An Efficient Network Classification based on Various-Widths Clustering and Semi-supervised Stacking

ABDULMOHSEN ALMALAWI\(^1\) and ADIL FAHAD\(^2\)
\(^1\)School of Computer Science & Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia. (e-mail: balmalowy@kau.edu.sa)
\(^2\)College of Computer Science & Information Technology, Department of Computer Science, Al Baha University, Al Baha, Saudi Arabia.

Corresponding author: A. Almalawi (e-mail: balmalowy@kau.edu.sa).

This project was funded by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under grant No. (J: 48-611-1441). The authors, therefore, acknowledge with thanks DSR for technical and financial support.

ABSTRACT Network traffic classification is basic tool for internet service providers, various government and private organisations to carry out investigation on network activities such as Intrusion Detection Systems (IDS), security monitoring, lawful interception and Quality of Service (QoS). Recent network traffic classification approaches have used an extracted and predefined class label which come from multiple experts to build a robust network traffic classifier. However, keeping IP traffic classifiers up to date requires large amounts of new emerging labeled traffic flows which is often expensive and time-consuming. This paper proposes an efficient network classification (named Net-Stack) which inherits the advantages of various widths clustering and semi-supervised stacking to minimize the shortage of labeled flows, and accurately learn IP traffic features and knowledge. The Net-Stack approach consists of four stages. The first stage pre-processes the traffic data and removes noise traffic observations based on various widths clustering to select most representative observations from both the local and global perspective. The second stage generates strong discrimination ability for multiview representations of the original data using dimensionality reduction techniques. The third stage involves heterogeneous semi-supervised learning algorithms to exploit the complementary information contained in multiple views to refine the decision boundaries for each traffic class and get a low dimensional metadata representation. The final stage employs a meta-classifier and stacking approach to comprehensively learn from the metadata representation obtained in stage three for improving the generalization performance and predicting final classification decision. Experimental study on twelve traffic data sets shows the effectiveness of our proposed Net-Stack approach compared to the baseline methods when there is relatively less labelled training data available.

INDEX TERMS Internet traffic classification; Semi-supervised learning; Multiview.

I. INTRODUCTION

The classification of traffic refers to the categorizing of the network traffic flows into a group of various categories based on either the applications (such as HTTP, P2P), or the protocols (such as UDP, TCP, IMAP). In the recent years, traffic classification methods help network administrators and Internet Service Providers ISPs as essential tools \([1]\) to monitor network, policy violations and network management for proficient planning and design of the network. It is usually deployed along with other preventive security mechanisms as a secondary barrier to block any unexpected or abnormal incident as well as undesired traffic. Over the last ten years, the research community and the networking industry have examined, proposed, and produced considerable schemes for network traffic classification. Fig. 1 shows the progression of various traffic classification strategies from year 1992 until 2021. We utilized Microsoft Academic to gauge the volume of publications in the field of computer science that matched the phrase “traffic categorization,” “traffic flows,” or “traffic identification”.

Internet service providers (ISPs) have employed traditional classification methods \([1]\) such as port-based planes in conjunction with deep packet inspection techniques to better manage their networks and deliver extra services to their consumers. Previous studies have demonstrated that traditional approaches failed to detect unknown threats in emerging.
cies, network classification approaches have been applied in network traffic classification to automatically monitor and analyze traffic data, which are obtained from the first stage. In the third stage, we introduced semi-supervised multi-view representation learning methods to exploit the statistical property of each view by applying heterogeneous semi-supervised learning algorithms directly to each single-view, which was obtained from the previous stage, to improve the learning process’s efficiency and prediction.

In order to avoid the complex correlations underlying different views and the over-fitting problem of semi-supervised learning algorithms, the final stage of our Net-Stack approach uses the metadata, obtained from the third stage, as the training set as well as an N-fold cross validation approach to jointly optimize all the functions of the meta-classifiers for improving the generalization performance and obtaining the final accurate and stable class label. To evaluate our suggested Net-Stack technique, we used 12 publicly available traffic data sets [10]–[13]. The experimental outputs indicate that our approach attained better result outperformed the relevant baseline methods, such as Semi-Supervised for Traffic Labeling (SemTra), Probabilistic graphical model (PGM), Offline/real-time semi-supervised classification (ORTSC), Bipartite graph-based maximizing (BGCM), and Various-Widths Clustering ($k$NNVWC).

This paper is organized as follows: The related work is summarized in Section II. In Section III, we introduce our proposed Net-Stack approach. In Section IV, we evaluate and discuss our proposed approach by comparing it with baseline methods. In section V, we conclude the paper and introduce future works.

II. RELATED WORK

Network traffic classification is the cornerstone of infrastructure management, service differentiation, security monitoring, traffic analysis, prediction and engineering. Machine learning based approaches have been applied in network traffic classification to automatically monitor and analyze the behaviors of network traffic with no or limited domain experts’ interference. Most of these approaches are based on various-Widths clustering and semi-supervised stacking, namely Net-Stack approach. As some network network traffic classifiers based on machine learning algorithms can be noise-fragile [9], the first stage of our proposed Net-Stack approach discards noise traffic flow from the original traffic data. This is performed by the use of the various-Widths clustering algorithm that partitions the original network traffic data into homogeneous clusters using a learned global width. Afterwards, a recursive various-Widths clustering process is carried out on every single formed cluster whose size exceeds the predefined criteria using its own local width. Finally, the nearest observation to each cluster centroid is proposed as informative and representative traffic flows. The second stage of Net-Stack approach inherits the advantages of several heuristics as dimensionality reduction methods to compensate for the deviation caused by insufficient information in a single perspective via building comprehensive multi-view demonstrations of the original traffic data, which are obtained from the first stage. In the third stage, we introduced semi-supervised multi-view representation learning methods to exploit the statistical property of each view by applying heterogeneous semi-supervised learning algorithms directly to each single-view, which was obtained from the previous stage, to improve the learning process’ efficiency and prediction.

In particular, training and testing are the two steps of the statistical classification approaches [2]–[6] of IP flows based on machine learning algorithms. The first phase feeds statistical characteristics data from IP flows (such as mean and the variance of size of packet along with the traffic flows duration of inter-packet) to an algorithm of machine learning (such as SVM, Naive Bayes, Neural Network, etc) to create classifier models with the testing phase using the generated model from the training phase to predict future application types. Based on the given class labels, two primary kinds of machine learning algorithms can be utilized for training as well as testing. With labelled data, such as supervised learning algorithms are employed; with unlabelled data, however, unsupervised learning algorithms are utilized.

Few models of the semi-supervised-classification have been proposed [7], [8] to tackle the issues of both supervised and unsupervised classification approaches, particularly with the rapid growth of new traffic applications. These methods increase traffic classification performance by combining a small amount of labelled data together with a higher number of unlabelled data. However, the majority of these models are prone to inaccuracies and inefficiency. This is due to two factors: (i) the notion that unlabeled flows must be categorised or assigned to handle traffic classes, and (ii) the failure to detect new threats and applications. As a result, instead of the time-consuming and costly human labeling technique, a semi-supervised approach using a minimum labelled traffic data and a large amount of unlabelled traffic data is needed for accurate network classification.

In this research work, we present an accuracy and efficiency network classification based on Various-Widths clustering and semi-supervised stacking, namely Net-Stack approach. As some network network traffic classifiers based on machine learning algorithms can be noise-fragile [9], the first stage of our proposed Net-Stack approach discards noise traffic flow from the original traffic data. This is performed by the use of the various-Widths clustering algorithm that partitions the original network traffic data into homogeneous clusters using a learned global width. Afterwards, a recursive various-Widths clustering process is carried out on every single formed cluster whose size exceeds the predefined criteria using its own local width. Finally, the nearest observation to each cluster centroid is proposed as informative and representative traffic flows. The second stage of Net-Stack approach inherits the advantages of several heuristics as dimensionality reduction methods to compensate for the deviation caused by insufficient information in a single perspective via building comprehensive multi-view demonstrations of the original traffic data, which are obtained from the first stage. In the third stage, we introduced semi-supervised multi-view representation learning methods to exploit the statistical property of each view by applying heterogeneous semi-supervised learning algorithms directly to each single-view, which was obtained from the previous stage, to improve the learning process’ efficiency and prediction.

In order to avoid the complex correlations underlying different views and the over-fitting problem of semi-supervised learning algorithms, the final stage of our Net-Stack approach uses the metadata, obtained from the third stage, as the training set as well as an N-fold cross validation approach to jointly optimize all the functions of the meta-classifiers for improving the generalization performance and obtaining the final accurate and stable class label. To evaluate our suggested Net-Stack technique, we used 12 publicly available traffic data sets [10]–[13]. The experimental outputs indicate that our approach attained better result outperformed the relevant baseline methods, such as Semi-Supervised for Traffic Labeling (SemTra), Probabilistic graphical model (PGM), Offline/real-time semi-supervised classification (ORTSC), Bipartite graph-based maximizing (BGCM), and Various-Widths Clustering ($k$NNVWC).

This paper is organized as follows: The related work is summarized in Section II. In Section III, we introduce our proposed Net-Stack approach. In Section IV, we evaluate and discuss our proposed approach by comparing it with baseline methods. In section V, we conclude the paper and introduce future works.
on predefined classes labelled by experts in order to classify network traffic. However, it is time-consuming for experts, if not impossible, to label all data for network traffic classification. Therefore, the application of Artificial Intelligence (AI) and machine learning techniques has been used to develop promising solutions for effective network management. Recently, there have been many approaches classify network traffic. Traditional traffic classification approaches can be broadly divided into two broadly categories, namely, port-based techniques [14] and payload-based techniques [15]. However, the former and the latter have many limitations. This is because in the former category, network traffic are classified based on publicly known ports, while the latter cannot classify encrypted traffic. To overcome the aforementioned limitations machine learning based techniques such as behaviour-based, statistics-based, and correlation-based have been proposed. The learning mechanism for all the existing work are categorised into three categories: supervised, semi-supervised and unsupervised [16], [17].

A. SUPERVISED LEARNING

The supervised learning techniques are well-known for their high performance; however, they require labelled data, which is a challenging requirement because labelling process is time-consuming and prohibitively expensive as it requires expert involvement. In recent years, Support Vector Machines (SVM) has received a great attention in the literature for network traffic classification. However, there are some limitations related to computational cost and memory usage in the training and testing processes. Therefore, Guanglu Sun et al. [18], proposed an Incremental Support Vector Machines (ISVM) approach based on SVM to address the aforementioned limitations. The ISVM only saves the Support Vectors (SVs) and periodically combines the new data with the SVs to keep the model updated over time. Feature extraction algorithm for fast network traffic classification is proposed in [19]. The proposed approach is mainly based on a feature normalization and a correlation-based feature extraction techniques. The experimental results showed promising performance with four well-known machine learning classifiers. Similarly, Shafiq et al. [20] proposed a hybrid feature selection algorithm that adapted two metrics, namely, area under ROC curve and weighted mutual information to obtain near optimal features from a network traffic flow. The experiment results demonstrated the efficiency of the proposed method with 11 classifiers. Tong et al. [21] proposed an online network traffic classification approach based on Entropy-MDL (Minimum Description Length) discretization algorithm and C4.5 decision tree algorithm using flow level features instead of packet level features. The authors practically selected the near optimal features by grouping the features in a different combination. The authors claimed that their approach is an efficient for a real time classification because they obtained the highest accuracy from the first four packets of a flow. In [22], a real time network classification approach based on parallelized Convolutional Neural Networks is proposed using the Spark platform. The results showed significant accuracy and while the classification time is greatly reduced as well.

B. UNSUPERVISED LEARNING

Unlike, supervised learning, in the unsupervised learning, we do not require domain experts to label training data. Therefore, this type of learning is widely used in network traffic classifications. However, this type is still suffering from low accuracy and efficiency. Shasha Zhao et al. [23] proposed unsupervised approach based on both the Self-Organizing Maps (SOM) and the K-means clustering algorithm. The SOM was adapted to determine the near optimal number of clusters. Then, the K-means was used to do the classification phase. The determination of the number of the clusters through the SOM has increased and reduced the classification accuracy and the computational cost of clustering respectively. The work in [24], proposed an unsupervised classification approach which is based on flow-based features and packet payloads statistics in the training phase, while in the testing phase, the authors used flow statistical strategies for the classification of traffic flows. The network traffic is clustered into small clusters and then the generated small and similar clusters are merged into a few large clusters as per their payload content by adopting a bag-of-words model to represent the clusters. Latent semantic analysis (LSA) is used to reduce dimensionality in order to able to analyse similarity of clusters. Jonas Höchst et al. [25] discusses an unsupervised neural autoencoder network traffic classification approach using statistical flow-based features and clustering technique. The neural autoencoder is adapted to cluster network traffic into specific mobile applications. In [26], a framework of clustering approaches such as gaussian mixtures (GM), K-Means, and hierarchy-based clustering algorithm named BIRCH is proposed. This framework extracts some network traffic properties within a given time window in addition to some features extracted from transport layer. The k-means clustering algorithm is proposed to infer Quality of Service (QoS) elicited from network-usage profiling under a wide scale of wireless network. The proposed approach groups similar profiles for users in clusters in order to automatically identify the quality of service of each profile [27].

C. SEMI-SUPERVISED LEARNING

Although supervised techniques are highly accurate and reliable for network traffic classification, they are costly and inefficient because the required expert domain knowledge. On the other hand, unsupervised-based techniques are proposed for avoiding the need of domain experts for labelling, but they are influenced by low efficiency and poor accuracy. Therefore, in the literature, semi-supervised approaches are proposed to take full advantages of both supervised and unsupervised techniques. In practice, semi-supervised techniques only require a limited amount of labelled data and therefore these techniques can overcome the challenges associated with labelling large data sets. Mahdavi et al. [28]
proposed semi-supervised approach that is based on the graph theory and minimum spanning tree algorithm to cluster the network traffic observations, which are unlabelled data except small portion, into a number of clusters. The resulting clusters are labelled based the labelled observations located in each cluster. If the cluster does not contain any labeled observations, then it should be manually labeled by expert. Afterwards, the labelled observations are used to build the network classification model using the C4.5 algorithm. Ran et al. [29] proposed semi-supervised network traffic classification approach that is mainly based on k-means algorithm. The proposed approach can adaptively select the near optimal flow features using the small labeled data. In addition, the initial clustering centers are automatically leaned by calculating the centroids of each class in the small labeled data. The proposed optimised k-means algorithm generates a number of clusters ranging from specific numbers. The minimum number is chosen based on the number of classes in the labeled data, while the maximum is based on heuristic procedure. The determination of the near optimal number of clusters is based on a probability mechanism that maps the generated clusters to the predefined classes (applications or protocols). However, if one of the generated clusters does not have any labeled observations, it is defined as unknown category and a manual inspection is carried out to help improve the quality of training phase. The classification of zero-day applications is one of the challenges that has attracted the interest of many researchers. Therefore, Zhang et al. [30] proposed a classification strategy that considers the improvement of the classification accuracy in the presence of zero-day application traffic. The authors adapted random forest and k-means algorithms in their framework to perform supervised and unsupervised learning mechanisms. The proposed framework involves two steps. The K-means algorithm is adapted in the first step to cluster the training data set that involves the labelled and unlabelled observations, into a large number of clusters. Therefore, the resulting clusters that do not contain any labelled observations, indicating that they are zero-day traffic clusters. In the second step, the random forest algorithm is proposed to address the issue of the high false positive and false positive rates of zero-day unknown detection process in the first step. Ede et al. [31] proposed a classification technique to classify applications in encrypted mobile network traffic. The proposed technique is able to identify zero-day mobile applications. The authors extracted some possible features from the network traffic and scored all features according to the Adjusted Mutual Information (AMI) in order to obtain highly ranked features. The proposed technique is based on the assumption that each mobile application communicates with a static set of network destinations. Therefore, different modules corresponding to different patterns are proposed for discovering unknown applications. Deep Convolutional Generative Adversarial Network (DCGAN) is adapted to classify encrypted network traffic in [32]. In addition to few labeled observations, the proposed approach used a combination of some samples generated by DCGAN generators and unlabelled observations to build a robust classification model. In [33], another semi-supervised approach is proposed developed on the x-means clustering algorithm in addition to a new label propagation technique where the x-means clustering algorithm is stemmed from Bayesian Information Criterion (BIC) metric is adapted to cluster the training data set, which is based on a mixture of labeled and unlabelled observations, into a number of clusters. Afterwards, the proposed labelling technique based on k-nearest labelled neighbours is used to label each unlabelled observation.

III. THE PROPOSED NET-STACK APPROACH
In this paper, a semi-supervised stacking-ensemble learning approach is proposed to efficiently and accurately classify network traffic. The proposed approach combines the strengths of both semi-supervised learning and meta-learning techniques. The subsequent subsections will explain each step of the proposed approach.

A. APPROACH OVERVIEW
The supervised learning techniques are well-known for their high performance; however, they require labelled data, which is challenging requirement because labelling process is inefficient and prohibitively costly as it needs expert involvement. To address such as issue, in this section we have proposed a novel semi-supervised stacking-ensemble learning approach that limits the high effort of domain experts in labelling processes for network traffic data.

Fig.2 demonstrates four essential components in the proposed approach, namely (i) various width clustering technique to generate micro-clusters that represent a summary of data distribution in the space from the traffic data, (ii) linear-nonlinear feature projection techniques to increase diversity and enhance the generalizability of traffic data, (iii) a semi-supervised stacking-ensemble learning technique to remarkably improve classification and prediction accuracy by making effective use of the mixtures of labeled and unlabelled, and (iv) meta-learning process that can effectively label the unlabelled data.

B. VARIOUS-WIDTHS CLUSTERING LAYER
In this layer we use the VWC algorithm to produce micro-clusters from the network traffic data. Each cluster is assumed to represent different distribution in the dimensional space. As illustrated in Fig 3, the generated clusters vary in size, distributions and radii. The clustering process is carried out through three main stages including: (i) the process of learning nearly optimal cluster-width, (ii) partitioning and (iii) merging processes. All these stages are connected and executed serially till the criteria are satisfied.

1) Cluster-width selection
One of the important steps of the VWC algorithm is the determination of the width parameter value that will be used in the clustering
phase. This value will be determined by the following formula:

$$w = \frac{1}{r} \sum_{i=1}^{clslWidth(NN_k(H_i), H_i)}$$

(1)

where, $NN_k(H_i)$ are the function of $k$-nearest neighbours for the observation $H_i$, $clslWidth$ is the function that determines the width (radius) of $NN_k(H_i)$. The width value will be calculated based on the Euclidean distance measure that will be used to calculate the distance between the observation $H_i$ and the farthest observation among its neighbors. As used in [2], the value of $k$ is set to $50\% \times |Dataset|$ to guarantee a large cluster. Then, we will compute the radii (widths) of a set of observations that are randomly drawn from the dataset using the function $clslWidth$. Finally, the average widths are will be used as a global width for the dataset.

2) Partitioning

In this process, we will use the determined width to partition a dataset into several clusters. However, the partitioning process will be repetitively applied to each resultant cluster that exceeds a user-defined threshold using a new radius (or width). In other words, the partitioning process will be repeated until the size of the biggest cluster becomes less as compared to a user-defined threshold. For more details about the partitioning process, we refer to [2].

3) Merging

The recursive partitioning process of the data set into several clusters can proceed to the cluster creation which is present within another cluster. Therefore, we will perform the merging process through which we can minimise the overlaps among clusters as much as possible. This process can result in nearly distinct clusters by which we can learn the most-representative observations. For more details about the merging process, we refer to [2].

4) Selection of Candidate observation

Algorithm 1 presents steps for candidature process of observations. Prior to the steps of the algorithm, let $d$ be the Euclidean distance between two observations $x_j = \{x_{j,1}, x_{j,2}, \ldots, x_{j,n}\}$ and $x_k = \{x_{k,1}, x_{k,2}, \ldots, x_{k,n}\}$

$$d(x_j, x_k) = \sqrt{\sum_{i=1}^{n} (x_{j,i} - x_{k,i})^2}$$

(2)

It has been demonstrated in the literature that Observation reduction techniques can reduce storage requirements as well as time complexity needed for machine learning processes. In addition, they can improve generalization accuracy as they showed robustness against noise and over-fitting [34], [35]. Therefore, we adapted $kNNVWC$ [36] as observation reduction technique to increase the efficiency of the labelling process because this method has shown a great success in clustering data set into homogeneous small clusters. Below, we describe steps involved in the extracting the most-representative observations using $kNNVWC$ algorithm.
Let \( D \) be traffic data, where \( D = \{d_1, \cdots, d_n\} \). Let \( F \) be a vector of values of \( d_1, F = \{f_1, f_1, \cdots, f_m\} \). Let \( O_i \) denotes an observation of \( D, O_i = \{o_{i1}, o_{i2}, \cdots, o_{ij}, \cdots, o_{in}\} \) where \( m \) is the number of observations, and \( n \) the number of features. The observation \( O_i \) is either labelled or unlabelled. Let \( C = \{C_1, C_2, \ldots, C_n\} \) be the generated clusters by kNNVWC. Each cluster may have labelled or unlabelled observations \( C_i = \{\mathbf{S}^L_1, \mathbf{S}^L_2, \cdots, \mathbf{S}^L_m\} \). Since we only aim to extract small and most-representative observations, we propose the nearest observation for each cluster centroid as a representative for all members of that cluster.

If the nearest observation is unlabelled and the majority of observations in the cluster are also unlabelled then no label assigned to this observation, otherwise majority vote scheme is used whereas the class label that receives the largest number of votes is assigned to this observation. Let \( L = \{l_1, l_2, \cdots, l_{n+1}\} \) be a set of class labels, \( l_{n+1} \) represents the class label of unlabelled observations. The class label for the representative observation \( s_i \) if it is unlabelled is defined as follows:

\[
Label(s_i) = \arg \max \sum_{i=1}^{L} \sum_{j=1}^{C_i} Class(s_j, l_i), s_i = l_{n+1} \tag{3}
\]

C. MULTI-VIEW LAYER

In the literature, multi-view learning paradigm performs better than single-view learning to achieve a better generalization performance in machine learning by exploiting the relationship among multiple views from multiple features [37]–[39]. However, the application of this approach is not possible in network classification because network traffic data is composed of one single view. For leveraging the process for labelling network traffic data, a multi-view approach is proposed to generate multiple representations of the original traffic data that has been produced by various-widths clustering layer, whereas each representation is separated into several distinct features. This is because interesting hidden patterns cannot be obtained by analyzing from a single data collection. Moreover, each representation may have distinct characteristics of network traffic data.

The proposed approach uses feature dimensionality reduction techniques to generate multi-view representations for the network traffic data based on various heuristics. The advantage of dimensionality reduction is not only to improve the accuracy of the labeling process for network traffic data, but also to achieve several benefits: (i) it is a promising method that would be useful to reduce noise in the spatial and spectral dimensional space and achieves better classification and predication results, (ii) it is used to facilitate the meaningful visual interpretation of high-dimensional data for performing exploratory data analysis, (iii) it is used to build efficient and robust classification and clustering algorithms that require less computation time and memory space, and (iv) it reduces the possibility of over-fitting by building combinations of the features.

There are two categories of nonlinear dimensionality reduction techniques, namely global and local. The former attempts to preserve embeddings in which all observations meet a given criterion, while the latter techniques construct embeddings in which all local observations meet a given criterion. The global paradigm of nonlinear dimensionality reduction is adapted in this approach because the global methods are likely to produce a more constant representation of the data’s structure and their metric-preserving properties which are theoretically easier to understand. As a result of intensive experiments, we have chosen three global dimensionality reduction methods in our proposed multi-view approach, including Isomap [40], random projections (RP) [41] and kernel principal component analysis (KPCA) [42].

The proposed approach is manifold learning which aims to provide a global optimal solution. In addition, adopting the geodesic manifold distances approach between the data observations optimally conserves the original structure of the non-linear data. Specifically, the principal steps involved in...
generating a latest outlook of the network traffic data are described in brief as follows:

1) **Isomap**

To build a neighborhood graph. In order to consider two flows as neighbors, one of the two predefined conditions must be true, either the distance of the two flows is less than a constant in the original datasets, or one of the two flow affiliates to the \( k \) nearest neighbors (KNN) of the other. Subject to this neighborhood details, a weighted graph is set up that holds all the flows in the datasets.

- Employ Dijkstra’s algorithm [43] on the neighborhood graph to calculate the shortest path distances. This results in a matrix manifesting the geodesic distances for each set of two flows.
- A classical Multidimensional scaling (MDS) is applied on the resultant matrix from the preceding step in order to build data embedding. This ensures to enhance conserves of the estimated original geometry of the manifold.

2) **Random Projections (RP)**

It is an exceptionally robust technique to scale down the dimension by using random projection matrices. Moving forward for building a new view of the traffic data, we formally discuss the steps involved in the random projection approach.

- Convert the observations in the network traffic data \( D \in \mathbb{R}^p \) into a scaled down dimensional space \( F \in \mathbb{R}^q \), where \( q \ll p \), via
  \[
  F = RD
  \]
  \[\text{(4)}\]

  The size of the output matrix \( M \) is \( q \times N \), where \( q \) is the number of dimensions and \( N \) is the size of the traffic data.

\[
G = (x_1 | x_2 | \cdots | x_N)
\]

  \[\text{(5)}\]

- Initiate \( j \) random matrices \( \{R_i\}_{i=1}^j \), where \( j \) refers to the number of required views. Following are the two familiar methods to create the random numbers: (i) The vectors are evenly spread on the \( q \) dimensional unit sphere and (ii) the items for the vectors are selected using a Bernoulli +1/-1 distribution. Here the vectors are being normalized

- Columns are normalized such that their \( l_2 \) norm shall be 1.

- Project the matrix \( G \) onto the random matrices \( \{R_i\}_{i=1}^j \), i.e. \( T_i = R_i \cdot G \), where \( i = 1, \ldots, j \), to get \( \{T_i\}_{i=1}^j \) views.

3) **Kernel Principle Component Analysis (KPCA)**

This is a common principal statistical method to extract feature and performing data modeling. This approach uses the kernel trick for modeling of the non-linear structures in the data of network traffic. In the following, we discuss about the steps to be taken in order to carry out KPCA on the data network traffic data:

- Select a kernel mapping \( K(x_m, x_n) \) (such as Sigmoid, Polynomial and Gussian).
- Out of the original traffic data, take a matrix of \( N \times N \) kernel items (called \( K \)).
- Unfold the eigenvalue problem of \( K \) in order to uncover the eigenvectors \( a_i = [a_{1(i)}, \ldots, a_{N(i)}]^T \), as well as the eigenvalues \( \lambda_i \) of the covariance matrix in the feature space.
- Against each defined observation \( X \) within the feature space, get the principal components:
  \[
  (f(x) \cdot \phi_i) = \sum_{n=1}^{N} a_{n(i)} k(x, x_n)
  \]
  \[\text{(6)}\]

- Apply a standard PCA in order to retain just a compact figure of principal components that are equivalent to the large-scale eigenvalues with no compromising on the information.

Based on the consideration of the multi-view layer design plan, the dimension scaling down approach that may be adopted to produce diverse presentations of the actual traffic data should not be restricted to what was expressed so far.

**D. SEMI-ENSEMBLE LAYER**

Semi-supervised learning has demonstrated significant results in various fields by utilizing just a small data via pre-built label class. On account of the complexity while dealing with the bulk volume of the network traffic data generated, we move forward to propose a Semi-ensemble Layer, which acquire knowledge from the various illustrations of the formerly generated traffic data by the multi-view layer. Later, meta-level data is generated which is a collection of the forecasted values of the label less data. A representation of semi-ensemble Layer process is shown in Fig. 4. The semi-ensemble refers to a class of semi supervised classifiers \( B^{semi}_j \), where \( \{Rep_1, rep_2, \cdots, rep_n\} \) are the input, which are extracted by the previous layer, of each semi-supervised classifier. The meta-level data is the output that represents the the predicted values of the unlabeled data with automatically assigned labels. In addition to the label assignment, it is presumed that each semi supervised classifier forecasts a probability distribution for the best-fit class values in the Semi-ensemble Layer. Therefore, when the forecasted semi-Supervised classifier \( B^{semi}_j \), is practiced for traffic flow \( x \) is refer as probability distribution \( P^{B_{semi}}(x) = \{P(c_1 \mid x), P(c_2 \mid x), \ldots, P(c_m \mid x)\} \) where \( \{B^{semi}_1, B^{semi}_2, \ldots, B^{semi}_j\} \) indicated a set of Semi-Supervised classifiers, while \( \{c_1, c_2, \ldots, c_m\} \) represents the set of possible class values and \( P(c_1 \mid x) \) indicates the traffic flow probability that \( x \) matches to class \( c_1 \) as forecasted by a semi-supervised classifier \( B^{semi}_1 \). In order to attain the meta-level data, we adopt the “Class-combiner” strategy [13] that incorporate the forecasted class only.
1) Choice of semi-supervised learning algorithms

With the aim of attaining a balanced trade-off between accuracy and scalability, we find a discriminative model decision boundary within the nonatomic clusters [44] which is contingent upon the Support Vector Machine SVM algorithm [45]. The underlying reasons for adopting SVM algorithm as a fitted option are discussed below:

- Extensible to very huge date sets.
- Results are very accurate and can model complex non-linear decision boundaries.
- Fewer user-adjustable parameters.

Practically, the SVM algorithm first convert the training data to an up-scaled dimension. Secondly, the new up-scaled dimension is searched to find the hyperplane, the linear optimal decision boundary. In order to distinguish an object of a class among different classes with the highest margin, a hyperplane needs to be found out that uses support vector as well as margin. The objective function for finding the maximum marginal hyperplane in SVM is discussed below:

\[
\min_{w,b,\xi} = C \left( \sum_{i=1}^{n} \xi_i + \frac{1}{2} \|w\|^2 \right)
\]

\[
s.t \; y_i (w^T x_i + b) \geq 1 - \xi_i
\]

\[
\xi_i \geq 0, \; i = 1, \cdots, n
\]

Here \(x_i\) refers to the training observations of \(D_t\), while \(\xi_i\) represents labelled data as a flexible inconstant, while \(C\) indicates a persistent parameter for the purpose of trade-off amid the two core learning objectives of error reduction and attaining higher margin. In this approach, the SVM algorithm is adopted for the purpose of classifying as well as prediction, which requires to perform prior labelling to the training flows. Therefore, it is essential to include pseudo labelled flows with the labelled flows by amending the fundamental functions of the SVM algorithm as discussed below:

\[
\min_{w,b,\xi} = C \left( \sum_{i=1}^{n} \xi_i + \lambda_1 \sum_{j=1}^{m} \xi_j \right) + \frac{1}{2} \|w\|^2
\]

\[
s.t \; y_i (w^T x_i + b) \geq 1 - \xi_i \quad \text{(labelled data)}
\]

\[
\lambda_1 y_j (w^T x_j + b) \geq 1 - \xi_j \quad \text{(pseudo - labelled data)}
\]

\[
\xi_i \geq 0, \; i = 1, \cdots, n, \; \xi_j \geq 0, \; j = 1, \cdots, m
\]

where \(x_j\) represents the pseudo labelled flows \(D_{p2}\) \(D_{p2} \subseteq D_u\); \(\xi_j\) represents pseudo labelled flows as flexible inconstant, while \(\lambda_1 \in [0,1]\) is used to manage the trade-off between the labelled flows and pseudo labelled flows as a weighted parameter. As a result, the amended fundamental functions are still effective to prevent the plague of local maxima (minima).

Moreover, in order to handle the optimization issue of amended fundamental function efficiently, we incorporated the Gauss-Seidel/smoothing technique [46] to determine its dual. The repetitive procedure of the self-training employs the amended fundamental function in order to expand the scope of pseudo-labelled flows step by step. This step has in prospect to be instrumental for new data labelling process. For this reason, the labelled as well as the pseudo-labelled data in the fundamental functions are not re-weighted. Additionally, we introduced a hybrid fundamental function in order to find the optimal \(\lambda_1\) values in the course of local self-training.

E. META-LEARNING LAYER

With the aim to reduce the undesirable variation and prejudice by enhancing the classifier predictions in the model, we used an unvaried base learner algorithm instead of a solo base. A general solution for this purpose is to use meta-learning layer due to advantage of its iterative procedure, as well as to calculate the descriptive models in which the learning
schemes are illustrated by the resultant features produced earlier from semi-ensemble layer instead of original descriptive characteristics such as IP address features, port number etc. Furthermore, there are two steps involved in meta-learning layer. In first step, the meta-learner algorithm is trained via forecasting of the semi-supervised classifiers as well as the correct class of the actual traffic data. Furthermore, the second step involves testing the meta-classifier model which turns out the finalized forecasting outputs. Fig III-E illustrates the processing steps of meta-learning layer where the data in this is layer is divided into two subsets, training and testing. The first subset of training is used first to label the data in order to train the meta-classifier, which in turn is used for the Decision Tree [3]. Essentially being an interpretable algorithm, the tree organization results in high quality visual illustrations of the forecasts. As soon as the training objective is achieved, the testing set that consist of labeled data as well as unlabeled, is used on the meta classifier model in order to attain the concluding forecasts. Cross validations of ten folds is used by producing additional traffic flows in order to enhance the meta-classifier model standard.

IV. EXPERIMENTAL EVALUATION

In this section, experiments are conducted to show the effectiveness of the proposal using twelve data sets. For illustration, we compare the proposed solution against five baseline methods, including: Probabilistic graphical model (PGM) [47], Bipartite graph-based maximization(BGCM) [48] and Various-Widths Clustering ($k$NNVWC) [36]. Semi-Supervised for traffic labeling (ORSSTC) [49]. In following subsections, there are mainly four parts are presented: in subsection IV-B, we describe the twelve network and SCADA traffic data sets, in subsection ?? present the evaluation metrics. In subsection IV-A, we provide important discussion about the experimental setting. In subsection IV-D, we present and discuss the results of our approach and compare it with the baseline method since they have close relationship.

![FIGURE 5: The Meta-Learning Layer Overview](image)

**TABLE 1: The steps of selecting the most representative observations.**

**Input:**
Given the input labeled data $L$, set of unlabeled data $U$.
Set of the Semi-supervised to $S_{meta} = \emptyset$
Set the meta-data to $S_{meta} = \emptyset$

**Generate Meta Data:**
1. Generate $N$ bins from the randomised $L$.
   a. $L = L_1 + L_2 + L_N$.

2. For each fold $i \in N$, do the following.
   a. create the new set as.
      1. $L_{new} = L - L_i$
      2. $L_{un} = L$
   b. For each Semi-supervised $j \in I_j^{semi}$, do the following.
      1. Train $I_j^{semi}$ on $L_{new}$
      2. Test $I_j^{semi}$ on $L_{un}$
      3. $S_i^j$ = probability results of $I_j^{semi}$ on $L_{un}$
   c. Update the meta-data set of $S_{meta}$.
      $S_{meta} = S_{meta} + S_i^j$

3. Train the meta-classifier on $S_{meta}$

**A. TRAFFIC DATA SETS**

In this paper, we have used twelve traffic data sets to evaluate the effectiveness and the performance of the proposed Net-Stack approach. Note that our focus were on TCP flows due to the clear start-end information for TCP flows. Table 2 briefly provides the characteristic of the benchmark data set, including the number of features, classes and the percentage...
of training and testing set. Since limited descriptions of the datasets in this paper, the reader is advised to consult the original references [50]–[53] for more complete descriptions and details.

**TABLE 2: The benchmark data set**

| Data sets | Features | Classes | Training set | Testing set |
|-----------|----------|---------|--------------|-------------|
| MHRID     | 20       | 14      | 10500        | 3500        |
| MHORD     | 149      | 12      | 15750        | 5250        |
| SHIRD     | 41       | 2       | 7500         | 2500        |
| SHORD     | 20       | 6       | 10030        | 3100        |
| SPFDS     | 20       | 14      | 10500        | 3500        |
| DOSDS     | 41       | 2       | 7500         | 2500        |
| SPDOs     | 20       | 6       | 10030        | 3100        |
| WTP       | 20       | 14      | 10500        | 3500        |

Multi-class dataset

| ITD       | 149      | 12      | 15750        | 5250        |
| DARPA     | 41       | 2       | 45000        | 25000       |
| WIDE      | 20       | 6       | 10030        | 3100        |
| ISP       | 12       | 149     | 15750        | 5250        |

**Internet Traffic Data (ITD):** The network traffic traces are collected from the high-performance network monitor (described in [11]). These traces captured using its loss-limited while the fully-payload were captured from a research-facility which hosts up to 1,000 within different periods of time (for more details [11], [54]).

**DARPA data sets:** The DARPA data sets are widely used for IDS evaluation based on ML techniques since 1999 [10]. These data sets were prepared by tracking and analysing audit data for users on ARPANET. The characteristic of these raw data contains flow observations associated with a label as either normal or an attack.

**ISP data sets [12]:** These annotated data sets consists of 30k flows generated by a medium-sized Australian ISP network, and sampled from fourteen different types of traffic applications.

**B. EXPERIMENTAL SETTING**

In order to obtain a robust result and to avoid bias toward the data ordering, we have used cross-validation strategy as an experimental settings for the proposed Net-Stack approach and the baseline methods. We have applied 10-fold cross validation on randomized data set and repeated 5 times to capture corresponding results of the proposed Net-Stack approach and the baseline methods, including the overall accuracy, F-measure value, run-time and stability scores.

For the implementation, we have used Weka library’s implementation [55] on a 64-bit Windows-based system with 4-duo and Core(i7), 3.30 GHz Intel CPU, 8-GB of memory.

**C. PERFORMANCE METRICS**

In the following, we explain two evaluation metrics, namely, accuracy and F-measure through which we quantitatively evaluate the effectiveness of the proposed approach and the baseline methods:

**Algorithm 2: Experimental processing steps**

1. **Input:**
   - Counter = 5;
   - SemiMethods(ST) = [ORTSC, PGM, BGCM, kNNVWC, SemiTra, Net-Stack]
   - DS = {DARPA, WTP, ···, Dn};

2. **Output:**
   - Metrics = [Accuracy, F-measure, Runtime, Stability];

3. **foreach** SemiMethods in [1, ST] do
   4. **foreach** ds ∈ DS do
      5. **foreach** c ∈ [1, Counter] do
         6. shuffle the observation-order for ds;
         7. produce B bins from the shuffled ds;
         8. **foreach** fold ∈ [1, B] do
            9. LabelledDS = bin[fold];
            10. UnlabelledDS = ds − LabelledDS;
            11. TrainDS = LabelledDS + Subset from UnlabelledDS;
            12. TestDS = UnlabelledDS − TrainDS;
            13. STi ∈ SemiMethods do
               14. STModel = learning(TrainDS, STi);
               15. Resultsi = testing(TestDS, STModel);
               16. Results = Results ∪ Resultsi;
      17. **Metrics** = average(Results);

**TABLE 3: Confusion Matrix**

| Actual label | Estimated label |
|--------------|-----------------|
| Normal flow  |                 |
| Attack flow  |                 |
| Normal flow  | True Negative (TN) | False Positive (FP) |
| Attack flow  | False Negative (FN) | True Positive (TP) |

- **Overall accuracy:** discovers one-to-one relationship between assigned/classified label by the proposed solution according to the actual label. We compute it by summing up well classified observations and then dividing it by the total observations number. For illustration, Table 3 used to defined overall accuracy (CC):

\[
    \text{Accuracy} = \frac{TP + TN}{T}
\]

where T is the total number of predictions made for a data set.

- **F-measure:** to avoid misleading with imbalanced data sets, F-measure has also been employed because this metric is harmonic mean of precision and recall, and therefore it is an appropriate to take both false positives and false negatives into account and compute the weighted average of the precision and recall rates, which is defined as follows:

\[
    F - \text{measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}
\]

where Precision quantifies the number of correct classified observations, and Recall quantifies the number
of correct positive observations among the correctly identified observations.

The proposed Net-Stack approach and the baseline methods may provide different predicted labels for specific observations across different runs. Thus, the stability of the Net-Stack approach and the baseline methods need to be measured using the pairwise stability index $SI_m$, as follows:

$$SI_m = \frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} SI_r(Y_i, Y_j).$$  \hspace{1cm} (10)

where

$$SI_r(Y_i, Y_j) = \begin{cases} 
1 & \text{if } Y_i(x_i) = Y_j(x_j) \\
0 & \text{otherwise}
\end{cases} \hspace{1cm} (11)$$

As defined above, $SI_m(C)$ computes the average stability score from different runs over the final predicted class labels. The scores are taken from zero to one. The value one is obtained if the prediction results are identical while the zero value indicates that the prediction results between $Y_i$ and $Y_j$ are totally different.

D. EXPERIMENT RESULTS AND DISCUSSION

Here we would like to present the results of our proposed Net-Stack approach and compare it against five baseline methods. Three salient comparisons of this evaluation are taken into account: accuracy, running time taken by each method and the stability.
1) Accuracy Performance

Fig 6 demonstrates the results of the proposed Net-Stack approach and the baseline methods (PGM, BGCM, ORTSC, kNNVWC and SemTra). We use accuracy and F-Measure metrics to evaluate the performance of these methods. As depicted in Fig 6, the proposed Net-Stack approach and the two baseline methods have achieved remarkable performance with respect to the overall accuracy on eight traffic data sets, including: SPFDS, MHIRD, MHORD, SHORD, DOSDS, WTP, SPDOS and SHIRD. For instance, the Net-Stack approach demonstrates the highest average accuracy scores of about 96%. As shown, the SemTra takes the second rank with an average of 94.76%. The accuracy of kNNVWC, PGM, BGCM and ORTSC rank are 92.98%, 92.53%, 86.89% and 85.075% on average, respectively. Extensive experiments against baseline methods demonstrate the effectiveness and robustness of the proposed Net-Stack approach to predict the accurate class label (whether normal or attack) on all the binary class data sets. Also, we can see some baseline methods show promising accuracy results on average on all the binary class data sets. Similarly, the performance of the proposed approach with F-measure metric against the baseline methods are consistent and are very close to the scores of accuracy metric. Clearly, it can be observed from Fig 6 that the proposed Net-Stack approach and the two baseline methods also have achieved remarkable performance, whereas the average F-measure scores of Net-Stack, SemTra, kNNVWC, PGM, BGCM and ORTSC are 94.06%, 91.03%, 86.60%, 84.18%, 83.37% and 81.3% respectively.

On the other hand, as shown in Fig 7, baseline methods performs poorly on ITD and DARPA. This can be attributed to the fact that the baseline methods are intended to be only used with binary class traffic data, while ITD and DARPA consist of different applications (e.g. FTP, WWW, MAIL, P2P etc.). In particular, the proposed Net-Stack approach achieves 97.60% on average, while we can observe that the SemTra method achieves the second rank with 94.76% on average. This is because our proposed approach firstly integrates various-widths clustering that produces micro-clusters that may have a small intra-cluster variance. In addition, it adapts ensemble learning to build collaborative decision combines accurately label observations and discard unsupported ones. We also compare and evaluate the performance of the Net-Stack approach with the baseline methods using F-measure metric. Fig 7 shows that the proposed Net-Stack approach achieves 94.07% on average on all multi-class traffic datasets (ITD, DARPA, WIDE and ISP), while baseline methods show not promising results.

2) Running times

Here, we discuss the comparison of the Net-Stack approach’s runtime performance against each distinct semi-supervised method. To obtain confident results, in the experiment, we conducted runtime experiments 10 times and the average value is reported. The runtime measurements for the six semi-supervised methods are displayed in Table 4. The PGM, BGCM and ORTSC are significantly faster than the proposed Net-Stack approach. Runtime of ORTSC on all twelve datasets is steadily faster than all other semi-supervised methods. Clearly, the ORTSC method is the fastest algorithm among the five methods. As shown, this method is faster than PGM with average time of 2.92 seconds on all data sets. Comparing to the proposed approach Net-Stack, ORTSC is faster with average time of 60.57 seconds. The reduction of
TABLE 4: Runtime performances for labelling process on different datasets

| Dataset | ORTSC | PGM  | BGCM | kNNVWC | SemTra | Net-Stack |
|---------|-------|------|------|--------|--------|-----------|
| SPFDS   | 0.30  | 0.92 | 1.72 | 14.71  | 36.49  | 8.34      |
| MHORD   | 0.6   | 1.23 | 1.94 | 30.10  | 78.81  | 16.91     |
| DOSOS   | 0.24  | 0.69 | 1.29 | 25.56  | 67.41  | 14.22     |
| MHIRD   | 0.40  | 1.28 | 1.93 | 30.43  | 80.03  | 17.08     |
| SHORD   | 0.53  | 1.42 | 2.35 | 34.64  | 90.78  | 19.45     |
| SHIRD   | 0.38  | 1.27 | 1.77 | 27.08  | 70.77  | 15.24     |
| WTP     | 0.10  | 0.13 | 1.93 | 1.89   | 2.71   | 1.11      |
| SPDOS   | 0.55  | 1.48 | 3.18 | 43.09  | 111.44 | 24.12     |
| ITD     | 0.87  | 2.39 | 166.08 | 381.63 | 824.9  | 86.25     |
| DARPA   | 10.33 | 28.12 | 591.4 | 879.88 | 1349.69 | 325.5     |
| WIDE    | 0.28  | 0.86 | 1.77 | 33.75  | 37.49  | 30.48     |
| ISP     | 0.39  | 1.25 | 1.9  | 71.48  | 78.85  | 64.11     |

FIGURE 8: The stability comparison of the Net-Stack approach and the baseline methods on binary-class traffic datasets
the execution time of the proposed Net-Stack approach can be achieved by parallel computing techniques.

3) Stability
The stability metric is important to ensure that a method is stable irrespective of different runs. Therefore, we would like to evaluate the stability performance of the proposed Net-Stack approach in accurately predicting class labels of unlabelled network traffic observations under different runs. The stability value for each method is calculated by Equation 10, and therefore we evaluate the stability performance of each method on all datasets (see Table 2). For confidence, we run 100 trails for each method on each data set. The network traffic data is divided into 30% and 70% as testing data and training data, respectively. The training data set was randomly sampled to execute each trial.

Fig 8 depicts the results for comparing the stability performance of our proposed Net-Stack approach against the baseline methods on binary-class traffic datasets. The results indicate that the proposed approach shows significant results with margin between 3.08% and 13.56%. The rationale behind this achievement is the fact that the Net-Stack approach integrates various machine learning techniques, which provides robustness in accurately predicting class labels. Notably, the ORTSC method demonstrates the worst stability scores on all traffic datasets, due to the characteristics of K-means which has a local optima problem and is sensitive to the selection of the initial cluster prototypes. Similarly, BGCM performs poorly on ITD and DARPA traffic datasets. Fig 9 shows the results of the stability performance on multi-class traffic datasets. As illustrated, the the proposed approach retains its level of stability scores on multi-class traffic datasets against
the baseline methods and demonstrates a highly consistent stability with all datasets. As expected, ORTSC method exhibits the worst stability performance in this evaluation.

V. CONCLUSION

In this research paper we have proposed an efficient network traffic classification, namely Net-Stack, which utilizes the characteristics of various-widths clustering, multiview representations and the advantages of semi-supervised stacking to attain accurate and reliable labelled data for effective training with low variance and bias of the predictive classifier model. The Net-Stack consist of four stages including: (i) selecting most representative observations from the local and global prospective based on various-widths clustering, (ii) building multiview representations of the original traffic data using dimensional reduction techniques, (iii) refining the decision boundaries for each traffic class and getting a low dimensional metadata representation by exploiting the complementary information contained in multiple views using heterogeneous semi-supervised learning algorithms, and (iv) comprehensively improving the generalization performance and predicting the final classification decision by applying a meta-classifier to the metadata representation obtained in the third stage. The experimental results on publicly available traffic dataset benchmark showed the strength of the proposed approach in the network traffic classification task in comparison to the state-of-art semi-supervised learning approaches. For future work, the reduction of the execution time of the proposed Net-Stack approach can be achieved by parallel computing techniques using multi-core CPUs or Graphics Processing Units (GPU).

The reduction of the execution time of the proposed Net-Stack approach can be achieved by parallel computing techniques.

REFERENCES

[1] Thomas Karagiannis, Konstantina Papagiannaki, and Michalis Faloutsos. Blinc: multitelevel traffic classification in the dark. In ACM SIGCOMM Computer Communication Review, volume 35, pages 229–240. ACM, 2005.

[2] Andrew W Moore and Konstantina Papagiannaki. Toward the accurate identification of network applications. In Passive and Active Network Measurement, pages 41–54. Springer, 2005.

[3] Thomas Karagiannis, Andre Broido, Michalis Faloutsos, et al. Transport layer identification of p2p traffic. In Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, pages 121–134. ACM, 2004.

[4] Laurent Bernaille, Renata Teixeira, Ismael Akodkenou, Augustin Soule, and Kave Salamatian. Traffic classification on the fly. ACM SIGCOMM Computer Communication Review, 36(2):23–26, 2006.

[5] Yu Wang, Yang Xiang, Jun Zhang, Wanlei Zhou, Guiyi Wei, and Laurence T Yang. Internet traffic classification using constrained clustering. IEEE Transactions on Parallel and Distributed Systems, page 1, 2013.

[6] Thuy TT Nguyen and Grenville Armitage. A survey of techniques for internet traffic classification using machine learning. Communications Surveys & Tutorials, IEEE, 10(4):56–76, 2008.

[7] Jeffrey Erman, Anirban Mahanti, Martin Arlitt, Ira Cohen, and Carey Williamson. Offline/realtime traffic classification using semi-supervised learning. Performance Evaluation, 64(9):1194–1213, 2007.

[8] Charalampos Rotos, Jurgen Van Gael, Andrew W Moore, and Zoubin Ghahramani. Probabilistic graphical models for semi-supervised traffic classification. In Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, pages 752–757. ACM, 2010.

[9] Adil Fahad, Karayman Alharthi, Zahir Tari, Abdulmohsen Almalwi, and Ibrahim Khalil. Chclus: Hybrid clustering-classification approach for accurate and efficient network classification. In 39th Annual IEEE Conference on Local Computer Networks, pages 168–176. IEEE, 2014.

[10] Salvatore J Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Philip K Chan. Cost-based modeling for fraud and intrusion detection: Results from the jam project. In DARPA Information Survivability Conference and Exposition, 2000. DISCEX’00. Proceedings, volume 2, pages 130–144. IEEE, 2000.

[11] A. Moore, J. Hall, C. Kreibich, E. Harris, and I. Pratt. Architecture of a network monitor. In Passive & Active Measurement Workshop 2003 (PAM2003), Citeseer, 2003.

[12] Yu Wang, Yang Xiang, Jun Zhang, and Shun-Zheng Yu. A novel semi-supervised approach for network traffic clustering. In Network and System Security (NSS), 2011 5th International Conference on, pages 169–175. IEEE, 2011.

[13] Mawi working group traffic archive. [ONLINE]. Available: http://mawi.wide.ad.jp/mawi/.

[14] Thomas Karagiannis, Andre Broido, Michalis Faloutsos, and KC Claffy. Transport layer identification of p2p traffic. In Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, pages 121–134, 2004.

[15] Yu Wang, Yang Xiang, and Shun-Zheng Yu. Automatic application signature construction from unknown traffic. In 2010 24th IEEE International Conference on Advanced Information Networking and Applications, pages 1115–1120. IEEE, 2010.

[16] Jingjing Zhao, Xuyang Jing, Zheng Yan, and Witold Pedrycz. Network traffic classification for data fusion: A survey. Information Fusion, 72:22–47, 2021.

[17] Fanniia Pacheco, Ernesto Exposito, Mathieu Gineste, Cedric Baudoin, and Jose Aguilar. Towards the deployment of machine learning solutions in network traffic classification: A systematic survey. IEEE Communications Surveys & Tutorials, 21(2):1988–2014, 2018.

[18] Guanglu Sun, Teng Chen, Yangyang Su, and Chenglong Li. Internet traffic classification based on incremental support vector machines. Mobile Networks and Applications, 23(4):789–796, 2018.

[19] Martin Andreoni Lopez, Diogo MF Mattos, Otto Carlos MB Duarte, and Guy Pujolle. A fast unsupervised preprocessing method for network monitoring. Annals of Telecommunications, 74(3):139–155, 2019.

[20] Muhammad Shafiq, Xiangzhan Yu, Ali Kashif Bashir, Hassan Nazer Chaudhry, and Dawei Wang. A machine learning approach for feature selection traffic classification using security analysis. The Journal of Supercomputing, 74(10):4867–4892, 2018.

[21] Da Tong, Yun Rock Qu, and Viktor K Prasanna. Accelerating decision tree based traffic classification on fpga and multicore platforms. IEEE transactions on parallel and distributed systems, 28(11):3046–3059, 2017.

[22] Xiao Wang, Ying Liu, and Wei Su. Real-time classification method of network traffic based on parallelized cnn. In 2019 IEEE International Conference on Power, Intelligent Computing and Systems (ICPCS), pages 92–97. IEEE, 2019.

[23] Shasha Zhao, Yi Xiao, Yueqiang Ning, Yuxiao Zhou, and Dengying Zhang. An optimized k-means clustering for improving accuracy in traffic classification. Wireless Personal Communications, pages 1–13, 2021.

[24] Jun Zhang, Yang Xiang, Wanlei Zhou, and Yu Wang. Unsupervised traffic classification using flow statistical properties and ip packet payload. Journal of Computer and System Sciences, 79(5):573–585, 2013.

[25] Jonas Hochst, Lars Baumgärtner, Matthias Hollick, and Bernd Freisleben. Unsupervised traffic flow classification using a neural autoencoder. In 2017 IEEE 42Nd Conference on local computer networks (LCN), pages 532–536. IEEE, 2017.

[26] Victor Labayen, Eduardo Magaña, Daniel Morató, and Mikel Izal. Online classification of user activities using machine learning on network traffic. Computer Networks, 181:107557, 2020.

[27] Lucio Henrik A Reis, Luiz Claudio S Magalhães, Dianne Scherly V de Medeiros, and Diogo MF Mattos. An unsupervised approach to infer quality of service for large-scale wireless networking. Journal of Network and Systems Management, 28(4):1228–1247, 2020.

[28] Ehsan Mahdavi, Ali Fanian, and Homa Hassannejad. Encrypted traffic classification using statistical features. ISeCure, 10(1), 2018.

[29] Jing Ran, Xiaocheng Kong, Gan Lin, Dongming Yuan, and Hefei Hu. A self-adaptive network traffic classification system with unknown flow.
detection. In 2017 3rd IEEE International Conference on Computer and Communications (ICCC), pages 1215–1220. IEEE, 2017.

[30] Jun Zhang, Xiao Chen, Yang Xiang, Waneli Zhou, and Jie Wu. Robust network traffic classification. IEEE/ACM transactions on networking, 23(4):1257–1270, 2014.

[31] Thijs van Ede, Riccardo Bortolameotti, Andrea Continella, Jingjing Ren, Daniel J Dubois, Martina Lindorfer, David Choffnes, Maarten van Steen, and Andreas Peter. Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic. In Network and Distributed System Security Symposium, NDSS 2020. Internet Society, 2020.

[32] Auwal Sani Biydyu and Huifang Deng. Semi-supervised encrypted traffic classification with deep convolutional generative adversarial networks. IEEE Access, 8:118–126, 2019.

[33] Fakhroddin Noorbehbahani and Sadeq 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), pages 120–125. IEEE, 2018.

[34] Ireneusz Czarnowski and Piotr Jdrzejowicz. An approach to instance reduction in supervised learning. In International Conference on Innovative Techniques and Applications of Artificial Intelligence, pages 267–280. Springer, 2003.

[35] Bartosz Krawczyk, Isaac Triguero, Salvador García, Michal Woźniak, and Francisco Herrera. Instance reduction for one-class classification. Knowledge and Information Systems, 59(3):601–628, 2019.

[36] Abdul Mohsen Almalawi, Adil Fahad, Zahir Tari, Muhammad Aamir Cheema, and Ibrahim Khalil. k nnwvc: An efficient k-nearest neighbors approach based on various-widths clustering. IEEE Transactions on Knowledge and Data Engineering, 28(1):68–81, 2015.

[37] Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning. arXiv preprint arXiv:1304.5634, 2013.

[38] Xijiong Xie and Shiliang Sun. Multi-view laplacian twin support vector machines. Applied intelligence, 41(4):1059–1068, 2014.

[39] Chang-Dong Wang, Jian-Huan Lai, and S Y Philip. Multi-view clustering based on belief propagation. IEEE Transactions on Knowledge and Data Engineering, 28(4):1007–1021, 2015.

[40] Li Teng, Hongyu Li, Xuping Fu, Wenbin Chen, and I-Fan Shen. Dimension reduction of microarray data based on local tangent space alignment. In Fourth IEEE Conference on Cognitive Informatics, 2005.(ICI CI 2005), pages 154–159. IEEE, 2005.

[41] Jean Bourgain. On lipschitz embedding of finite metric spaces in hilbert space. Israel Journal of Mathematics, 52(1-2):46–52, 1985.

[42] John Shawe-Taylor, Nello Cristianini, et al. Kernel methods for pattern analysis. Cambridge university press, 2004.

[43] Michael Barbehenn. A note on the complexity of dijkstra’s algorithm for graphs with weighted vertices. IEEE transactions on computers, 47(2):263, 1998.

[44] Brijesh Verma and Asifahqar Rahman. Cluster-oriented ensemble classifier: Impact of multicluster characterization on ensemble classifier learning. IEEE Transactions on knowledge and data engineering, 24(4):605–619, 2013.

[45] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, and Asdrubel Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. Neurocomputing, 408:189–215, 2020.

[46] Zhen Jiang, Shiyoung Zhang, and Jianping Zeng. A hybrid generative/discriminative method for semi-supervised classification. Knowledge-Based Systems, 37:137–145, 2013.

[47] Charalampos Rotsos, Jurgen Van Gael, Andrew W Moore, and Zoubin Ghahramani. Probabilistic graphical models for semi-supervised traffic classification. In Proceedings of the 6th International workshop on large-scale data classification and mobile computing conference, pages 752–757, 2011.

[48] Jing Gao, Feng Liang, Wei Fan, Yizhou Sun, and Jiawei Han. A graph-based consensus maximization approach for combining multiple supervised and unsupervised models. IEEE Transactions on Knowledge and Data Engineering, 25(1):15–28, 2011.

[49] Sebastian Zander, Thuy Nguyen, and Grenville Armitage. Automated traffic classification and application identification using machine learning. In Local Computer Networks, 2005. 30th Anniversary. The IEEE Conference on, pages 250–257. IEEE, 2005.

[50] Shian Suthaharan, Mohammed Alzhrani, Sutharan-Rajasegarar, Christoph Leckie, and Marimuthu Palaniwami. Labelled data collection for anomaly detection in wireless sensor networks. In Proceedings of the 6th International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), pages 269–274. IEEE, 2010.

[51] S.J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P.K. Chan. Cost-based modeling for fraud and intrusion detection: Results from the jam project. In Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX), pages 130–144. IEEE, 2000.

[52] Abdulmohsen Almalawi, Zahir Tari, Ibrahim Khalil, and Adil Fahad. SCADAVT-A framework for SCADA security testbed based on virtualization technology. In Proceedings of the 38th IEEE Conference on Local Computer Networks (LCN), pages 639–646. IEEE, 2013.

[53] A. Moore, J. Hall, C. Kreibich, E. Harris, and I. Pratt. Architecture of a network monitor. In Proceedings Passive & Active Measurement Workshop (PAM), 2003.

[54] Adil Fahad, Zahir Tari, Ibrahim Khalil, Abdulmohsen Almalawi, and Albert Y Zomaya. An optimal and stable feature selection approach for traffic classification based on multi-criterion fusion. Future Generation Computer Systems, 36:156–169, 2014.

[55] Ian H Witten, Eibe Frank, Mark A Hall, and Geoffrey Holmes. Weka: Practical machine learning tools and techniques with java implementations. 1999.