEC, D-CORPORA, FOI	Accuracy, low misclassification

5. Machine learning-based traffic statistical classification

Statistical classification solution relies on the statistical feature extraction in combination with machine learning algorithms when classifying network traffic. The first step of this framework is the representation of the network traffic in the form of flows, which is the aggregation of packets that share the 5-tuples; source and destination IP addresses, source and destination port numbers, and TCP or UDP protocol. The second step is the extraction of statistical features at the packet level or flow level. Packet level feature extraction is conducted over single or aggregate packets, resulting in features as packet length and inter-arrival time. Flow level feature extraction occurs on the entire flow, resulting in features as the total number of packets, total bytes, and the flow duration. The solutions proposed in the literature can be divided into supervised machine learning classifiers, unsupervised machine learning classifiers, and semi-supervised machine learning classifiers. In general, the machine learning approach follows the framework depicted in Fig. 1. The general process of the machine learning framework is summarized as follows:

- **Data collection**: This step is the initial step of any network traffic classification technique that must contain a sufficient number of targeted applications/protocols traffic. Two approaches are usually applied: private data collection or public dataset utilization. The former requires the implementation of measuring procedures to capture the network traffic of the targeted applications generated from computers, IoT, or mobile devices. The latter deploys public traffic traces that have been collected by other scientists and provided to other researchers free of charge. Section 7 sheds light on the recent public datasets that are available in the field of network classification. Usually, the raw traces are collected in the form of PCAP files for further processing.
- **Flow representation**: The raw traffic collected must be represented in a form that allows the extraction of statistical features for each connection generated by the application. The most common approach is the deployment of flow representation. It aggregates packets that share the 5-tuple, i.e., the source port, destination port, source IP, destination IP, and the used protocol (i.e., TCP, UDP). The packets collection for each flow could be unidirectional or bidirectional. Unidirectional collection aggregates packets that share the 5-tuple for a single direction. Bidirectional collection aggregates packets that share the 5-tuple for both directions.
- **Feature engineering**: This process is a crucial step as it affects overall performance of the classifier. It computes different metrics extracted from each flow, which reflect the properties of the collected traffic. Feature engineering usually involves two steps: feature extraction and feature reduction. Feature extraction is the art of identifying statistical features that characterize each application's flows and help to distinguish different applications' flows. This process could be conducted manually after carefully analyzing the collected flows, or using publicly available tools that extract the statistical features in an automated fashion. The result of the feature extraction process is a 2D matrix, in which each row represents a flow of an application and each column represents a statistical feature. Feature reduction/selection is an optional yet preferable process that helps reduce the irrelevant and redundant extracted features, potentially improving the accuracy and computational resource requirements of the classifier. The two common categories of feature selection algorithms are wrapper and filter. Wrapper identifies the optimal feature subset by assessing the classifier using different feature subsets iteratively. Filter selects the irredundant and relevant features by analyzing the training dataset without any evaluation of the classifier. Correlation–based Feature Selection (CFS) [29] applies correlation and inter-correlation processes to select a subset of features from the entire extracted features. The correlation process identifies the relevant features by measuring the relevance of features to a class. The cross-correlation process identifies the redundant features by measuring the inter-correlation amongst features in a subset. The resulting features are highly correlated with the defined classes and and have low correlation with each other. Consistency-based subset search (CON) selection algorithm [30] identifies features that have a robust association with the same class. Gain Ratio (GR) feature selection algorithm ranks features based on their relevance to the defined classes in the dataset. GR depends on Information Gain (IG) algorithm, which helps improve its performance since IG tends to have features with a large number of values [31]. IG ranks the features in the dataset based on their entropy, where it scores the difference of the entropy of the classes before and after observing features in the dataset. Chi-square [32] feature selection algorithm ranks features based on their relevance to the training dataset by calculating the chi-square statistics of a feature with respect to the class.
- **Datasets preparation**: After extracting the features and applying the feature selection algorithm, a dataset containing the historical data of the targeted applications is ready to be used for building and testing the classifier using separate training and testing datasets or applying N-fold cross validation. Separate training and testing datasets require the deployment of two datasets to build and evaluate the classifier. The training phase requires a large training set for optimal training process. Similarly, acceptable testing dataset is required to properly evaluate the classifier being built. Deploying the same dataset for training and testing is considered a poor practice since the results could be misleading. N-fold cross validation can be applied. In a nutshell, it divides the dataset into approximately N equal folds. N-1/N folds are used for training, and the remaining 1/N is used for testing. This process is iterated N times with the predefined folds. The literature has showed that $N = 10$ parameter provides an acceptable classification performance [33].
- **Model building**: The training dataset generated in the previous step is used to build a model that classifies network flows into their causal applications/protocols. Various machine learning algorithms have been developed to resolve tasks like classification and clustering. The selection of the machine learning algorithm is related to the type of knowledge and problem that the analyst aims to discover and solve. In general, two groups of algorithms are applied, namely the supervised and unsupervised algorithms. A hybrid approach, called semi-supervised solutions, has recently been gaining popularity. Later subsections discuss them in more details.
- **Model evaluation**: After the establishment of the model and before the deployment of the solution into the production environment, an evaluation analysis is required to assess the built classifier performance. In a nutshell, the performance metrics are quantified for the built model (i.e., supervised learning models) by analyzing the model prediction instances to their correspondence ground truth labels. The performance metric is computed according to the classification goal of its binary or multiclass classification. Binary classification happens when an input instance is classified into one of two distinct classes. On the other hand, multi-class classification indicates that the input

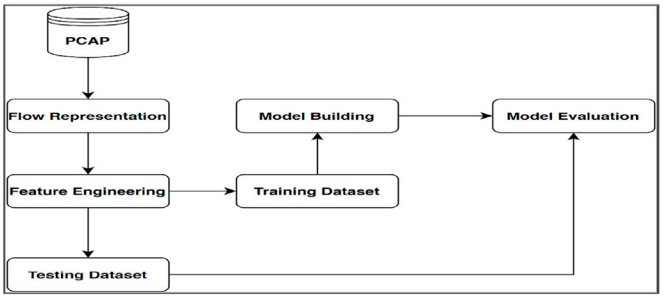

Fig. 1. Supervised learning flowchart.

instance is classified into one class within many classes. Five common metrics are used to evaluate a built classifier for a supervised classification problem [34], namely the accuracy, precision, recall, F-measure, and Receiver Operator Characteristic (RoC) curve. Accuracy indicates the overall accuracy of the model by dividing the correctly classified flows over the total number of flows in a dataset. Literature has shown that accuracy metric alone might provide misleading interpretation for imbalanced datasets [35]. Precision, recall, and F-measure metrics provide more accurate results for this case. Precision indicates the classifier's performance in terms of the degree of accuracy by which the flows are classified into a class. Recall indicated the classifier's performance in terms of the number of flows that are classified into a targeted class from the whole dataset. F-measure is the harmonic mean of both precision and recall. The equations of the four metrics are provided as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$F - Measure = 2*\frac{Precision*Recall}{Precision + Recall} \tag{4}$$

Where TP is true positive, TN is true negative, FN is false negative, and FP is false positive. RoC curve is the plot of the FP rate (x-axis) versus the recall (TP rate) (y-axis) for each possible classification threshold. The RoC curve pictures all the possible classification thresholds, whereas accuracy represents only the performance for a single threshold.

5.1. Supervised machine learning solution

Supervised machine learning algorithms generate knowledge structures that help classify untrained instances into pre-defined classes [36]. In supervised learning, there are two main phases: training and testing. The training phase is the process of building a classification model. This is conducted by providing the supervised algorithm with labeled instances along with their statistical features, allowing it to extract patterns and create the knowledge that distinguish classes from each other. The learnt knowledge can be represented as rules, decision tree, or flowchart according to the applied algorithm. The testing phase is the process of classifying unseen/untrained instances using the classifier built in the previous phase. Supervised learning algorithms are deployed to build the classifier to classify network traffic.

- Support Vector Machines (SVMs) [37] are a group of related supervised learning methods for classification and regression purposes. SVM classifies data by using a single hyperplane or multiple hyperplanes in a high-dimensional space. SVM aims to reduce the probability of generalization error when building the classifier. This is conducted when the primal hyperplane is selected, which ensures the largest distance between the closest instances of the classes in the training dataset.
- Naïve Bayes (NB) [38] learning algorithm is based on Bayes' theorem and probability notions. The algorithm considers the presence or absence of features as independent of each other, which enables it to provide good performance on a small number of instances datasets.
- C4.5 [39] is a decision tree learning algorithm that provides a top-down structure trees with an iterative division of the training dataset. A Node in the trees donates a feature, a branch donates a possible value, and a leaf represents a class label.
- C5.0 [40] algorithm is based on a decision tree and considered an improved version of C4.5, where it provides more accurate rules with less time to be generated. C5.0 generates several decision trees and combines them for a better prediction. In addition, it provides misclassification cost that helps avoid error. Furthermore, it supports sampling and cross-validation techniques.
- Random Forest (RF) [41] applies an ensemble approach in building the classification model. Contrary to a single decision tree, RF builds multiple classifiers for the classification problem, which helps to provide a strong classifier from several weak individual classifiers.
- K Nearest Neighbor (kNN) [42] is considered a lazy and non-parametric learning algorithm. Contrary to the previous algorithms, it does not require the training phase. Its classification time is proportional to the collected data size. The algorithm measures the distance between the tested instances with the labeled instances. As a result, those instances are assigned to K-nearest class. Usually, Euclidean distance is used to measure the distance between two instances features vectors.

Researchers in the literature have addressed extensively the deployment of supervised machine learning algorithms to classify network traffic. Their focus and aim could be summarized in the three following categories: full flow monitoring, sub flow monitoring, and detecting untrained versions. The next subsections discuss in detail the relevant work in the literature of each category.

Full flow monitoring. In this category, the literature focused on building a supervised machine learning solution to classify network flows with high accuracy results and low computational and time requirements in establishing and testing the classifier. The classifier collects the statistical features of the full flow, starting from the sync packet to the finish packet or when the connection tears down. A comparison between various feature selection and learning algorithms has been conducted in terms of the mentioned criteria.

Williams et al. [43] compared the performance of Bayes Net, C4.5, AdaBosot C4.5, NBTree, AdaBoost NBD, and Nearest Neighbor learning algorithms for network classification, along with CSF, wrapper, and consistency feature selection algorithms. National Laboratory for Applied Network Research (NLANR) dataset has been used for building and evaluating the models, containing network traffic of various protocols as HTTP, SMTP, and others. The authors extracted features such as packet length and inter-arrival statistical values. Wrapper feature selection algorithm and AdaBosot C4.5 learning algorithm provided the best accuracy results. Moreover, the work evaluated the speed of the building and testing phases. NBK algorithm was the fastest in the building phase, and C4.5 was the fastest in the testing phase. The assessment of different SVM kernel functions to classify network flows, including linear, polynomial, sigmoid, and radial kernels, have been carried out in the literature [44, 45]. Sequential forward feature selection algorithm has been applied to reduce the irrelevant and redundant features, resulting in better accuracy. Radial Kernel showed the best performance amongst the rest of the kernels. Jenefa and Moses [46] compared the accuracy results of C5.0 against C4.5, SVM and NB learning algorithms. The extracted features included packet rate, data rate, and inter-arrival time statistics (minimum, mean, maximum, and standard deviation). The experimental evaluation has been conducted using a private dataset collected at their labs, which included 17 applications. The results proved the outperformance of C5.0, achieving 99% recall and precision for most of the classes. Dias et al. [47] proposed a machine learning framework for the classification of real-time applications, specifically video network traffic. The framework was trained using 13 statistical features, including packet arrival time, the average of the decimal value, and the average value of the IP datagram length. The dataset has been collected in their labs, including YouTube videos, Netflix videos, and files downloaded. The average accuracy for this proposed algorithm achieved 98.88%, showing promising results for the classification of real-time applications.

Compared to NB, the proposed solution took less time for the establishment and classification.

Alshammari and Zincir-Heywood [48] discussed the effectiveness of three machine learning algorithms, i.e., AdaBoost, C4.5, and Genetic Programming (GP), in classifying network flows. Their experiment included statistical features, such as the duration of a flow and the statistics of packet length and arrival intervals in both directions. The data they privately collected included VoIP traffic such as Skype, Gtalk, and Yahoo messenger as well as non-VoIP traffic. C4.5 scored highest with a 99% Detection Rate (DR) to classify Skype and Gtalk traffic with less than 1% and 0.2% False Positive Rate (FPR) respectively. The same authors address the robustness of the built classifier [49]. Their experiment compared AdaBoost, SVM, NB, RIPPER, and C4.5 learning algorithms to classify Skype and SSH traffic. Four datasets were used for evaluation purposes, including a privately collected dataset, NLANR dataset, Measurement and Analysis of the WIDE Internet (MAWI) dataset, and DARPA dataset. C4.5 accuracy results surpassed the rest of the algorithms for both Skype and SSH detection. Sun et al. [50] addressed the reduction of the computational resources of the traditional SVM learning classifier by introducing the concept of Incremental SVM (ISVM), which reduces the high training cost of memory and CPU. Furthermore, the authors have proposed a framework called Authenticator ISVM (AISVM) that uses the valuable information found in the previous training datasets. The tested learning algorithms, NB and NBKDE, showed lower accuracy than SVM. The experiment also presented that the proposed frameworks (ISVM and AISVM) provided higher accuracy results with less computational resources than SVM. Cao et al. [51] focused on improving the traffic classification accuracy of the SVM-based model using two modules, feature engineer and classifier building. Feature engineer module deploys a novel filter-wrapper mixed feature selection algorithm to extract the best feature set that best represents the original feature set. The classifier building module deploys an Improved Grid Search parameter optimization algorithm that identifies the best key parameter combination of SVM algorithm to improve model training and enhance accuracy results. The proposed solution has been evaluated using Moore dataset and achieved high classification accuracy results with small feature space when compared to SVM, NB, and kNN algorithms. Khatouni and Zincir-Heywood [52] conducted a comparative analysis between different off-the-shelf network traffic flow exporters to classify the network traffic named Argus [53], Silk [54], Tstat [55], and Tranalyzer [56]. NIMS dataset collected at different locations have been used for evaluation purposes. They were also utilized to address the robustness of the built model that includes various services traffic like browsing and Audio network flows. The evaluation results showed the robustness of the built model is the highest when tested at different locations, and the average TP score is 85%.

Dong [57] proposed an enhanced SVM solution, named cost-sensitive SVM (CMSVM), which aims to improve accuracy, increase computational cost and solve data imbalance problems. The solution utilizes an active learning technique to help dynamically assign the weights for the targeted applications. The proposed solution has been evaluated using Moore and NOC_SET datasets to classify network flows into their correspondent service groups. The obtained results showed that the proposed solution is more effective than the traditional SVM classifier in accuracy and imbalance. Afuwape et al. [58] addressed the effectiveness of classifying VPN and non-VPN network traffic using ensemble classifiers, in terms of precision, recall and F1-score. The authors evaluated various ensemble and single classifiers in their experiments using ISCX dataset. The results showed better accuracy performance of Gradient Boosting (GB) and Random Forest ensemble classifiers when compared with the single classifiers like decision tree, Multi-Layer Perceptron (MLP) and kNN. Ganesan et al. [59] introduced a machine learning scheduling framework that aims to prioritize network traffic in IoT environment based on its QoS requirements. The work compared the performance of seven supervised learning algorithms, including RF, kNN, MLP, NB, logistic Regression (LR) and SVM. UNSW dataset with 21 IoT/non-IoT devices have been used for the evaluation purposes, where RF achieved the highest accuracy results.

Sub-flow monitoring. In this category, the literature has addressed the effectiveness of the supervised machine learning algorithms to classify network traffic at an early stage by observing the first few packets or bytes of a flow instead of monitoring the whole flow's packets as previous solutions. This helps to reduce the computational time and resources requirement as well as the detection time. Besides, it leads to the classification of the flows in real time. Different sub-flow sizes, feature selection and learning algorithms have been evaluated in the literature in terms of supporting real time classification with high accuracy results.

Li et al. [60] tackled the classification of P2P network traffic by observing the first few packets and compared their obtained results with those of Bernaille et al. [61]. For building the classifiers, C4.5 and REPTree learning algorithms were used with features as flow duration and total packets and bytes in a flow. A dataset in their lab was obtained to evaluate the approach, containing P2P traffic as BitTorrent, Kazaa, and Skype. C4.5 showed the highest accuracy and classification time results. Gu et al. [62] addressed C4.5, SVM, NB, and RF learning algorithms to classify network flows at an early stage. The duration of the flow, the number of packets and bytes for each direction, and the statistics of packet length and inter-arrival time features were utilized in the building phase. Privately collected datasets in their lab were used, including HTTP, FTP, and Skype. The authors observed an enhancement of the performance when applying CFS feature selection, with C4.5 being superior amongst the rest of the algorithms.

Liu et al. [63] proposed the Window First N Packets (WFNP) algorithm that classifies network flows on the fly by monitoring the first N packets. The solution contains two phases: the extraction and classification. The first phase extracts the first N packets of each flow to calculate the statistical features. In the second phase, WFNP deploys a C4.5 decision tree learning algorithm to classify the traffic. A total of 11 features were extracted as the first packet length, initialization window, and push packet number. The proposed algorithm, along with SVM, kNN, and NB, were evaluated against a private dataset collected in their labs. The evaluation results showed the outperformance of the proposed framework, achieving 96% accuracy by observing the first eight packets. Peng et al. [64] addressed the efficiency of applying payload size statistical features, including mean, standard deviation, minimum, maximum, and variance, in classifying network flows early. The experimental evaluation has been conducted over three public datasets, namely Auckland II, UNIBS, and UJN traces, where each one contains different network traffic generated from various protocols. Ten learning algorithms, including J48, NBTree, RF, and NB, were built and tested to classify the network traffic by monitoring the first six packets. The evaluation results showed the effectiveness of the used features and learning algorithms, where most of the built classifiers achieved higher than 80% accuracy results.

Detecting untrained versions. The previous solutions did not address the efficiency of the built classifier in classifying untrained versions of different applications. Not all statistical features share similar values for different versions of an application. Therefore, the built classifier's performance will deteriorate when detecting the untrained versions. For example, Skype version 3 uses the SVOPC codec, whereas in version 4 and later, it deploys the SILK codec [23]. As a result, the statistical features will have different values for the different versions. The literature shed light on this dilemma and proposed frameworks to classify the untrained version by building the classifier on a different version.

Branch and But [65] proposed generic features for classifying untrained versions of VoIP traffic. Skype versions 2, 3, 4, and Gtalk traffic were privately collected to evaluate the proposed features. For experimental evaluation, the training dataset used a single version of Skype traffic with other network traffic and the testing dataset contained the untrained versions traffic of Skype. C4.5 successfully identified the untrained Skype versions while using a single version in the building process. Azab et al. [66] proposed a solution that utilized cost-sensitive algorithms to detect untrained versions. The authors evaluated the

proposed framework using C4.5 and RF algorithms and noticed that the traditional cost-insensitive algorithms operate poorly in detecting the untrained version of Skype flows. The proposed framework has improved the detection results of the untrained version by 40% when using RF. Later, the same authors [67] improved the framework by integrating multi-classifiers, named lenient and strict classifiers, in the building process in conjunction with a feature combination phase. The lenient classifier maximizes the detection of the trained version, thus the detection of the untrained version. The strict classifier reduces the FP results of the lenient classifier. The feature combination maximizes the possibilities of detecting the trained version, thus increasing the likelihood of detecting the untrained version. The evaluation experiment, using C4.5, NB, RF, and SVM learning algorithms, has shown the effectiveness of the proposed framework, achieving high recall results in detecting the untrained version of Skype network traffic. The cost-sensitive algorithms have been evaluated in detecting botnet Command and Control (C&C) traffic and differentiating it from the legitimate HTTP traffic [68,69]. The authors used Zeus malware as their case study and privately collected its C&C traffic for versions 1 and 2, as well as the legitimate HTTP traffic from various websites. The building process utilized C4.5 learning algorithm and CFS as feature selection algorithms. The evaluation results showed that the proposed framework provided better accuracy results in detecting the untrained version of Zeus C&C traffic than the standard cost-insensitive machine learning approach.

Table 3 summarizes the discussed supervised learning solutions in terms of the granularity, number of classes, used feature selection and learning algorithms, the ability to detect early stages flows, the support of detecting untrained versions, and the goal of the conducted evaluation. It could be inferred that the focus ranged from accuracy and computational requirements enhancement to robustness of the built classifier, detection of flows at an early stage, and detection of untrained versions. Furthermore, the table shows that the addressed granularity tested to classify flows to their causal protocol, service group, or application.

5.2. Unsupervised machine learning solution

Unlike supervised learning, unsupervised learning does not require labeled datasets in the building process and aims to identify patterns of the input data, where instances with the similar properties are grouped with each other. Clustering, which is a technique in unsupervised

Table 3
Summary of supervised learning solutions.

Study	Granularity	Binary/Multi class	Algorithms	Feature Algorithm	Dataset	Early-Stage Detection	Untrained Version Detection	Focus
[43]	Protocol	Multiclass (6 classes)	Bayes Net, C4.5, AdaBosot C4.5, NBTree, AdaBoost NBD, Nearest Neighbor	CSF,wrapper, consistency	NILMA	No	No	Accuracy, computational time
[44, 45]	Service Group	Multiclass (8, 7 classes)	SVM	Sequential forward feature	Moore, Private	No	No	Accuracy
[46]	Application	Multiclass (17 classes)	C5.0, C4.5, SVM, NB	No	Private	No	No	Accuracy
[47]	Application	Multiclass (3 classes)	NB, Novel framework based on NB	No	Private	No	No	Accuracy, computational time
[48]	Application	Binary (Skype/ Non-Skype) (Gtalk/ Non-Gtalk)	AdaBoost, C4.5, GP	No	Private	No	No	Accuracy
[49]	Application	Binary (Skype/ Non-Skype) (SSH/ Non-SSH)	AdaBoost, SVM, NB, RIPPER, C4.5	No	Private, (NLANR)	No	No	Accuracy
[50]	Service Group	Multiclass (10)	NB, NBKDE, SVM, ISVM, AISVM	No	Moore	No	No	Accuracy, computational time
[51]	Service Group	Multiclass (10)	SVM, improved SVM, NB, kNN	Filter Wrapper	Moore	No	No	Accuracy
[52]	Service Group	Multiclass (11)	Decision Tree	No	NIMS	No	No	Accuracy, robustness, feature extraction tools comparison
[57]	Service Group	Multiclass (10)	SVM,CMSVM	No	Moore, NOC_SET	No	No	Accuracy, computational cost, imbalanced dataset
[58]	Service Group	Multiclass (7)	GB, RF, kNN, Decision Tree	No	ISCX	No	No	Accuracy
[59]	Application	Multiclass (21)	RF, kNN, MLP, NB, LR	No	UNSW	No	No	Accuracy
[60]	Application	Binary (Skype/ Non-Skype), Multiclass (12 classes)	C4.5, REPTree	No	Private	5 packets	No	Accuracy, computational time, detecting unknown P2P traffic
[62]	Application	Binary (Skype/ Non-Skype)	C4.5, SVM, NB, RF	CFS	Private	12 packets	No	Accuracy
[63]	Protocol	Multiclass (6 classes)	WFNP, SVM, kNN, NB	CFS, IG, CON	Private	8 packets	No	Accuracy
[64]	Protocol/ Application	Multiclass (various classes for each tested dataset)	NB, BayesNet, AdaBoost, Bagging, OneR, PART, kNN, J48, NBTree, RF	No	Auckland II, UNIBS, UJN	6 packets	No	Accuracy
[65]	Application	Multiclass (8 classes)	C4.5	No	Private	100 packets	Yes	Accuracy
[66]	Application	Binary (Skype/ Non-Skype)	C4.5, RF	CFS	MAWI, Private	6 s	Yes	Accuracy
[67]	Application	Binary (Skype/ Non-Skype)	C4.5, RF, NB, SVM	CFS	MAWI, Private	6 s	Yes	Accuracy
[68, 69]	Application	Binary (Zeus/Non-Zeus)	C4.5	CFS	Private	No	Yes	Accuracy

learning algorithms, is widely used in the network traffic identification field. It aims to find hidden or unknown patterns in unlabeled input data in the form of clusters. To fulfill this, it groups the data based on the similarity between different features defined in the feature extraction step. The clustering process aims to organize data with inter-cluster and high intra-cluster similarity. An instance may belong to a single cluster, called a group, which is measured with a certain probability. Unlike supervised algorithms, clustering evaluation contains intermediate steps [70]. Useful criteria for cluster evaluation include the number of generated hidden clusters, comparing two algorithms, labelling cost, resources cost and the building speed process. According to Halkidi et al. [71], cluster validity examination is classified into external criteria, internal criteria, and relative criteria.

In general, clustering methods are divided into the following techniques: Hierarchical clustering, Bayesian clustering, and Partitional clustering. Hierarchical clustering applies a bottom-up or top-down approach. It does not require the pre-identification of the number of clusters. However, this could affect the efficiency of the solution. In terms of complexity, hierarchical clustering has quadratic computational complexity. Bayesian clustering deploys the concept of probabilistic clustering where it assigns an instance to a class with a probabilistic value. The infinite dimension of the parameter space is called non-parametric Bayes, otherwise, it is called parametric Bayesian. The major challenge of Bayes clustering is the selection of the probability distribution, which might degrade the projection of the data. Partitional clustering defines data as a set of separate clusters. It aims to divide the data into $K<n$ clusters, where n is the specified observation. K-means is the most popular algorithm in partitional clustering and mostly deployed in network traffic identification as illustrated later in this section.

K-means is a popular and simple learning algorithm designed to group similar data points and discover patterns. To fulfill this, it divides the provided dataset into K unique non-overlapping clusters, where each data point belongs to only one group, as illustrated in Fig. 2. A cluster is a collection of data points accumulated together based on certain similarities. The grouping process is conducted by minimizing of the sum of squared distance (as the Euclidean distance) between data points and the defined centroid.

The literature addressed the effectiveness of unsupervised learning algorithms in the field of network traffic identification without the need to have a labeled dataset. Although it is not as rich as supervised learning algorithms, it shows its applicability in constructing high-performance.

Wang et al. [72] addressed the effectiveness of deploying K-means clustering algorithm to group similar network flows. The proposed framework expresses on side of information in the form of pair-wise constraints and applies a few constrained K-means variants. For flow representation, 31 features were used, including duration, inter-arrival time and packet size statistics, and the transferred bytes. The evaluation has been conducted on publicly available datasets like MAWI.

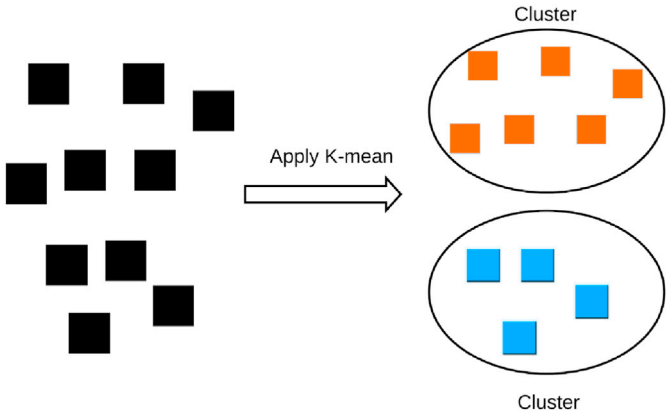

Fig. 2. K-means clustering overview.

Sampling technique was applied, resulting in 10 major classes, such as HTTP, POP3, and SMTP. The evaluation results show that the constrained algorithms outperform the unconstrained algorithms. Moreover, the larger number of clusters provided better superclass precision results. The identification or time application using K-means was addressed by Dubin et al. [73]. The framework deploys K-means algorithm as part of the proposed framework which utilizes DPI as a preprocessing stage to determine if the flow is a real-time application. The statistical feature used by the proposed framework was the bit rate throughput in the time period based on the user's TCP stack implementation. The evaluation was conducted over YouTube videos traffic with different bitrates, such as 360 and 720 bps, which has been collected in their labs. The framework provided 97% overall accuracy when using 14 clusters. Du and Zhang [74] deployed K-means algorithm for P2P traffic identification. The authors established a lightweight clustering solution to detect three applications: BitTorrent, BitSpirit, and eMule. The features used included packet lengths, source and destination ports, and the number of packets in a flow. For evaluation purposes, they have tested the proposed solution using a privately collected dataset in their labs. The results showed high TPR results for the three targeted applications. Singh [75] compared the accuracy of K-means and Expectation Maximization (EM) clustering algorithms in identifying HTTP, DNS, SMTP, and ICMP network traffic. The extracted features included inter-arrival time and packet size statistics, flow duration, and the number of transferred bytes. The evaluation results revealed the superiority of K-means over EM in achieving a higher accuracy for the targeted applications, thus achieving the best results for both when using 80 clusters.

Zhang et al. [76] proposed an unsupervised approach that is capable of discovering application-based traffic classes and classifying network flows into their generated applications. To fulfill their goal, the researchers introduced the Bag-of-Word (BoW) model to represent traffic clusters, and the aggregation process was conducted via Latent Semantic Analysis (LSA). The authors' evaluation process was carried out over a privately collected data in their labs, containing traffic from 13 applications such as BitTorrent, MSN, and Yahoo. The extracted statistical features included packet size and inter-arrival time statistics. The proposed solution provided the best accuracy results amongst the tested algorithms that included C4.5, kNN, SVM, NB, Neural Network, and Bayes Nets. Alalousi et al. [77] conducted an evaluation comparison among K-means, kNN, and EM learning algorithms in terms of classifying accuracy, speed, and memory consumption. The testing environment took place over Moore dataset, using 249 statistical features. KNN achieved the highest accuracy results with the lowest memory consumption. On the other hand, K-means showed the lowest required classification speed, which made it suitable for online classification. Höchst et al. [78] achieved a traffic flow classification solution using statistical features based on a neural autoencoder algorithm. The framework automatically clusters traffic flows into downloads, uploads, or voice calls, independent of the application's protocol. A newly time interval-based feature for vector construction has been proposed, using time window of 2048 s. A privately collected dataset has been utilized for evaluation purposes, containing seven classes. The autoencoder with 100 clusters provided the highest accuracy results. Furthermore, the identification time results were low, allowing their utilization at the network edges. Alizadeh et al. [79] evaluated the effectiveness of deploying Gaussian Mixture Models (GMMs) unsupervised algorithm to classify network flows. The distribution of the extracted feature vectors from network flows are modelled by a Gaussian mixture density. UNIBS-2009 dataset, containing five classes, was used for evaluation comparison. GMM scored higher accuracy results than decision tree, NB, and kNN.

Table 4 summarizes the unsupervised learning solutions discussed in terms of granularity, number of clusters, applied learning algorithms, and deployed datasets. The popularity of K-means algorithm in the field of network traffic identification can be seen. Also, the solutions showed the efficacy of unsupervised learning algorithms to map network traffic to applications, service groups, or protocols.

5.3. Semi-supervised solution

Hybrid solutions, also known as semi-supervised solutions, combine both supervised and unsupervised algorithms to achieve the network traffic classification goal. Usually, unsupervised algorithms are first used for feature selection process or the labeling of unknown flows. Afterward, supervised learning algorithm is utilized for the classification of the network traffic flows to their causal application. This approach overcomes the shortcomings of standalone supervised and unsupervised solutions and usually offers more granularity of traffic classification.

Prior works have used semi-supervised learning algorithms to overcome the shortages of both supervised and unsupervised learning algorithms (more discussion in Section 8). The literature used unsupervised learning algorithms first to either label unlabeled datasets or select the most relevant features. Later, supervised learning algorithms were used to classify network flows to their causal application, protocol, or service group.

Wiradinata and Paramita [80] proposed a framework to improve the building and classification time of the standalone kNN algorithm. The solution first integrated the Principal Component Analysis (PCA) feature selection reduction to identify the most discriminant features for the classification problem. Then, the Fuzzy C-Mean clustering algorithm was deployed to create clusters of the reduced feature dataset. The utilization of Fuzzy C-Mean helps kNN algorithm not to perform the calculation of all distances between the existing data. For evaluation purposes, the authors utilized Moore dataset with the defined statistical features. The obtained results showed comparable accuracy results between the traditional kNN and the proposed framework. On the other hand, the proposed framework decreased the execution time by almost 400 s. Zhang et al. [81] addressed the problem of unknown applications of a small supervised training dataset. The authors deployed K-means to group network flows and label unknown traffic flows, where the output of K-means could be used as the input of the classifier. Compound classification was used to classify the correlated flows instead of individual flows. Nearest Cluster-based Classifier (NCC) has been used for this purpose. MAWI dataset has been deployed for testing the proposed framework with the extracted statistical features such as packet size and inter-arrival time. The evaluation results have shown the superiority of the proposed framework in classifying unknown classes over the C4.5, NB, and kNN algorithms. The solution of Glennan et al. [82] aimed to classify known and unknown network traffic flows. The suggested hybrid framework contained a feature selection algorithm to reduce the feature set, a clustering algorithm to conduct labelling for unknown classes, and a classification algorithm. The evaluation has been conducted over the MAWI dataset, containing flows from five classes. The feature selection algorithm reduced that feature set to 17 features, including inter-arrival time and bytes size statistics. K-Mean with 500 clusters successfully labeled the unknown classes' traffic. The accuracy results were high, especially for unknown class traffic with more than 97%. Moreover, the proposed framework has shown that it requires low time complexity for the labelling process. Bakhshi and Ghita [83]proposed a two-phase machine learning solution to traffic classification over a per-flow traffic. The first phase utilized K-means unsupervised learning algorithm to group applications' flows into general granular classes. For example, Skype traffic was grouped in comms class and YouTube and Netflix traffic were grouped in streaming class. In total, cluster analysis resulted in 12 unique flow classes. The second phase integrated C5.0 supervised learning algorithm to classify network flows to the identified classes in the first phase. The experiment included the collection of a private dataset in their labs. An average prediction accuracy of the best feature set composed of 14 attributes was 92.37%, and the average prediction accuracy increased to 96.67% with adaptive boosting. Fahad et al. [84] suggested a framework called SemTra which aims to label unlabeled instances for better classification results. The framework first generated multi-view representations of the data. Then, these representations were fed into the ensemble K-means clustering models in order to provide a combined clustering output. Finally, the framework identified the class decision by joining the decisions of previous steps. SemTra significantly outperformed the other algorithms, such as Probabilistic Graphical Model (PGM) [85], Offline/Real-Time Semi-Supervised Classification (ORTSC) [86], Bipartite Graph-Based Maximization (BGBM) [87], and Various-Widths Clustering (VWC) [88] when tested over Moore dataset. Zhao et al. [89] aimed to detect unknown applications and label unlabeled flows based on unsupervised and the tri-training method. In general, the solution contained the three following stages: clustering stage, building classifier stage, and prediction stage. K-means was used to extend the labelling of the unlabeled instances. Tri-training built three classifiers using the output of the cluster to provide a robust predictive solution. The used statistical features included inter-arrival time and packet size statistics. The evaluation results over MAWI dataset showed the better accuracy performance of the proposed solution over the traditional supervised algorithms.

Zhang et al. [90] designed a Robust Statistical Traffic Classification (RSTC) that identifies the traffic of zero-day applications and accurately discriminates predefined application classes. The solution consisted of three modules: unknown discovery, bag of flows traffic classification, and system update. Unknown discovery identified new samples of zero-day traffic in a set of unlabeled traffic. The module of bag of flows built a classifier for robust traffic classification using the clustered input. The system update module constructed new classes to complement the system's knowledge. RF and K-means learning algorithms have been employed to perform supervised classification and unsupervised learning. RTC has been tested over FTP, HTTP, and POP3 network traffic from MAWI and outperformed the classical RF and single class SVM accuracy. Noorbehbahani and Mansoori [91] established a semi-supervised framework based on X-means clustering and label propagation to classify network traffic. The solution consisted of two phases. The first phase contained the processes of data preprocessing, the conversion of nominal values to numerical values, and the selection of most related features. In the second phase, clustering, label propagating, and classification processes were conducted. CFS has been used for feature selection, X-means for clustering, Euclidean distance for labelling unlabeled instances based on the cluster neighbor, and J48 and NB for classification. The evaluation results using Moore dataset showed that the proposed framework revealed comparable accuracy results to an already labeled dataset. Van Ede et al. [92] proposed the FlowPrint framework which deploys a semi-supervised approach to fingerprint both visible and invisible mobile applications' traffic by clustering the destination networks and measuring

Table 4
Summary of unsupervised learning solutions.

Study	Granularity	Number of clusters	Algorithm	Dataset	Accuracy(%)	DataSet Size
[72]	Application/protocol	0–1000	K-means	MAWI	88	>10000 items
[73]	Application	14	K-means	Private	97	N/A
[74]	Application	N/A	K-means	Private	95	N/A
[75]	Protocol	80	K-means, EM	N/A	90	N/A
[76]	Application	400	K-means	Private	90	N/A
[77]	Service Group	N/A	kNN, K-means, and EM	Moore	95	> 370000 items
[78]	Service Group	100	Autoencoder	Private	90	N/A
[79]	Protocol/Application	N/A	GMM	UNIBS-2009	99	> 79000 flows

the temporal correlation using the cross-correlation approach. The solution included device, destination, and timing features ranked with Adjusted Mutual Information (AMI) algorithm which allow the traffic to be classified within 300 s. Using a privately collected dataset, the solution achieved 89.2% accuracy in classifying apps and web traffic and 93.5% accuracy in detecting unseen traffic. Andreoni Lopez et al. [93] presented a solution that preprocesses the feature selection step using an unsupervised approach before employing a supervised learning algorithm for classification. First, a proposed normalization algorithm was utilized to enforce the data values between −1 and 1 in order to provide less classification error. Second, a Pearson's correlation coefficient identified the correlation of all the pairs of the defined features to select the best feature set. For evaluation purposes, SVM, RF, decision tree, kNN, and NB have been utilized on three different datasets, achieving 11% improvement in accuracy when compared to the traditional feature selection algorithms.

Table 5 compares the discussed semi-supervised solutions in terms of granularity, deployed algorithms, and datasets and their aims. It shows that the focus of the discussed literature ranged from detecting zero-day applications to labelling unlabeled datasets and classifying unknown classes to enhance the classifier's accuracy and computational requirement.

6. Deep learning solution

Neural Networks (NN) are defined as interconnected processing elements that process information according to the state response to external inputs. NN are built using neurons that are connected via links, and each link has a weight value. One way to adjust the weights of the links is backpropagation. Deep learning could be classified as a variant of NN with many hidden layers. Most common deep learning methods, especially in network traffic classification, are MLP, Recurrent Neural Network (RNN), Auto Encoders (AE), and CNN.

Table 5
Summary of semi-supervised learning solutions.

Study	Granularity	Algorithms	Dataset	Aim
[80]	Service Group	PCA, Fuzzy C-means, kNN	Moore	Reducing classification time
[81]	Protocol	K-means, NCC	MAWI	Classifying unknown classes
[82]	Protocol	Extra trees classifier algorithm, K-means, NCC	MAWI	Classifying unknown classes
[83]	Service Group	K-means, C5.0	Private	Deriving individual flow classes per application through K-means. C5.0 decision tree classifier for classification purposes
[84]	Service group	K-means, SVM	Moore	Labelling unlabeled instances
[89]	Protocol	K-means, C4.5, RF, NB	MAWI	Detecting unknown applications and extending labeled flows from a few labeled and many unlabeled flows
[90]	Protocol	RF, K-means	MAWI	Detecting zero-day applications
[91]	Service Group	CFS, X-means, NB, J48	Moore	Comparing semi-supervised labelling process against already labeled dataset
[92]	Application	Cross correlation, Adjusted Mutual Information (AMI)	Private	Detecting unknown apps, real time classification
[93]	Service Group	Pearson Coefficient, SVM, RF, decision tree, kNN. and NB	NSL-KDD, GTA/ UFRJ, NetOp	Feature selection, accuracy, classification time

- MLP is considered as a feedforward of NN architectures and mainly consists of input, output, and hidden layers, where each layer contains several neurons that are connected to the adjacent layers. Each neuron uses a non-linear activation function to calculate its output by using the sum of its input. MLP is considered to be hard to train and very complex due to the numerous parameters used by the model. This encourages researchers to use MLP as part of a framework rather than as a standalone solution.
- RNN is another form of the deep neural network that aims to identify the temporal correlation of an input. RNN has proven its efficiency in the field of speech recognition and time series anomaly detection. Contrary to a feedforward neural network, the outputs of certain layers in RNN are used as a feedback into the inputs of a previous layer. RNN contains a recurrent connection on the hidden state, ensuring the capture of the sequential information in the input data. RNN involves the utilization of parameter sharing, which shares the parameters across different time steps. Long Short-Term Memory (LSTM) is a specific implementation of RNN that solves the vanishing gradient problem in RNN by providing a long-term learning relationship.
- CNN is one form of the deep neural networks that aims to identify the spatial correlation of an input and has proven its efficiency in image recognition field. CNN simplifies the image to a simpler form for processing while avoiding the feature loss that might affect a good prediction. In order to fulfill this, the input image is denoted as a vector of pixel values and a filter vector is specified. The filter vector goes over the image by sliding from the top left to the right with a specific stride, allowing the extraction of features until the entire width is covered. Afterward, it moves down and starts from the left with the same specified stride. This process is repeated until the entire image is covered. The number of extracted features is defined through the process of parameter tuning. The resulted feature map is fed into the pooling layer, which reduces the spatial size of the convolved features. The output of the pooling layer is eventually flattened and used as an input to a fully connected neural network for the classification process.
- AE aims to reconstruct the input and output layers by having for fewer hidden layers than the input and output layers. It fulfills this by using an encoder that compresses data and reduces the dimensionality. AE can integrate CNN, RNN, and MLP networks as part of their architecture. It is mainly used for weights initialization in deep learning solutions. Stacked Auto-Encoders (SAEs) is a NN architecture that stacks multiple AEs, where the output of each AE is used an input to the next AE. During the training phase, SAE as a whole deploys a greedy layer-wise approach. AE is considered as an unsupervised learning algorithm and is usually used for feature extraction and reduction.

In recent years, researchers started to integrate deep learning to classify network traffic into their causal application or service group. In general, raw network traffic is used as an input to the deep learning algorithm to identify the spatiality or temporality of the network traffic. The advantage of deep learning is the elimination of feature engineering phase, unlike previous solutions, which makes it an easier and more convenient solution to implement without the need of a technical background of the targeted field. CNN is one form of the deep neural networks that aims to identify the spatial correlation of an input and has proven its efficiency in image recognition field. CNN simplifies the image to a simpler form for processing while avoiding the feature loss that might affect a good prediction. In order to fulfill this, the input image is denoted as a vector of pixel values and a filter vector is specified. The filter vector goes over the image by sliding from the top left to the right with a specific stride, allowing the extraction of features until the entire width is

covered. Afterward, it moves down and starts from the left with the same specified stride. This process is repeated until the entire image is covered. The number of extracted features is defined through the process of parameter tuning. The resulted feature map is fed into the pooling layer, which reduces the spatial size of the convolved features. The output of the pooling layer is eventually flattened and used as an input to a fully connected neural network for the classification process. Fig. 3 depicts the overview of CNN architecture, which has been widely used in the literature.

The deployment of deep learning classifiers has recently gained popularity in the literature to classify the network traffic, which is due to the manual feature engineering process. Researchers have deployed CNN for spatial correlation of the traffic representation and RNN for temporal correlation of the traffic representation. The input of the models used either the raw traffic or extracted statistical features of the collected flows.

Wang [94] was one of the first to address the classification of network traffic using deep learning algorithms. The framework deployed Artificial Neural Network (ANN) and Stacked Auto Encoder (SAE) to classify network traffic to their causal applications. Their privately collected network traffic contained more than 58 encrypted and non-encrypted protocols. The evaluation results showed the efficiency of the proposed solution and highlighted the importance of the first *n* bytes for the classification process. Wei et al. [95] evaluated the effectiveness in deploying CNN2D to classify raw network traffic according to their spatial correlation. First, network packets were grouped as unidirectional flows and bidirectional flows (sessions). The first 784 bytes were used for each generated traffic to support early stage identification. Zero padding was used for flows less than 784 bytes. Each flow was converted into images of 28*28 pixels and used as an input to the CNN classifier. Dataset USTC-TFC2016 has been used, which contains malicious and benign traffic for different applications. The evaluation results showed the effectiveness of the proposed solution in classifying and differentiating malicious and benign network traffic. Wang et al. [96] deployed CNN1D to classify network traffic, regardless of whether they were generated using VPN tunnel. The authors' argument was that network traffic is essentially sequential data and CNN1D is more effective for such a case. ISCX dataset that contains both VPN and non-VPN network traffic for various applications has been used for evaluation purposes. The experimental evaluation proved the better accuracy of CNN1D than CNN2D and C4.5 when using 784 bytes. Lopez-Martin et al. [97] integrated the spatial and temporal characteristics of network traffic to fulfill the classification purpose. CNN and LSTM have been used for spatial and temporal correlation, respectively. For image preparation, the first 20 packets were used; for each individual packet, six features were extracted, including the source port, destination port, the number of bytes in packet payload, TCP window size, inter-arrival time, and direction of the packet. RedIRIS dataset was used to evaluate the proposed solution, and the combination of CNN and LSTM provided the best results. Lotfollahi et al. [98] established a deep packet framework that contains CNN and SAE for network traffic classification. SAE architecture consisted of five fully connected layers, whereas two CNN1D layers were used for the CNN model. They noticed that most packets have a payload size of 1480 bytes, which was used for their experiments. The evaluation results showed the effectiveness of the proposed framework over ISCX dataset in terms of application identification and service group classification. Multi-Task Learning architecture based on CNN to classify network traffic has been addressed by Huang et al. [99]. The architecture contained three convolution layers that provide the highest accuracy performance.Using the CTU-13 and ISCX datasets respectively to evaluate malicious and benign traffic, the framework was able to classify the network traffic to their classes effectively using 1024 bytes (32*32 pixels images).

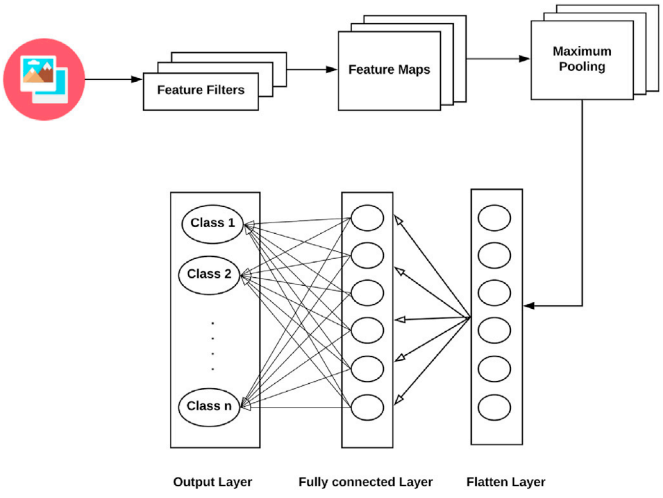

Fig. 3. CNN overview.

Wang et al. [100] proposed an SDN-HGW framework to support QoS network management in the smart home network. The framework deploys multilayer perceptrons, SAE, and CNN using ISCX datasets with data traffic generated from 15 applications. They have used 1480 bytes of each application traffic as their vector input to the proposed solution. The experimental evaluation showed the high accuracy with low computational cost, which is helpful to realize real-time decision. Aceto et al. [101,102] conducted a performance evaluation comparison between different deep learning solutions for mobile network traffic classification. They have privately collected an unbiased dataset at their labs, and the experimental evaluation was constructed for binary and multiclass classification. CNN1D with 784 bytes has shown the best accuracy performance. Zhou et al. [103] deployed the concept of the min-max normalization to convert network traffic flows features into gray images, which are used as the input of the CNN classifiers. The resulted images of Moore 249 statistical features were 16*16 pixels. The evaluation results using the Moore dataset led to a highly accurate classifier to classify network flows into their service group. Tong et al. [104] used a CNN classifier in conjunction with a supervised learning algorithm to classify QUIC-based network traffic. First, flow statistical features were extracted to classify google hangout services from the rest of the applications. Afterward, packet level features were extracted and fed into CNN1D as images to classify flows to their services. The evaluation accuracy results over the privately collected dataset proved the effectiveness of the proposed framework. Chen et al. [105] introduced a framework named seq2image which converts a network flow of an application into an image using RKHS kernel embedding by observing only 10 packets. An evaluation of a private dataset that contains five protocols showed the superiority of the proposed framework over the classical NB, SVM, and decision trees supervised learning algorithms.

Aceto et al. [106] introduced the Multimodal DL-based Mobile TraffIc Classification (MIMETIC) framework that effectively exploits the heterogeneous nature of mobile network traffic by learning both intra-modality and inter-modality dependences. The solution overcame the performance limitations of the single modality approaches as it captured traffic pattern from two different viewpoints, utilizing CNN and Gated Recurrent Unit (GRU) algorithms for payload and time-series features, respectively. For evaluation purposes, the authors privately collected iOS and Android applications' traffic represented in three different datasets. The solution scored 89.49% and 89.14% accuracy for Android and iOS traffic respectively by observing 576 bytes and 12 packets of each flow. The same authors [107] provided a taxonomy of the traditional statistical machine learning and deep learning classifiers, highlighting the attraction gain offered by the latter approach. Then, they provided a general deep learning-based framework to classify mobile network traffic for both iOS and Android applications. The general framework offered modularity in applying CNN or RNN algorithms.

Furthermore, it enabled the single modal or multi-modal input, using the first N bytes/packets and observing the content of packets' payload or header. The evaluation analysis over a generated human traffic dataset showed the efficacy of the proposed general framework compared to the base deep learning and machine learning classifiers.

Bu et al. [108] suggested a deep learning solution to classify network traffic based on the deep parallel Network-In-Network (NIN) with multiple MLP convolutional modules. The framework deploys a parallel decision strategy by building two sub-networks that distinctly process packet payload and packet header. Contrary to traditional CNN, NIN utilizes a micro-network after each convolution. It also deploys a global average pooling before the final classification, reducing the required model's parameters. When evaluated against the ISCX dataset, the proposed solution provided a better accuracy than the traditional CNN by observing 1480 bytes of the monitored flow. Liu et al. [109] introduced an end-to-end framework called Flow Sequence Network (FS-Net) to classify network traffic. It automatically identifies features by monitoring raw traffic without the need of manually identifying them. The framework contained an encoder, a decoder, and a softmax classifier. The encoder generates the features, the decoder reconstructs and restores the input sequence, and the softmax classifies the network flow. The solution has been evaluated against the dataset collected by Liu et al. [110], containing traffic for various applications. The evaluation results showed the outperformance of FS-Net in classifying network traffic compared to other state of the art solutions.

Table 6 summarizes the work conducted in the literature. The wide deployment of CNN to classify network traffic can be clearly seen. Besides, raw traffic as classifier input was widely used due to the elimination of the feature engineering requirement. It shows the efficacy of deep learning algorithms to classify network traffic for various types such as legitimate traffic, malicious traffic, and mobile traffic. The deep learning algorithms showed their feasibility in supporting real-time identification by observing the first few packets/bytes of the network flows.

7. Datasets

In order to build a classification model, datasets must be utilized for training and evaluation purposes. Researchers in the literature deployed either a privately collected dataset or a public dataset. We summarize eleven datasets that have been used in the literature in Table 7. The criteria of the dataset selection process were: the diversity of the collected network traffic, the availability of the PCAP/Attribute-Relation File Format (ARFF), the wide deployment of the dataset in the literature and the availability of the dataset.

- **Moore dataset** [112]. Network traces collected at Cambridge University labs in 2005. It consists of many applications, such as HTTP, FTP, MySQL and BitTorrent, which were collected and labeled to 10 service groups like P2P and multimedia games. Its availability in ARFF format with 249 extracted statistical features leads to its popularity.
- **USTC-TFC2016 dataset** [113]. A privately collected dataset that contains network traffic for malicious and benign applications. Each dataset contains traffic for 10 different applications in PCAP format, including FaceTime, Skype, Zeus, and Cridex.
- **ISCX dataset** [114]. Network traffic traces collected by the Canadian institute for cybersecurity in 2016. It contains network traffic for various applications that are grouped into seven service groups. VPN traffic represents applications' traffic that has been run through a VPN tunnel, while non-VPN traffic represents applications' traffic that has been run through a non-VPN tunnel.
- **CTU-13 dataset** [115]. A collection of botnet network traffic that has been collected at CTU University in 2011. It contains both malicious botnet and benign traffic from different malware and applications. Several C&C channel protocols are collected for the addressed malware.
- **MAWI dataset** [116]. MAWI is a working group under the WIDE project in Japan. It provides a joint project between Japanese academic institutions with corporates for traffic measurement analysis. This group has been collecting real network traffic since 2000 to date over the WIDE network at different sample points. The collected traces include benign and malicious traffic for multiple applications.
- **Auckland II dataset** [117]. A dataset collected by the University of Auckland. The traces have been collected mostly in 2000 and each lasted 24 h or less, where all non-IP traffic were discarded. The common collected protocols included HTTP, IMAP, POP, and SMTP.
- **University of Brescia (UNIBS) dataset** [118]: A three-day duration network traffic collected at the University of Brescia using 20 workstations in 2009. Most of the collected traffic is TCP, including BitTorrent, Skype, HTTP, and eDonkey.
- **University of Jinan (UJN) dataset** [119]. Two-day duration network traffic traces collected at the University of Jinan in 2013 using Traffic Labeler (TL). Network flows with less than 10 packets are filtered out. The collected traffic includes browsing, chat, cloud disk, and live updates using Windows host devices. The traces have been accurately labeled and evaluated for their causal application.
- **Anon17 dataset** [120]. Anonymous real network traffic collected between 2014 and 2017 at the Network Information Management and Security (NIMS) lab. Tor, JonDonym, and I2P tools have been used to generate the traffic, and Tranalyzer tool has been used to extract statistical features from them. In total, there are five ARFF

Table 6
Summary of deep learning solutions.

Study	Granularity	Algorithm	Input	Dataset	Early Stage Detection	Traffic Type
[94]	Application	SAE	Raw traffic	Private	N/A	Encrypted/Non-Encrypted
[95]	Application	CNN	Raw traffic	USTC-TFC2016	784 bytes	Malicious/Benign
[96]	Service group	CNN	Raw traffic	ISCX	784 bytes	VPN/Non-VPN
[97]	Application	CNN, LSTM	Packet level features	RedIRIS	10 packets	Mixed application traffic
[98]	Application/Service Group	CNN/SAE	Raw traffic	ISCX	N/A	VPN/Non-VPN
[99]	Application	CNN	Raw traffic	CTU-13/ISCX	1024 bytes	Malware/VPN/Non-VPN
[100]	Application	CNN/MLP/SAE	Raw traffic	ISCX	1480 bytes	VPN/Non-VPN
[101, 102]	Application	CNN, SAE, LSTM	Raw/Packet level feature	Private	784 bytes	Android/iOS application traffic
[103]	Service Group	CNN	Flow statistical features	Moore	No	Application to service group
[104]	Application	Random Forest/CNN	Flow/Packet features	Private	No	QUIC traffic identification
[105]	Protocol	CNN	Raw traffic	Private	10 packets	Various protocols
[111]	Application	CNN, GRU	Raw traffic, time series features	Private	576 bytes/12 packets	Android/iOS application traffic
[107]	Application	CNN, RNN	Raw traffic, time series features	Private	576 bytes/12 packets	Android/iOS application traffic
[108]	Application	NIN, MLP	Raw traffic	ISCX	1480 bytes	VPN/Non-VPN
[109]	Application	RNN, AE	Raw traffic	[110]	N/A	Mixed application traffic

Table 7
Summary of public datasets.

Dataset	# of Classes	Traffic Types	Year	Labeled	Format
Moore	10	Web, SMTO, POP3, FTP, DNS, BitTorrent, MySQL, Virus, Windows media player, Telnet, WOW	2005	Yes	ARFF
ISCX	7	HTTPS, SMTP, Facebook, Chrome, FTP, Skype, BitTorrent	2016	Yes	Full PCAP
USTC-TFC2016	2	Facetime, Gmail, Skype, Zeus, Cridex, Htbot	2016	Yes	Full PCAP
CTU-13	13	Botnet sample for each scenario	2014	Yes	Full PCAP, ARFF
MAWI	Varies	HTTP, FTP, SSH, DNS, SSL, DoS, other	2000–2020	No	Full PCAP
Auckland II	10	FTP, FTP-data, http, imap, pop3, smtp, nntp, ssh, dns, Telnet	2000	No	Packet header (No payload)
UNIBS traces	10	BitTorrent, Skype, HTTP, eDonkey, IMAP, POP3, MSN, SMTP, urd, SSH	2009	Yes	Full PCAP
UJN traces	9	Browsing, chat, cloud disk, live update, stream media, mail, ftp, p2p, other	2013	Yes	Ful PCAP
Anon 17	Varies	Browsing, IRC, streaming, Torrent, other	2014–2017	Yes	ARFF
Mirage	40	40 Android apps	2017–2019	Yes	JSON
UNSW	28	28 IoT devices	2016–2017	No	Full PCAP, CSV
MobileGT	12	12 Mobile apps	2016–2018	Yes	ARFF

datasets, including various classes such as browsing, IRC, streaming, and Torrent.

- **MIRAGE-2019 dataset** [121]. Android mobile application real dataset collected at the ARCLAB laboratory at the University of Napoli between 2017 and 2019 in JSON file format for each PCAP capture. Three Android devices named Xiaomi Mi5, Google Nexus 7, and Samsung Galaxy A5 are used to generate the mobile traffic for 40 applications.
- **UNSW IoT dataset** [122]. IoT real network traffic collected at the UNSW between 2016 and 2017 over a period of 26 weeks. The lab setup at the university's campus comprised of 28 smart devices that connect to the Internet gateway via a WiFi access point.
- **MobileGT dataset** [123]. Mobile apps network traffic collected in a monitored smartphone environment. The data includes Active Mobile Traffic Data (AMTD) and Passive Mobile Traffic Data (PMTD). AMTD have been collected through utilizing the mgtClient on 10 smartphones between 2016 and 2017 and PMTD have been collected in 2018. Twelve applications are used for traffic generation and the traces are available as ARFF format.

8. Discussion and challenges

The discussed four main network traffic classification approaches have shown their efficacy in identifying the used application, protocol, or service of the monitored network traffic. Table 8 lists the strengths, weaknesses, and challenges of the discussed techniques.

- Port-based identification accesses the packet's header and examines the utilized port number. This process is a very simple approach that can be implemented with low computational time and computational resources requirements. However, this approach does not offer fine granularity classification of a specific application and only identifies the used protocol in the monitored network.
- DPI overcomes port-based identification, where it provides application traffic classification in addition to protocol and service group identification. Furthermore, DPI has higher classification accuracy since it accesses the packets' content of the monitored flow. On the other hand, it has several disadvantages. First, it is a slow technique since the content of aggregate packets must be stored and examined against predefined signatures. Second, the solution demands high computational resources to store and examine the content, especially if the signature is spanned over multiple packets. Third, DPI cannot detect zero-day applications that have no signatures, where signatures must be updated to fulfill their detection. Fourth, accessing the content of users' traffic raises concerns of privacy and is considered a privacy breach in various countries.
- Supervised learning overcomes the shortages of DPI and port-based identification techniques. It is simple to implement, provides high accuracy, requires low computational resources, provides fine grain detection, and does not require access to the content of the network traffic since the statistical features of flows are used in building the classifiers. However, it has two disadvantages. The classifier's performance depends heavily on the examples used in the training dataset, where new application's detection requires the retrain of the built classifier. This disadvantage has been addressed, by selecting generic features to classify untrained new applications to their service groups, as shown in the previous study. Another disadvantage is feature engineering process. The network analysts must identify and extract features that best represent the collected network traffic that helps to distinguish network flows and correlate them with their causal applications, protocols, or service groups.
- Unlike the supervised learning algorithms, unsupervised learning does not require attaining a labeled dataset. This reduces the needed efforts of attaining a labeled dataset or labeling an already collected dataset. However, when compared to the supervised learning approach, it provides less accurate results.

Table 8
Traffic classification techniques comparison.

Technique	Advantages	Disadvantages	Challenges
Port-based	Simple, fast, low resources	Detects mostly protocol level	Randomization, masquerading
DPI	Accurate, fine grain detection	Slow, high computational resources, unable to detect undefined signatures, privacy breaching	Encryption
Supervised classification	Accurate, simple to implement, fine grain detection, fast	Performance depends on the trained dataset, feature engineering requirement	Adversarial example, imbalanced dataset, dataset collection, dataset labelling
Unsupervised clustering	Do not require labelled dataset, fine grain detection	Lower performance than the supervised classification	Adversarial example, dataset collection
Semi-supervised	Fine grain classification, higher accuracy than standalone supervised or clustering	Higher computational resources than supervised and supervised solutions	Adversarial example, dataset collection, dataset labeling
Deep learning	Feature engineering elimination, accurate, fine grain detection	The requirement of large amount of annotated data and computational resources	Adversarial example, dataset collection, dataset labeling

- Semi-supervised technique deploys the strengths of supervised and unsupervised solutions to fulfill the classification process. It provides high accurate results, deploys unsupervised learning algorithms for feature selection or labeling the dataset, and utilizes the supervised learning algorithm for classifying the network flows. The fact that it integrates two techniques leads to higher computational requirements for establishing the model.
- Deep learning has the superiority of eliminating the feature engineering process that is required by the statistical techniques, supervised, unsupervised, and semi-supervised algorithms. It has also shown its high accuracy in classifying network traffic with fine grain detection. On the other hand, this technique, especially in the training phase, requires a large amount of data to build the classifier and high computational resources.

In the literature, researchers face several challenges in applying the aforementioned techniques to classify network traffic. These challenges are related to a specific technique or various techniques, and affect the overall performance of the built classifier.

- **Randomization.** New applications utilize non-standard port numbers to deliver their network traffic. This poses a challenge for using port numbers to classify network traffic, especially for the port-based technique, as those numbers are not correlated to a specific application or protocol. In this case the false negative results of the classifier increase. To overcome this problem, packets headers and their contents must be inspected.
- **Masquerading.** Nowadays, malicious and benign applications deploy standard ports to deliver their network traffic. For example, Zeus botnet utilizes port 80 to deliver its command and control network traffic, bypassing port-based blocking/identification technique.
- **Encryption.** Ciphering network traffic provides a strong privacy measure for users to surf the Internet freely without worrying about breaching their privacy. Furthermore, cybercriminals deliver their malware payload via encrypting the network traffic. This hurdles the ability to classify the network traffic by inspecting the packets' content, especially for the DPI technique.
- **Dataset collection and labeling.** The most challenging part in building and evaluating a classifier is the collection of network traffic and labeling of a dataset. Dataset collection is not a trivial task because it must contain enough instances for the targeted applications to avoid imbalance dataset challenges, especially when building the classifier. Moreover, the absence of the targeted application traffic requires retraining the classifier to detect that application. The collection process may face challenges related to privacy concerns, including network traffic content and IP addresses as they relate to the identification of the user's behavior and the origin of the traffic. This impact is reduced by anonymizing the packet headers, including IP addresses, employing encryption, and obtaining consent from the users. Data labeling, which is the process of obtaining the ground truth of each generated flow, is not a trivial task and is considered to be a burden and time consumption for network analysts. DPI and port-based identification techniques are commonly utilized to fulfill such a task. Although public traces reduce the hurdle of data collection and labelling, they might neither provide the targeted application traffic nor provide a proper representation of the collected data.
- **Imbalance dataset.** The classification model must not be biased toward a specific class during the training phase. An imbalanced classification model appears when the number of instances in the training dataset for each class is not equally balanced, providing misleading performance results. This challenge appears clearly for labeled datasets used in supervised learning, semi-supervised learning and deep learning.
- **Adversarial example.** An adversarial example is an intentional modified input that causes the built classifier to provide a wrong decision. This challenge exists in machine learning techniques, especially deep learning classifier [124]. Various models using different architectures and trained on different subsets of the training dataset are pruned to misclassify the adversarial examples [125]. A few adversarial methods and techniques are used to fulfill the detection evasion, such as L-BFGS algorithm [126], Fast gradient sign method [127], Carlini and Wagner attack [128], and Jacobian-based saliency map approach [129].

9. Conclusion and future work

Network traffic classification aims to identify the used applications, protocols, or services in a monitored network. It is an important field for ISPs, corporates, and countries since it helps to apply QoS over real-time applications' traffic, block the usage of specific applications, fulfill lawful interception regulations, and detect malicious activities. Research communities have been active in proposing various solutions to fulfill the classification goal. Port-based identification is the earliest and simplest solution but suffers from randomization, masquerading, and general granularity identification. DPI is more fine grain and accurate than the port-based solution. However, it suffers from encryption and expensive resources requirement.

Supervised learning classifier combined with statistical features is a popular solution in the literature owing to its simplicity and high accuracy. The proposed solutions in the literature addressed the accuracy, computational resources and classification speed of the built classifiers, and the ability to support early stage detection and untrained versions identification. Unsupervised and semi-supervised learning solutions have been actively evaluated in the literature and they suffer from lower performance and expensive resources respectively when compared to the supervised approach. On the other hand, they overcome the shortcoming of the necessity of acquiring a large labeled dataset. Deep learning is the most recent technique and is gaining popularity since it eliminates the feature engineering process that is required by other machine learning algorithms. Researchers have deployed various either private or public datasets, which include traffic from various applications and protocols for evaluating the network classification models, and eight public datasets are described in this paper.

The survey has discussed the challenges of each technique and found that there is no single solution that provides a flawless performance in terms of accuracy, computational resources, speed, early stage detection, and evasion immunity. In order to address this issue, multilayer classification models could be integrated to overcome the shortages from each solution and fill their gaps. Besides, we found that both supervised and deep learning solutions were addressed intensively due to their simplicity and accuracy. The most common network traffic classification techniques have been discussed in this study. We aim to extend this study by discussing other techniques and frameworks aside from the addressed models. The survey raises questions for future directions that could be addressed by the research as follows:

- The deployment of supervised learning algorithms to detect untrained versions has been briefly addressed in the literature, especially for VoIP and Botnet traffic. The evaluation of this approach against various applications would reduce the shortcoming of the necessity of retraining the classifier to detect the untrained versions.
- The evaluation process of the built models has been conducted offline using a testing dataset. The performance address of the built model in a real environment with high scale network traffic is an important area to consider, which addresses the scalability of the built mode.
- The survey showed that the conducted literature focused solely on computer or mobile network traffic when building and evaluating the classifier. The model's performance in simultaneously classifying network traffic from various sources, including computers, mobiles, and IoT, would be an important field to address since most infrastructures nowadays host those devices on the same network.

- Deep learning solutions mainly addressed the spatial and temporal correlation of the raw network traffic to fulfill the classification goal. Statistical feature spatial and temporal correlation analysis in conjunction with deep learning has been slightly addressed in the literature showing promising results.

Declaration of competing interest

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

References

[1] E. Al Neyadi, S. Al Shehhi, A. Al Shehhi, N. Al Hashimi, Q. Mohammad, S. Alrabaee, Discovering public wi-fi vulnerabilities using raspberry pi and kali linux, in: 2020 12th Annual Undergraduate Research Conference on Applied Computing (URC), IEEE, 2020, pp. 1–4.

[2] International Telecommunication Union, Measuring digital development facts and figures. https://www.itu.int/en/ITU-D/Statistics/Documents/facts/FactsFigures 2019.pdf (accessed 25 June 2003).

[3] H. Mohajeri Moghaddam, Skypemorph: Protocol Obfuscation for Censorship Resistance, Master's Thesis, University of Waterloo, 2013.

[4] A. Azab, Classification of Network Information Flow Analysis (CONIFA) to Detect New Application Versions, Ph.D. thesis, Federation University, 2015.

[5] S. AlDaajeh, H. Saleous, S. Alrabaee, E. Barka, F. Breitinger, K.-K.R. Choo, The role of national cybersecurity strategies on the improvement of cybersecurity education, Comput. Secur. 119 (2022). ARTN 102754.

[6] S. Alrabaee, M. Al-Kfairy, E. Barka, Efforts and suggestions for improving cybersecurity education, in: 2022 IEEE Global Engineering Education Conference (EDUCON), IEEE, 2022, pp. 1161–1168.

[7] A. Azab, Packing resistant solution to group malware binaries, Int. J. Secur. Network. 15 (3) (2020) 123–132.

[8] S. Alrabaee, A stratified approach to function fingerprinting in program binaries using diverse features, Expert Syst. Appl. 193 (2022), 116384.

[9] P. Casey, M. Topor, E. Hennessy, S. Alrabaee, M. Aloqaily, A. Boukerche, Applied comparative evaluation of the metasploit evasion module, in: 2019 IEEE Symposium on Computers and Communications (ISCC), IEEE, 2019, pp. 1–6.

[10] A. Khraisat, A. Alazab, M. Hobbs, J.H. Abawajy, A. Azab, Trends in crime toolkit development, in: Network Security Technologies: Design and Applications, IGI global, 2014, pp. 28–43.

[11] S. Alrabaee, M. Debbabi, L. Wang, A survey of binary code fingerprinting approaches: taxonomy, methodologies, and features, ACM Comput. Surv. 55 (1) (2022) 1–41.

[12] M. Finsterbusch, C. Richter, E. Rocha, J.-A. Muller, K. Hanssgen, A survey of payload-based traffic classification approaches, IEEE Commun. Tutorial. 16 (2) (2014) 1135–1156.

[13] S. Valenti, D. Rossi, A. Dainotti, A. Pescapè, A. Finamore, M. Mellia, Reviewing Traffic Classification, Springer Berlin Heidelberg, 2013, pp. 123–147.

[14] F. Pacheco, E. Exposito, M. Gineste, C. Baudoin, J. Aguilar, Towards the deployment of machine learning solutions in network traffic classification: a systematic survey, IEEE Commun. Tutorial. 21 (2) (2019) 1988–2014.

[15] O. Salman, I.H. Elhajj, A. Kayssi, A. Chehab, A review on machine learning–based approaches for internet traffic classification, Annal Telecommun. 75 (11) (2020) 673–710.

[16] H. Tahaei, F. Afifi, A. Asemi, F. Zaki, N.B. Anuar, The rise of traffic classification in iot networks: a survey, J. Netw. Comput. Appl. 154 (2020), 102538.

[17] P. Wang, X. Chen, F. Ye, Z. Sun, A survey of techniques for mobile service encrypted traffic classification using deep learning, IEEE Access 7 (2019) 54024–54033.

[18] J. Zhao, X. Jing, Z. Yan, W. Pedrycz, Network traffic classification for data fusion: a survey, Inf. Fusion 72 (2021) 22–47.

[19] Internet Assigned Numbers Authority (IANA), Service name and transport protocol port number registry. https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml (accessed 1 July 2020).

[20] A.W. Moore, K. Papagiannaki, Toward the accurate identification of network applications, in: C. Dovrolis (Ed.), Passive and Active Network Measurement, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 41–54.

[21] A. Madhukar, C. Williamson, A longitudinal study of p2p traffic classification, in: 14th IEEE International Symposium on Modeling, Analysis, and Simulation, 2006, pp. 179–188.

[22] S. Sen, O. Spatscheck, D. Wang, Accurate, scalable in-network identification of p2p traffic using application signatures, in: Proceedings of the 13th International Conference on World Wide Web, WWW '04, Association for Computing Machinery, New York, USA, 2004, pp. 512–521.

[23] A. Azab, P. Watters, R. Layton, Characterising network traffic for skype forensics, in: 2012 Third Cybercrime and Trustworthy Computing Workshop, 2012, pp. 19–27.

[24] P. Khandait, N. Hubballi, B. Mazumdar, Efficient keyword matching for deep packet inspection based network traffic classification, in: 2020 International Conference on COMmunication Systems & NETworkS (COMSNETS), IEEE, 2020, pp. 567–570.

[25] X. Wang, J. Jiang, Y. Tang, B. Liu, X. Wang, Strid2fa: scalable regular expression matching for deep packet inspection, in: 2011 IEEE International Conference on Communications, ICC, 2011, pp. 1–5.

[26] S. Fernandes, R. Antonello, T. Lacerda, A. Santos, D. Sadok, T. Westholm, Slimming down deep packet inspection systems, in: IEEE INFOCOM Workshops, 2009, pp. 1–6.

[27] N. Hubballi, M. Swarnkar, $bitcoding$: network traffic classification through encoded bit level signatures, IEEE/ACM Trans. Netw. 26 (5) (2018) 2334–2346.

[28] N. Hubballi, M. Swarnkar, M. Conti, Bitprob: probabilistic bit signatures for accurate application identification, IEEE Trans. Network Service Manage. 17 (3) (2020) 1730–1741.

[29] M. Hall, Correlation-based Feature Selection for Machine Learning, Ph.D. thesis, The University of Waikato, 2000.

[30] M. Dash, H. Liu, H. Motoda, Consistency based feature selection, in: Pacific-asia Conference on Knowledge Discovery and Data Mining, Springer, 2000, pp. 98–109.

[31] C.A. Ratanamahatana, D. Gunopulos, Scaling up the naive bayesian classifier: using decision trees for feature selection, Appl. Artif. Intell. 17 (5–6) (2003) 475–487.

[32] H. Liu, R. Setiono, Chi2: feature selection and discretization of numeric attributes, in: Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence, 1995, pp. 388–391.

[33] I.H. Witten, E. Frank, Data mining: practical machine learning tools and techniques with java implementations, SIGMOD Rec 31 (1) (2002) 76–77.

[34] P. Christen, Evaluation of Matching Quality and Complexity, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 163–184.

[35] A. Azab, R. Layton, M. Alazab, J. Oliver, Mining malware to detect variants, in: 2014 Fifth Cybercrime and Trustworthy Computing Conference, 2014, pp. 44–53.

[36] Y. Reich, S.J. Fenves, in: D.H. Fisher, M.J. Pazzani, P. Langley (Eds.), Chapter 12 - the Formation and Use of Abstract Concepts in design Concept Formation, 1991, pp. 323–353.

[37] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297.

[38] D.D. Lewis, Naive (bayes) at forty: the independence assumption in information retrieval, in: C. Nédellec, C. Rouveirol (Eds.), Machine Learning: ECML-98, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, pp. 4–15.

[39] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[40] T. Bujlow, T. Riaz, J.M. Pedersen, A method for classification of network traffic based on c5.0 machine learning algorithm, in: 2012 International Conference on Computing, Networking and Communications (ICNC), 2012, pp. 237–241.

[41] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.

[42] S. Huang, K. Chen, C. Liu, A. Liang, H. Guan, A statistical-feature-based approach to internet traffic classification using machine learning, in: 2009 International Conference on Ultra Modern Telecommunications Workshops, 2009, pp. 1–6.

[43] N. Williams, S. Zander, G. Armitage, Evaluating Machine Learning Algorithms for Automated Network Application Identification, Tech. Rep, Centre for Advanced Internet Architectures (CAIA), 2006.

[44] Z. Fan, R. Liu, Investigation of machine learning based network traffic classification, in: 2017 International Symposium on Wireless Communication Systems, ISWCS, 2017, pp. 1–6.

[45] R. Yuan, Z. Li, X. Guan, L. Xu, An svm-based machine learning method for accurate internet traffic classification, Inf. Syst. Front 12 (2) (2010) 149–156.

[46] A. Jenefa, M.B. Moses, An Upgraded c5.0 Algorithm for Network Application Identification, in: 2018 2nd International Conference on Trends in Electronics and Informatics, ICOEI, 2018, pp. 789–794.

[47] K.L. Dias, M.A. Pongelupe, W.M. Caminhas, L. de Errico, An innovative approach for real-time network traffic classification, Comput. Network. 158 (2019) 143–157.

[48] R. Alshammari, A.N. Zincir-Heywood, An Investigation on the Identification of Voip Traffic: Case Study on Gtalk and Skype, in: 2010 International Conference on Network and Service Management, 2010, pp. 310–313.

[49] R. Alshammari, A.N. Zincir-Heywood, Machine Learning Based Encrypted Traffic Classification: Identifying Ssh and Skype, in: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–8.

[50] G. Sun, T. Chen, Y. Su, C. Li, Internet traffic classification based on incremental support vector machines, Mobile Network. Appl. 23 (4) (2018) 789–796.

[51] J. Cao, D. Wang, Z. Qu, H. Sun, B. Li, C.-L. Chen, An improved network traffic classification model based on a support vector machine, Symmetry 12 (2) (2020) 301.

[52] A.S. Khatouni, N. Zincir-Heywood, Integrating machine learning with off-the-shelf traffic flow features for http/https traffic classification, in: 2019 IEEE Symposium on Computers and Communications (ISCC), 2019, pp. 1–7.

[53] Argus: the network audit record generation and utilization system. https://qosient.com/argus/downloads.shtml (accessed 25 June 2020).

[54] Silk (system for internet-level knowledge). https://tools.netsa.cert.org/silk/ (accessed 25 June 2020).

[55] A. Finamore, M. Mellia, M. Meo, M.M. Munafo, P.D. Torino, D. Rossi, Experiences of internet traffic monitoring with tstat, IEEE Network 25 (3) (2011) 8–14.

[56] S. Burschka, B. Dupasquier, Tranalyzer: versatile high performance network traffic analyser, in: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), 2016, pp. 1–8.

[57] S. Dong, Multi class SVM algorithm with active learning for network traffic classification, Expert Syst. Appl. 176 (2021), 114885.

[58] A.A. Afuwape, Y. Xu, J.H. Anajemba, G. Srivastava, Performance evaluation of secured network traffic classification using a machine learning approach, Comput. Stand. Interfac. 78 (4) (2021), 103545.

[59] E. Ganesan, I.-S. Hwang, A.T. Liem, M.S. Ab-Rahman, Sdn-enabled fiwi-iot smart environment network traffic classification using supervised ml models, Photonics 8 (6) (2021) 201.

[60] J. Li, S. Zhang, Y. Lu, J. Yan, Real-time p2p traffic identification, in: IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference, 2008, pp. 1–5.

[61] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, K. Salamatian, Traffic classification on the fly, SIGCOMM Comput. Commun. Rev. 36 (2) (2006) 23–26.

[62] C. Gu, S. Zhang, Y. Sun, Realtime encrypted traffic identification using machine learning, J. SW 6 (6) (2011) 1009–1016.

[63] Y. Liu, J. Chen, P. Chang, X. Yun, A Novel Algorithm for Encrypted Traffic Classification Based on Sliding Window of Flow's First N Packets, in: 2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA), 2017, pp. 463–470.

[64] L. Peng, B. Yang, Y. Chen, Z. Chen, Effectiveness of statistical features for early stage internet traffic identification, Int. J. Parallel Program. 44 (1) (2016) 181–197.

[65] P. Branch, J. But, Rapid and generalized identification of packetized voice traffic flows, in: 37th Annual IEEE Conference on Local Computer Networks, 2012, pp. 85–92.

[66] A. Azab, R. Layton, M. Alazab, P. Watters, Skype traffic classification using cost sensitive algorithms, in: 2013 Fourth Cybercrime and Trustworthy Computing Workshop, 2013, pp. 14–21.

[67] A. Azab, O. Maruatona, P. Watters, AVOCAD: adaptive terrorist comms surveillance and interception using machine learning, in: 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering, TrustCom/BigDataSE, 2019, pp. 85–94.

[68] A. Azab, M. Alazab, M. Aiash, Machine Learning Based Botnet Identification Traffic, in: 2016 IEEE Trustcom BigDataSE ISPA, 2016, pp. 1788–1794.

[69] A. Azab, The effectiveness of cost sensitive machine learning algorithms in classifying zeus flows, J. Info. Comput. Security. 17 (3–4) (2021) 332–350.

[70] T.T. Nguyen, G. Armitage, A survey of techniques for internet traffic classification using machine learning, IEEE Commun. Tutorial. 10 (4) (2008) 56–76.

[71] M. Halkidi, Y. Batistakis, M. Vazirgiannis, Cluster validity methods: Part i, SIGMOD Rec 31 (2) (2002) 40–45.

[72] Y. Wang, Y. Xiang, J. Zhang, S. Yu, A novel semi-supervised approach for network traffic clustering, in: 2011 5th International Conference on Network and System Security, 2011, pp. 169–175.

[73] R. Dubin, A. Dvir, O. Pele, O. Hadar, I. Richman, O. Trabelsi, Real Time Video Quality Representation Classification of Encrypted Http Adaptive Video Streaming - the Case of Safari, 2016 arXiv:1602.00489.

[74] Y. Du, R. Zhang, Design of a method for encrypted p2p traffic identification using k-means algorithm, Telecommun. Syst. 53 (1) (2013) 163–168.

[75] H. Singh, Performance analysis of unsupervised machine learning techniques for network traffic classification, in: 2015 Fifth International Conference on Advanced Computing Communication Technologies, 2015, pp. 401–404.

[76] J. Zhang, Y. Xiang, W. Zhou, Y. Wang, Unsupervised traffic classification using flow statistical properties and ip packet payload, J. Comput. Syst. Sci. 79 (5) (2013) 573–585.

[77] A. Alalousi, R.R. Othman, M. Abualhaj, M. Anbar, S. Yaakob, A preliminary performance evaluation of k-means, knn and em unsupervised machine learning methods for network flow classification, Int. J. Electr. Comput. Eng. 6 (2016) 778.

[78] J. Höchst, L. Baumgärtner, M. Hollick, B. Freisleben, Unsupervised traffic flow classification using a neural autoencoder, in: 2017 IEEE 42nd Conference on Local Computer Networks, LCN, 2017, pp. 523–526.

[79] H. Alizadeh, A. Khoshrou, A. Zúquete, Traffic classification and verification using unsupervised learning of Gaussian mixture models, in: 2015 IEEE International Workshop on Measurements Networking, 2015, pp. 1–6.

[80] T. Wiradinata, A. Paramita, Clustering and feature selection technique for improving internet traffic classification using k-nn, J. Adv. Comput. Network. 4 (1) (2016) 24–27.

[81] J. Zhang, C. Chen, Y. Xiang, W. Zhou, A.V. Vasilakos, An effective network traffic classification method with unknown flow detection, IEEE Trans. Network Service Manage. 10 (2) (2013) 133–147.

[82] T. Glennan, C. Leckie, S. Erfani, Improved classification of known and unknown network traffic flows using semi-supervised machine learning, in: Information Security and Privacy, vol. 9723, Springer International Publishing, 2016, pp. 493–501.

[83] T. Bakhshi, B. Ghita, On internet traffic classification: a two-phased machine learning approach, J. Comput. Network. Commun. 2016 (2016) 21.

[84] A. Fahad, A. Almalawi, Z. Tari, K. Alharthi, F.S. Al Qahtani, M. Cheriet, Semtra: a semi-supervised approach to traffic flow labeling with minimal human effort, Pattern Recogn. 91 (2019), 1–12.

[85] C. Rotsos, J. Van Gael, A.W. Moore, Z. Ghahramani, Probabilistic graphical models for semi-supervised traffic classification, in: Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, IWCMC '10, Association for Computing Machinery, New York, USA, 2010, pp. 752–757.

[86] J. Erman, A. Mahanti, M. Arlitt, I. Cohen, C. Williamson, Offline/realtime traffic classification using semi-supervised learning, Perform. Eval 64 (9) (2007) 1194–1213.

[87] J. Gao, F. Liang, W. Fan, Y. Sun, J. Han, A graph-based consensus maximization approach for combining multiple supervised and unsupervised models, IEEE Trans. Knowl. Data Eng. 25 (1) (2013) 15–28.

[88] A. Almalawi, A. Fahad, Z. Tari, M.A. Cheema, I. Khalil, knnvwc: an efficient k-nearest neighbours approach based on various-widths clustering, in: 2016 IEEE 32nd International Conference on Data Engineering, ICDE, 2016, pp. 1572–1573.

[89] S. Zhao, Y. Zhang, P. Chang, Network traffic classification using tri-training based on statistical flow characteristics, in: 2017 IEEE Trustcom/BigDataSE/ICESS, 2017, pp. 323–330.

[90] J. Zhang, X. Chen, Y. Xiang, W. Zhou, J. Wu, Robust network traffic classification, IEEE/ACM Trans. Netw. 23 (4) (2015) 1257–1270.

[91] F. Noorbehbahani, S. Mansoori, A new semi-supervised method for network traffic classification based on x-means clustering and label propagation, in: 2018 8th International Conference on Computer and Knowledge Engineering, ICCKE, 2018, pp. 120–125.

[92] T. Ede, R. Bortolameotti, A. Continella, J. Ren, D. Dubois, M. Lindorfer, D. Choffnes, M. Steen, A. Peter, Flowprint: semi-supervised mobile-app fingerprinting on encrypted network traffic, in: Proc. Network and Distributed System Security Symposium (NDSS 2020), USA, 2020.

[93] M.A. Lopez, D.M. Mattos, O.C.M. Duarte, G. Pujolle, A fast unsupervised preprocessing method for network monitoring, Annal Telecommun. 74 (3) (2019) 139–155.

[94] Z. Wang, The applications of deep learning on traffic identification, BlackHat USA 24 (11) (2015) 1–10.

[95] W. Wang, M. Zhu, X. Zeng, X. Ye, Y. Sheng, Malware traffic classification using convolutional neural network for representation learning, in: 2017 International Conference on Information Networking (ICOIN), 2017, pp. 712–717.

[96] W. Wang, M. Zhu, J. Wang, X. Zeng, Z. Yang, End-to-end Encrypted Traffic Classification with One-Dimensional Convolution Neural Networks, in: 2017 IEEE International Conference on Intelligence and Security Informatics, ISI, 2017, pp. 43–48.

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

[98] M. Lotfollahi, R. Shirali hossein zade, M. Jafari Siavoshani, M. Saberian, Deep packet: a novel approach for encrypted traffic classification using deep learning, Soft Comput. 24 (2020) 1999–2012.

[99] H. Huang, H. Deng, J. Chen, L. Han, W. Wang, Automatic multi-task learning system for abnormal network traffic detection, Int. J. Eng. Technol. Learn. 13 (4) (2018) 4–20.

[100] P. Wang, F. Ye, X. Chen, Y. Qian, Datanet: deep learning based encrypted network traffic classification in sdn home gateway, IEEE Access 6 (2018) 55380–55391.

[101] G. Aceto, D. Ciuonzo, A. Montieri, A. Pescapé, Mobile encrypted traffic classification using deep learning: experimental evaluation, lessons learned, and challenges, IEEE Trans. Network Service Manage. 16 (2) (2019) 445–458.

[102] G. Aceto, D. Ciuonzo, A. Montieri, A. Pescapé, Mobile encrypted traffic classification using deep learning, in: 2018 Network Traffic Measurement and Analysis Conference, TMA, 2018, pp. 1–8.

[103] H. Zhou, Y. Wang, X. Lei, Y. Liu, A method of improved cnn traffic classification, in: 2017 13th International Conference on Computational Intelligence and Security, CIS, 2017, pp. 177–181.

[104] V. Tong, H.A. Tran, S. Souihi, A. Mellouk, A novel quic traffic classifier based on convolutional neural networks, in: 2018 IEEE Global Communications Conference, GLOBECOM, 2018, pp. 1–6.

[105] Z. Chen, K. He, J. Li, 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), 2017, pp. 1271–1276.

[106] G. Aceto, D. Ciuonzo, A. Montieri, A. Pescapè, Mimetic: mobile encrypted traffic classification using multimodal deep learning, Comput. Network. 165 (2019), 106944.

[107] G. Aceto, D. Ciuonzo, A. Montieri, A. Pescapé, Toward effective mobile encrypted traffic classification through deep learning, Neurocomputing 409 (2020) 306–315.

[108] Z. Bu, B. Zhou, P. Cheng, K. Zhang, Z.-H. Ling, Encrypted network traffic classification using deep and parallel network-in-network models, IEEE Access 8 (2020) 132950–132959.

[109] C. Liu, L. He, G. Xiong, Z. Cao, Z. Li, Fs-net, A flow sequence network for encrypted traffic classification, in: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, 2019, pp. 1171–1179.

[110] C. Liu, Z. Cao, G. Xiong, G. Gou, S.-M. Yiu, L. He, Mampf: encrypted traffic classification based on multi-attribute markov probability fingerprints, in: 2018 IEEE/ACM 26th International Symposium on Quality of Service, IWQoS, 2018, pp. 1–10.

[111] G. Aceto, D. Ciuonzo, A. Montieri, A. Pescapè, Mimetic: mobile encrypted traffic classification using multimodal deep learning, Comput. Network. 165 (2019), 106944.

[112] A.W. Moore, D. Zuev, Internet traffic classification using bayesian analysis techniques, SIGMETRICS Perform. Eval. Rev. 33 (1) (2005) 50–60.

[113] Wangwei, Ustc-tfc. https://github.com/echowei/DeepTraffic/tree/master/1.malware_traffic_classification/1.DataSet(USTC-TFC2016, 2016 (accessed 1 July 2020).

[114] The Canadian Institute for Cybersecurity, Vpn-nonvpn dataset (ISCXVPN2016). https://www.unb.ca/cic/datasets/vpn.html (accessed 1 July 2020).

[115] S. García, M. Grill, J. Stiborek, A. Zunino, An empirical comparison of botnet detection methods, Comput. Secur. 45 (2014) 100–123.

[116] MAWI, Mawi working group traffic archive. http://mawi.wide.ad.jp/mawi/ (accessed 1 July 2020).
[117] W.R. Group, Auckland ii. https://wand.net.nz/wits/auck/2/auckland_ii.php (accessed 1 July 2020).
[118] Unibs, Unibs, Data sharing. http://netweb.ing.unibs.it/ntw/tools/traces/ (accessed 1 July 2020).
[119] P. Lizhi, Z. Hongli, Y. Bo, C. Yuehui, W. Tong, Traffic labeller: collecting internet traffic samples with accurate application information, China Communications 11 (1) (2014) 69–78.
[120] Anon17, Network traffic dataset of anonymity services. https://web.cs.dal.ca/shahbar/data.html (accessed 1 July 2020).
[121] G. Aceto, D. Ciuonzo, A. Montieri, V. Persico, A. Pescapé, Mirage: mobile-app traffic capture and ground-truth creation, in: 2019 4th International Conference on Computing, Communications and Security, ICCCS), 2019, pp. 1–8.
[122] A. Sivanathan, H.H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, V. Sivaraman, Classifying iot devices in smart environments using network traffic characteristics, IEEE Trans. Mobile Comput. 18 (8) (2019) 1745–1759.
[123] R. Wang, Z. Liu, Y. Cai, D. Tang, J. Yang, Z. Yang, Benchmark data for mobile app traffic research, in: Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, MobiQuitous '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 402–411.
[124] M. Usama, A. Qayyum, J. Qadir, A. Al-Fuqaha, Black-box adversarial machine learning attack on network traffic classification, in: 2019 15th International Wireless Communications Mobile Computing Conference, IWCMC), 2019, pp. 84–89.
[125] K. Ren, T. Zheng, Z. Qin, X. Liu, Adversarial attacks and defenses in deep learning, Engineering 6 (3) (2020) 346–360, https://doi.org/10.1016/j.eng.2019.12.012.
[126] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, in: 2nd International Conference on Learning Representations, ICLR, 2014.
[127] I. Goodfellow, J. Shlens, C. Szegedy, Explaining and Harnessing Adversarial Examples, arXiv 1412.6572.
[128] N. Carlini, D. Wagner, Towards evaluating the robustness of neural networks, in: 2017 IEEE Symposium on Security and Privacy, SP, 2017, pp. 39–57.
[129] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, A. Swami, The limitations of deep learning in adversarial settings, in: 2016 IEEE European Symposium on Security and Privacy, EuroS P, 2016, pp. 372–387.