

Comparison of machine-learning algorithms for classification of VPN network traffic flow using time-related features

Sikha Bagui, Xingang Fang, Ezhil Kalaimannan, Subhash C. Bagui & Joseph Sheehan

To cite this article: Sikha Bagui, Xingang Fang, Ezhil Kalaimannan, Subhash C. Bagui & Joseph Sheehan (2017) Comparison of machine-learning algorithms for classification of VPN network traffic flow using time-related features, Journal of Cyber Security Technology, 1:2, 108-126, DOI: [10.1080/23742917.2017.1321891](https://doi.org/10.1080/23742917.2017.1321891)

To link to this article: <https://doi.org/10.1080/23742917.2017.1321891>



Published online: 01 Jun 2017.



Submit your article to this journal [↗](#)



Article views: 1391



View related articles [↗](#)







View Crossmark data [↗](#)



Citing articles: 4 View citing articles [↗](#)



Comparison of machine-learning algorithms for classification of VPN network traffic flow using time-related features

Sikha Bagui , Xingang Fang , Ezhil Kalaimannan , Subhash C. Bagui 
and Joseph Sheehan

Department of Computer Science, University of West Florida, Pensacola, FL, USA

ABSTRACT

Network traffic classification and characterisation is playing an increasingly vital role in understanding and solving security-related issues in internet-based applications. The priority of research studies in this area has focused on characterisation of network traffic based on various layers of communication protocols as outlined in the TCP/IP stack and even further expanded to concentrate on specific application-layer protocols. Virtual Private Networks (VPNs) have become one of the most popular remote access communication methods among users over the public internet and other Internet Protocol (IP)-based networks. VPNs are governed by IP Security, which is a suite of protocols used for tunnelling the already encrypted IP traffic, to guarantee secure remote access to servers. In this paper, we propose and develop a framework to classify VPN or non-VPN network traffic using time-related features. Our focus is on classification of network traffic which is encrypted, tunnelled through a VPN, and the one which is normally encrypted (non-VPN transmission), using machine-learning techniques on data sets of time-related features. Six classification models: logistic regression, support vector machine, Naïve Bayes, *k*-nearest neighbour and ensemble methods – the Random Forest (RF) classifier and Gradient Boosting Tree (GBT) classifiers – are compared, and recommendations of optimised RF and GBT models over other models are provided in terms of high accuracy and low overfitting. Features which contributed to achieve 90% accuracy in each category were also identified.

ARTICLE HISTORY

Received 2 December 2016
Accepted 19 April 2017

KEYWORDS

VPN; supervised machine learning; network traffic flow classification; random forest; gradient boosting tree

1. Introduction

The continuous expansion of the internet and ever-growing dynamic nature of mobile and wireless networks has made the classification of network traffic a significant challenge to experts [1]. Recent years have seen much effort focused on characterisation of network traffic based on various layers of communication protocols as outlined in the Transmission Control Protocol/Internet Protocol (TCP/IP) stack. The focus has expanded to concentrating on specific application-layer protocols which are dedicated to using popular web-based applications.

A key cryptographic feature of network traffic is encryption, which is becoming more pervasive on the internet. Classification of IP traffic can be defined based on various functions such as encryption of traffic (using HTTPS) during communication from source to destination based on the IP address, protocol encapsulation using IP Security (IPSec) protocol (or VPN tunnelling), application-specific port numbers (HTTP, FTP or SSH) or specific applications such as Skype, Facebook, Gmail etc. Some of these real-time applications are complex since they use multiple services, which leads to identification of the application and the specific task associated with it.

Virtual Private Network (VPN) technologies, which are constantly evolving, have now become the preferred network access medium for routing internet traffic between two end-points connected together over the internet (public network). The most important feature of VPN is the tunnelling of already encrypted IP traffic, to guarantee secure remote access to servers. The IPSec protocol, which governs the VPN tunnelling mechanism, maintains packet-level encryption, hence making it almost impossible to identify the application running through end-points of the tunnel.

Plethora of prior research has been carried out on internet traffic classification and characterisation both in real time and non-real time. Most of these studies utilised statistical and machine-learning techniques to solve problems found in practice. However, these studies were limited in focus. Most of these earlier studies were related to specific types of applications or devices facilitating network communication. In light of the absence of any existing work on specific network applications, we propose and develop a framework to classify VPN or non-VPN network traffic using time-related features.

In this paper, our focus is on classification of encrypted network traffic, tunnelled through a VPN, and traffic which is encrypted normally (non-VPN transmission). This problem is really challenging, since one cannot make this classification by just analysing the corresponding IP packet header. By referring to encrypted VPN network traffic, we relate to all the three implementations or protocols which are IPSec, Point-to-Point Tunnelling Protocol and Secure Socket Layer. The difference between VPN and non-VPN traffic is the privacy of the sender and receiver involved in the internet communication. When VPN traffic is intercepted, the attacker will not be able to identify the legitimate IP address of the end-points of the tunnel, that is, the source or destination.

Our work compares supervised machine-learning techniques for the classification of VPN versus non-VPN traffic. The machine-learning models compared are logistic regression (LR), support vector machine (SVM), Naïve Bayes (NB), k -nearest neighbour (KNN), Gradient Boosting Trees (GBTs) and Random Forest (RF).

The rest of the paper is organised as follows. [Section 2](#) provides a brief literature review of prior works on classification and characterisation of internet traffic. [Section 3](#) presents the data analysis. [Section 4](#) presents the results and discussion and [Section 5](#) presents the conclusions and future work.

2. Related literature

Due to the ever-changing nature of internet technologies, network traffic classification is a constantly revisited topic. There are a few popular approaches to internet traffic classification. A basic method of classification is via port analysis. A method that classifies

traffic by analysing registered port numbers, well-known port numbers, reserved port numbers and ephemeral port numbers is described [2]. While the port analysis classification method performs well, it is not practical. Since port usage is unregulated (applications have a lot of freedom in terms of the ports they use), it cannot reliably classify network traffic.

Research studies on network traffic classification and effectiveness measures were introduced in early 1990s, where Paxson [3] described his work related to how internet traffic evolved and how existing users found multiple ways to incorporate the internet into their work patterns. Paxson [4] also presented analytical models describing random variables associated with specific internet applications like telnet, nntp, smtp and ftp connections. In this work, statistical features such as packet length, flow duration, inter-arrival times were used to construct and prove that simple analytical models are a good approximation to measure effectiveness among various models based on specific network applications.

A novel methodology to detect application protocols using the first five packets of a stateful TCP connection was developed by Bernaille et al. [5]. This work defines a two-stage approach called the learning phase and online traffic classification phase to cluster TCP flows that share a common behaviour and to determine the application associated with each identified TCP flow, respectively.

Park et al. [6] proposed a similar methodology to classify internet traffic according to specific types of associated applications in order to achieve scalable QoS provisioning. The authors of this paper use feature selection on the internet traffic obtained from ISPs and perform feature reduction to balance the performance and complexity. The focus on these research works was to achieve higher efficiency and performance over the traditional port-based classification of the internet traffic.

Few interesting works employed machine-learning tools and techniques to identify and classify applications based on internet traffic flow and patterns. Erman et al. [7] developed a clustering-based machine-learning framework for classifying network traffic using only unidirectional flow statistics. This paper found that better classification performance is achieved when statistics for the server-to-client direction is used versus when statistics for the client-to-server direction is used. Williams et al. [8] presented an alternative way to using machine-learning algorithms such as NB. The Bayesian Network was used to identify the application based on per-flow statistics derived from payload-independent features such as packet length and inter-arrival time distributions.

A real-time IP traffic classification using the NB optimisation machine-learning algorithm was proposed by Nguyen and Armitage [9]. The authors showed that the classification of IP traffic is possible, even when a flow's beginning is lost or initiated mid-way through a flow. Crotti et al. [10] proposed a new methodology called 'protocol fingerprinting' to develop an algorithm for the classification of IP traffic. This method was based on three simple properties of the captured IP packets: size, inter-arrival time and arrival order. Bonfiglio et al. [11] presented an approach to classify Skype traffic in real time using two complementary techniques which were based on the Bayes Classifier and deep packet inspection. While these methods were widely adopted by many works in the literature for classifying IP-based real-time applications, this research idea of leveraging on randomness to reveal traffic flow was novel.

In our work, we employed supervised machine-learning methods to classify a network flow as either VPN encrypted or non-VPN encrypted. Supervised learning uses labelled training data to predict the class label on the unlabelled data. The classification models were trained on the data set with time-related features as independent variables.

3. Analysing the data

In this section, we discuss the data sets, data pre-processing that was performed, and the generation of the training and test data. We also go on to discuss the machine-learning models that were used and model selection and optimisation processes.

3.1. Data sets

The time-related feature VPN/non-VPN data set used in this work is based on the data sets from the ISCX Research Center of University of New Brunswick, Canada [12]. In this paper, the multi-label classification models were trained to classify network traffic flows into seven application categories including browsing, chat, file transfer, email, peer-to-peer, media streaming and voice over IP. Eight data sets of four different flow timeouts (15, 30, 60 and 120 s) on network flows, both with and without VPN encryption, were generated. In each data set, 24 features were generated in 8 time-related feature families, as shown in Table 1. The seven application categories were the class labels in these data sets.

3.2. Data pre-processing

Since our goal was to perform binary classification of VPN and non-VPN traffic, the eight data sets were reorganised into data sets of seven categories. Data cleaning and normalisation was performed to improve the performance of the model. The details are described below in the following breakdowns: data set reorganisation, data cleaning and normalisation.

3.2.1. Data reorganisation

Since the pattern of time-related features may be very different in different categories, the binary classification of VPN and non-VPN network traffic is performed separately on the data sets of the seven application categories. We also combined records with the

Table 1. Time-related features.

Feature	Description
Duration	The duration of the flow
Fiat	Forward Inter Arrival Time, the time between two packets sent forward direction (mean, min, max, std)
Biat	Backward Inter Arrival Time, the time between two packets sent backwards (mean, min, max, std)
Flowiat	Flow Inter Arrival Time, the time between two packets sent in either direction (mean, min, max, std)
Active	The amount of time, the time a flow was active before going idle (mean, min, max, std)
Idle	The amount of time, the time a flow was idle before becoming active (mean, min, max, std)
fb_psec	Flow bytes per second
fp_psec	Flow packets per second
Timeout	Flow timeout

same timeouts. Thus, the eight data sets (four different timeouts, and both VPN and non-VPN data sets for each timeout) were combined. The resulting data set was partitioned by the seven application categories into seven data sets. In each data set, the VPN/non-VPN label was employed as the new class label, and the 24 time-related features plus the timeout served as the new 25 time-related variables.

3.2.2. Data cleaning

A quick check of the data sets indicated the presence of empty records and duplicate records in each data set. Empty records and duplicate records were removed. A total of 495 empty and 1289 duplicate records were removed from all the 7 data sets.

3.2.3. Normalisation

Normalisation is a process that is generally applied to numerical data when the range of raw data varies widely. This variance could reduce the efficiency of many optimisation algorithms used in machine learning. There are many normalisation methods including standard score, min–max normalisation etc. Standard score (z-score) is a commonly used normalisation method that transforms the data such that the new feature vector has a zero mean and unit standard deviation. Standard score works particularly well with normally distributed data and will accelerate convergence of many optimisation algorithms. Min–max normalisation scales the features to the range between 0 and 1. This works well in the scenarios where positive values are preferred or required, for example, non-negative matrix factorisation and RGB image data scaling. Among our candidate models, none requires positive features and several will benefit from accelerated optimisation algorithms; hence, for this work, we applied the z-score normalisation [13] on all data sets:

$$z = \frac{(x - \mu)}{\sigma}$$

where z is standard score, x is raw datum, μ is mean and σ is standard deviation.

3.2.4. Generating training and test data

After pre-processing, we observed a nearly balanced amount of VPN and non-VPN records in each category, which is preferred to avoid class bias during the learning process. For each category, about 80% of the records were randomly partitioned into the training set and the rest were partitioned into the test set. The training set was employed in model selection and hyper-parameter optimisation. The test data sets were employed in the final test step to ensure that the model works on unknown data (the test data set was never seen by the model during model selection/optimisation process). By comparing the predictive performance of the trained model on separate test data, problems such as overfitting or underfitting can be identified. The partition was performed in a strategic fashion such that the VPN and non-VPN records were almost evenly distributed in the test and training data sets, as shown in Table 2.

Given that there were a total of 25 features, the observation to variable ratio in our data sets was at least 72 (the streaming data set), which is greater than the minimum sample amount suggested in many works. The potential under-fitting problem, due to

Table 2. Training and test sets of the various categories.

Traffic category	Sample distribution					
	Training		Testing		Total	
	VPN	Non-VPN	VPN	Non-VPN	VPN	Non-VPN
BROWSING	8000	7733	2000	1934	10,000	9667
CHAT	2135	1972	534	493	2669	2465
FT	3582	2647	896	662	4478	3309
MAIL	1056	938	264	235	1320	1173
P2P	3140	1985	785	497	3925	2482
STREAMING	917	890	229	223	1146	1113
VOIP	5145	4438	1287	1109	6432	5547

not enough records, can be monitored by checking the prediction accuracy score of the trained model on the test data set later.

3.3. Supervised machine-learning models

Our work is a binary classification problem. Classification is also known as supervised machine learning. In machine learning, the identification of suitable algorithms is highly data set dependent and hard to predict. Among available supervised machine-learning models, six models were compared in our work to evaluate the predictive performance of the classification of VPN and non-VPN network flow. The choice of candidate models covers linear decision boundary models such as LR, radius basis function kernel SVM, probabilistic models such as the NB model, neighbour-based models such as KNN, ensemble decision tree models such as RF and GBT. With the broad coverage of different types of classification models, we try to ensure the identification of the right classification model for our data sets.

LR [14] is a model that converts linear regression results of input variables to probabilities using a logit function to predict the predefined class labels, say $y = 0$ or 1 . LR is widely used as either an independent classifier or part of other models such as deep learning. It is one of the simplest and fastest models and performs well on data sets with linear separation boundaries. Unless the decision boundary is already known to be non-linear, it is usually the first model to try.

SVM [15] identifies the hyper-plane with maximum margin that separates the high dimensional space of input variables belonging to different classes. Although SVM with a linear kernel has a similar linear decision boundary to LR, SVM is capable of performing non-linear classification with extension of kernel functions that maps the input to other high-dimensional feature spaces for classification. The radial basis function (RBF) kernel is one of the fastest and mostly used kernels of SVM with known success on many data sets.

NB [16] methods are a family of probabilistic algorithms based on the Bayes theorem, with 'naïve' assumptions that attributes (features) are conditionally independent. Applications of the NB models are limited by possible poor performance on data sets that violate that assumption. Out of the many related algorithms in this family, the Gaussian Naïve Bayes algorithm was employed in this work.

KNN [17] classification model is a 'lazy' algorithm which skips the training step and performs classification by using majority votes of its KNN. k and the distance measure are the two key parameters of the KNN model.

Ensemble methods [18] are a family of techniques that combine base models to provide a meta-model with improved performance in terms of generalisability, accuracy and robustness over a single model. The widely used decision tree-based ensemble methods have exhibited the capability to capture extremely complicated non-linear patterns and provide excellent performance and robustness on structured data sets. RF [19] classifier and GBT [20] classifiers are two widely used decision tree-based ensemble methods which have had great success in many supervised machine-learning applications. They differ in the ways that subtrees are grown and how the results are ensembled. RF grows random decision trees based on a random subset of data and subset of features, while GBT grows decision trees one after another guided by a mechanism called ‘boosting’. RF models are known to be robust and easier to tune while GBT models are known to perform better with less subtrees when well tuned.

Among the six models, the first three models (LR, RBF SVM, NB) work well on simple linear hypothesis functions. They are very fast and efficient and will be preferred over other models when comparative predictive accuracies must be obtained. KNN is chosen because it may show unique predictive performance even when other models do not perform well. The RF and GBT models are chosen in cases of very complicated non-linear hypothesis functions and are expected to perform well if the data set has complicated non-linear relationships.

Our choice of models also took into consideration the potential bias-variance trade-off. Both high-bias models (LR, RBF SVM, NB) and high-variance models (GBT and RF) were used. The bias-variance trade-off is introduced and discussed in a later section.

The artificial neural network family [21] of machine-learning models has gained much attention in recent years because of the success of deep neural networks in many research areas [22]. However, the size of our data set is not even close to the comfort zone of any deep neural network; hence, these models are not attempted. Shallow artificial neural networks may capture complicated non-linear hypothesis functions but are usually not comparable to the performance of ensemble decision tree models such as GBT. Also, shallow neural networks are usually slow to train, hard to interpret, large number of hyper-parameters to tune and easier to overfit, compared to models like RF; hence, we did not include this family of models in the comparison.

3.4. Model selection and optimisation methods

Predictive model selection and optimisation are iterative processes guided by statistical metrics generated with certain model evaluation methods. With each iteration, models that exhibit better evaluation metrics are saved as the candidates to be optimised in the next iteration cycle.

In a typical iteration cycle of a single supervised model, the model is trained on the training set and validated on the validation set to generate the metrics. The validation set must be statistically independent of the training set to limit ‘overfitting’. An overfitted model will accurately predict on a validation set, but poorly on an unseen data set.

There are two common methods of model evaluation: holdout and cross-validation methods [23]. The holdout method holds out an independent validation set (different from the test data set, which is only used once in the final test) from the original training data set. This method suffers from the limitation that the model evaluates on the same

validation set in every iteration and thus the model tends to ‘overfit’ the validation set after a certain number of iterations. The cross-validation method rotates the partitioning of the data sets in such a manner that records in the training set are used both as training data and validation data at least once. Since the evaluation metric is calculated multiple times on each rotation, they are averaged to serve as the evaluation metric of cross validation. Since all records in the training data set are used for validation, the cross-validation method will ‘overfit’ the whole training data set after certain iterations of model optimisation. As a better model evaluation method compared to the holdout method, the k -fold cross-validation method was employed in this work.

In a k -fold cross-validation process, the model training/evaluation process is done in k rounds (k is a positive integer). First, the training data set is split into k folds of nearly equal sizes. In each training/evaluation round, $k - 1$ -folds are used to train the model, and the last fold is used to evaluate the model. After k rounds, each fold has acted as validation data once and as training data $k - 1$ times. The metrics generated in k rounds is averaged to provide the overall cross-validation metric.

3.5. Statistical metrics

In the field of supervised machine learning, a confusion matrix is a tabular layout to visualise the performance of binary classification models. In a confusion matrix for binary classification, rows represent the actual class labels from the data set and columns represent the predicted class labels by the trained model, as shown in Table 3. The terms TP, FP, TN and FN are short for true positive, false positive, true negative and false negative, respectively. In these terms, the first word (true or false) indicates if the prediction is correct; the second word (positive or negative) means the predicted class. For example, the cell corresponding to the row of actual positive and the column of predicted positive is TP because it is the correct prediction of positive class; the FN is the wrong prediction of negative which is actually positive; the FP is the wrong prediction of positive class which is actually negative; the TN is the correct prediction of negative class. In the completed confusion matrix, the TP, FN, FP and TN terms will be substituted by the occurrence of the corresponding events as an integer.

For this VPN and non-VPN network flow classification study, accuracy, precision, sensitivity and specificity were used.

Accuracy indicates the overall correctness of prediction of both positive and negative classes by looking at the ratio of correct predictions over all predictions. It is the best metric to evaluate when positive and negative are both important.

Table 3. Statistical classification metrics and confusion matrix.

Confusion matrix		Predicted			
		Positive	Negative		
Actual	Positive	TP	FN	Sensitivity	TP/(TP + FN)
	Negative	FP	TN		
		Precision TP/(TP + FP)	Negative predictive value TN/(FN + TN)	Specificity TN/(FP + TN)	
				Accuracy (TP + TN)/(TP + FP + TN + FN)	

TP: True positive; FP: false positive; TN: true negative; FN: false negative.

Precision, also known as positive predictive value, indicates the correctness of the positive predictions by looking at the ratio of the correct positive predictions over all positive predictions. It is the metric to focus on when a high hit rate of positive prediction is desirable. Negative Predictive Value gives the proportion of predicted values classified as TN.

Sensitivity, also known as recall or TP value, indicates the correctness of how all actual positives are predicted by looking at the ratio of correct positive prediction over all actual positives. It is the most useful when we want the model to pick as many as positive samples from actual positives.

Specificity indicates the correctness of how all actual negatives are predicted by looking at the ratio of correct negative prediction over all actual negatives. It is the counterpart of sensitivity on the negative class, which is important when we want the model to pick as many as possible from actual negatives.

3.6. Hyper-parameter optimisation

Hyper-parameter optimisation is a process of choosing a set of parameters, outside the learning process, that helps to obtain the best outcome of a model. Depending on the complexity of the model, hyper-parameter optimisation is often a 'black art' that requires brutal force [24]. Many approaches, including grid search, random search [25], Bayesian optimisation [26] and gradient-based optimisation [26], were developed to handle hyper-parameter optimisation under different conditions.

Grid search is a traditional and simple way to perform hyper-parameter optimisation. It is an exhaustive search through manually specified grids (full combination of manually picked values of each hyper-parameter) in the hyper-parameter space to look for the set of hyper-parameters that give the best statistical metric score. It provides a good-enough method to find the best combination of hyper-parameters with a simple setup when the search space has reasonable size and the data volume is not huge. In this work, grid searches were performed on both of the RF and GBT models.

3.7. Feature selection

Feature selection, aka variable selection or attribute selection, is a process of selecting the subset of the most relevant features from the data set. By reducing the number of features (variables) in the training data set, the model becomes simpler and the training time is shortened. Also, the removal of less relevant features may improve the generalisation by reducing overfitting. In this work, we focused on recursive feature elimination (RFE), a wrapper-type feature selection method that recursively constructs learning models with each feature excluded and examines the resulting validation metric without this feature. The feature with the least importance, which does not affect the validation metric much when eliminated, in each round is eliminated. As the iterations go on, a smaller and smaller set of features are considered, until further elimination of a feature can no longer improve the metric. The computational cost of RFE methods grows exponentially expensive as the number of features and samples scale. However, it is still an intuitive method to select the best set of features for a given model subject to allowance of resources.

3.8. Bias-variance trade-off

Bias-variance trade-off is a dilemma in supervised learning. This is because it is hard to have a model with high representation power on a training data set and generalise well on unseen data simultaneously. Models with high representation power have the risk of ‘overfitting’ the noisy training data while models that generalise well have the risk of failing to capture (underfitting) the regularity in the training data set.

The underfitting problem is also known as a high bias problem. The underfitting problem is usually caused by (1) not enough data or (2) high-bias model (i.e. the model is too simple to capture the complicated hypothesis function). Little can be done to mitigate underfitting, except to gather more data (more records, more features) and/or switch models.

The overfitting problem is also known as a high variance problem. It has attracted much more attention compared to the underfitting problem. There are many powerful high variance models, and many approaches have been developed to control overfitting. The approaches to limit overfitting include feature selection, hyper-parameter tuning, using ensemble models and adding regularisation (also model dependent).

An easy way to check the existence of the overfitting or underfitting problem of a model is to hold out a test data set for the final model performance confirmation. Low training cross-validation predictive accuracy means underfitting. High training cross-validation predictive accuracy combined with low test accuracy indicates an overfitting problem. To address the overfitting or underfitting problem, in this work, the test data set is held out at the beginning.

The strategy to address potential underfitting and overfitting problem is (1) optimise towards the best cross-validation metric we can get on training data set, (2) if the best cross-validation metric we can get is low (underfitting), change models or rework on the data set, and (3) if cross-validation metric is good but final evaluation on test data set is low (overfitting), try to control overfitting with the techniques mentioned above.

4. Results and discussion

4.1. Model comparison

The six models (GBT, KNN, LR, NB, RF and SVM) were compared in terms of accuracy, precision, sensitivity and specificity metrics for the training data sets of the seven network traffic flow categories. Table 4 and Figure 1 present the average accuracy, precision, sensitivity, specificity across all categories.

Table 4. Average metrics across all categories.

Model	Accuracy	Precision	Sensitivity	Specificity
GBT	0.946	0.948	0.946	0.945
KNN	0.835	0.839	0.825	0.839
LR	0.686	0.704	0.667	0.681
NB	0.565	0.605	0.738	0.421
RF	0.944	0.944	0.946	0.941
SVM	0.676	0.721	0.625	0.690

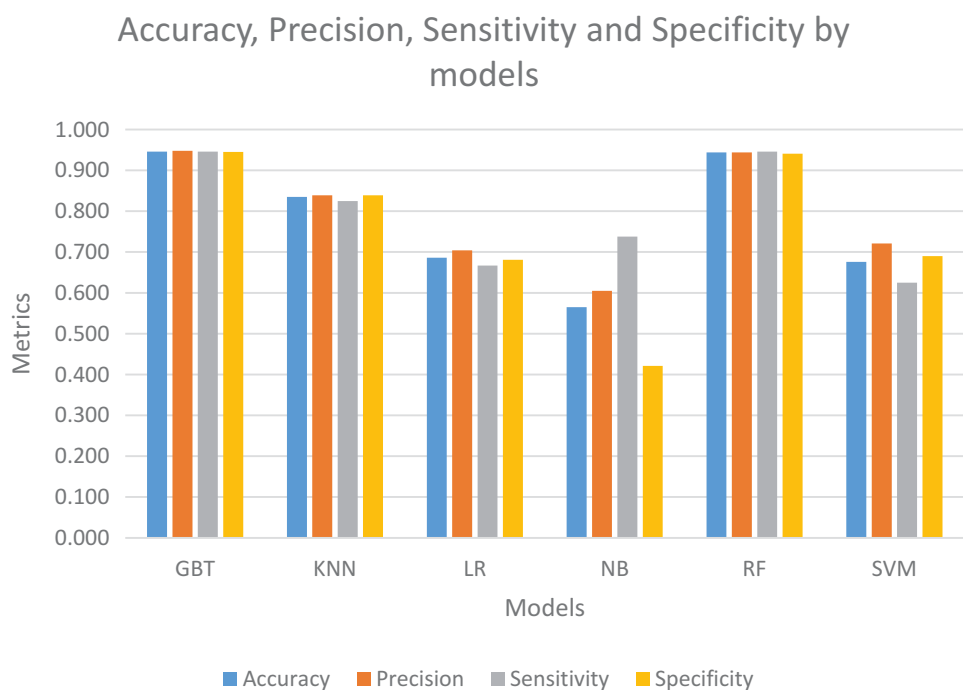


Figure 1. Metrics by internet flow category.

From Table 4, we see that SVM (RBF kernel), LR and NB models exhibited moderately consistent accuracy metrics across the different categories, but the other metrics (precision, sensitivity and specificity) were inconsistent. The KNN model exhibited moderate and consistent metrics. All these four models have (relatively) low metrics indicating an underfitting problem and not worth further optimisation.

The two ensemble decision tree models (RF and GBT) exhibited equally excellent predictive performance in terms of all metrics (average of 94% accuracy for both models). The two models share many common characteristics and do well on other aspects. Thus, the RF and GBT models were optimised in the subsequent stages.

4.2. Model parameter search

For both RF and GBT, the optimal hyper-parameters were determined using the grid search method. The values of the hyper-parameter grid search were chosen manually following the documentation of the SciKit Learn library as listed in Tables 5 and 6.

Because the GBT and RF models are both ensemble models based on decision trees, they share several common hyper-parameters. The number of base estimators indicates how many base decision tree models to generate for the ensemble. This is the most important hyper-parameter for both GBT and RF models. The smaller this number, the simpler the resulting ensembled model, the lesser the resources (time and space) required in the training process, the lesser the chance to overfit, the more the chance

Table 5. Values for random forest grid search.

Hyper-parameter	Values to search	Work-for-all value
Criterion	Gini, entropy	Entropy
Maximum tree depth	Unlimited, 3	Unlimited
Maximum features to use for each decision tree	All, square root of all, two based logarithms	All
Minimum samples leaf	1, 10, 100	1
Number of base estimators	10, 100	100

Table 6. Values for gradient boosting tree grid search.

Hyper-parameter	Values to search	Work-for-all value
Learning rate	0.1, 0.5, 1.0	0.5
Maximum tree depth	Unlimited, 1, 3	Unlimited
Maximum features to use for each decision tree	All, square root of all, 2 based logarithms	All
Minimum samples leaf	1, 10, 100	10
Number of base estimators	10, 100	100

to underfit. The three hyper-parameters that control the individual decision tree growth are maximum tree depth, maximum features to use for each tree and minimum sample leaf. The smaller this number, the simpler the model, the lesser the resources (time and space) required in the training process, the lesser the chance to overfit, and the more the chance to underfit.

The learning rate for GBT model controls the rate at which to apply the gradient in each iteration during the optimisation. The greater the number, the faster the objective function will converge, but the lesser accurate the result. The criterion hyper-parameter for the RF model is the objective function in the decision tree growth [27]. Gini indicates the gini impurity while entropy indicates information gain. Gini is easier to calculate, but generally this hyper-parameter does not change the cross-validation metrics much.

The grid search results indicated that each data set of the seven categories had a different optimal set of parameters. However, a general set of hyper-parameters (work-for-all optimal set) would greatly simplify the subsequent optimisations. The following strategy was applied. Each hyper-parameter of the work-for-all optimal set (our common set) was obtained as the voting result of the same hyper-parameter of the individual optimal set from all seven categories. The scores obtained with work-for-all optimal set parameters were compared to the scores obtained with individual optimal set of hyper-parameters for each category as shown in Table 7 and Figure 2. The small gap between scores in the case of both RF and GBT models confirmed that the work-for-all (general)

Table 7. Hyper-parameter search results.

	GBT individual optimal	GBT general optimal	RF individual optimal	RF general optimal
BROWSING	0.937	0.932	0.930	0.928
CHAT	0.936	0.938	0.924	0.922
FT	0.961	0.959	0.953	0.953
MAIL	0.921	0.925	0.906	0.887
P2P	0.984	0.980	0.974	0.974
STREAMING	0.980	0.973	0.971	0.970
VOIP	0.983	0.982	0.982	0.981

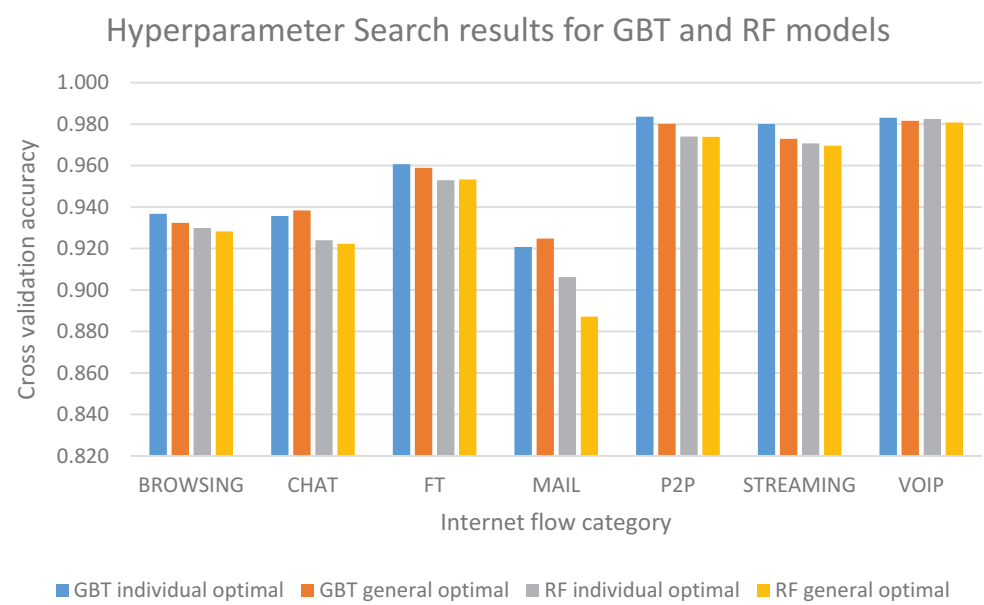


Figure 2. Hyper-parameter search results.

optimal set of hyper-parameters can be safely used on all data sets without compromising the predictive accuracy of the model.

4.3. Feature selection

The optimal amounts of features to select to obtain best model performance on various data sets were examined using RFE. RFE was pipelined to both the RF and GBT models and the resulting accuracy scores were compared to the scores without RFE involvement. As shown in [Figure 3](#), the improvement of the accuracy brought by RFE was trivial for both RF and GBT models. Given the high computational cost of RFE, feature selection with RFE was not considered worthy.

However, maximum accuracy may not always be the most important factor. [Figures 3](#) and [4](#) present the cumulative cross-validation accuracy obtained from the minimum number of features selected by RFE method. It is interesting that although a fairly large number of features was required to obtain the maximum accuracy, only a few features were necessary to obtain ‘good’ results (>90% accuracy as listed in [Tables 8](#) and [9](#)). This result can be considered useful in the scenarios when simplicity (e.g. when building rule based classifiers) and speed (e.g. in real-time classification) are of higher priority than accuracy.

Principal components analysis (PCA) was also attempted by pipelining to the RF model with optimal parameters. The reduced cross-validation score (around an average of 5% reduced accuracy) suggested that the original time-based features were effective enough such that PCA would only reduce the model performance, hence PCA was not used.

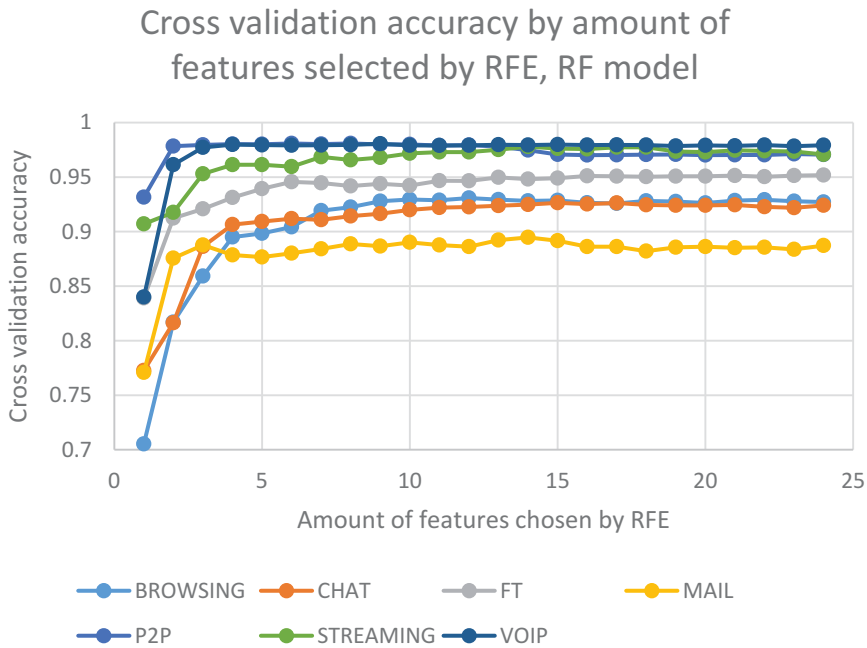


Figure 3. Cross validation accuracy by the number of features selected by RFE, RF model.

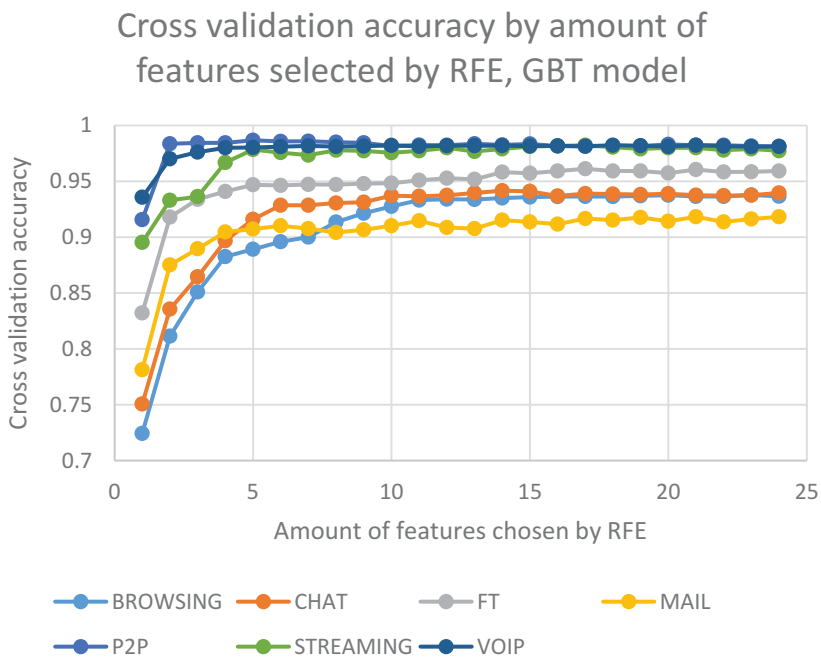


Figure 4. Cross validation accuracy by the amount of features selected by RFE, GBT model.

Table 8. Features that gave >90% accuracy for each category, RF model.

Category	Features to make >90% accuracy (in the order of importance)
BROWSING	max_flowiat, mean_biat, min_flowiat, total_fiat, flowBytesPerSecond, min_fiat
CHAT	min_flowiat, max_biat, flowBytesPerSecond, flowPktsPerSecond
FT	mean_flowiat, max_flowiat
MAIL	min_flowiat, total_fiat, flowBytesPerSecond (To reach 88.7% as maximum accuracy)
P2P	min_flowiat
STREAMING	max_flowiat
VOIP	mean_fiat, mean_flowiat

Table 9. Features that gave >90% accuracy for each category, GBT model.

Category	Features to make 90% accuracy (in the order of importance)
BROWSING	Duration, max_biat, min_biat, max_fiat, flowBytesPerSecond, max_flowiat, mean_flowiat
CHAT	max_flowiat, min_flowiat, mean_flowiat, total_fiat, max_biat
FT	mean_flowiat, max_flowiat
MAIL	mean_flowiat, flowBytesPerSecond, min_flowiat, total_fiat
P2P	min_flowiat
STREAMING	max_flowiat, mean_flowiat
VOIP	mean_flowiat

4.4. Test data set results

In the final test, we used only the RF and GBT models because these two models outperformed the other models. We considered that only the models with a high cross-validation metrics (RF and GBT) are worth an extra check. The models were trained with the training set and used to perform predictions on the test set. Statistical metrics were calculated. As shown in [Tables 10 and 11](#), [Figures 5 and 6](#), both models exhibited excellent predictive power on the test data set. Both models exhibited excellent metrics, while GBT performed slightly better than RF. The small difference, though, was not significant enough to assert that GBT should be chosen over RF because the results are highly data dependent.

Table 10. Final validation metrics of GBT model.

Category	Accuracy	Precision	Sensitivity	Specificity
BROWSING	0.936	0.940	0.929	0.943
CHAT	0.922	0.922	0.929	0.915
FT	0.964	0.959	0.980	0.943
MAIL	0.912	0.901	0.936	0.885
P2P	0.988	0.992	0.978	0.995
STREAMING	0.978	0.995	0.960	0.996
VOIP	0.982	0.980	0.982	0.983

Table 11. Final validation metrics, RF model.

Category	Accuracy	Precision	Sensitivity	Specificity
BROWSING	0.946	0.945	0.945	0.947
CHAT	0.939	0.935	0.948	0.929
FT	0.965	0.966	0.973	0.953
MAIL	0.934	0.914	0.966	0.898
P2P	0.991	0.992	0.986	0.995
STREAMING	0.985	0.991	0.978	0.991
VOIP	0.985	0.981	0.987	0.984

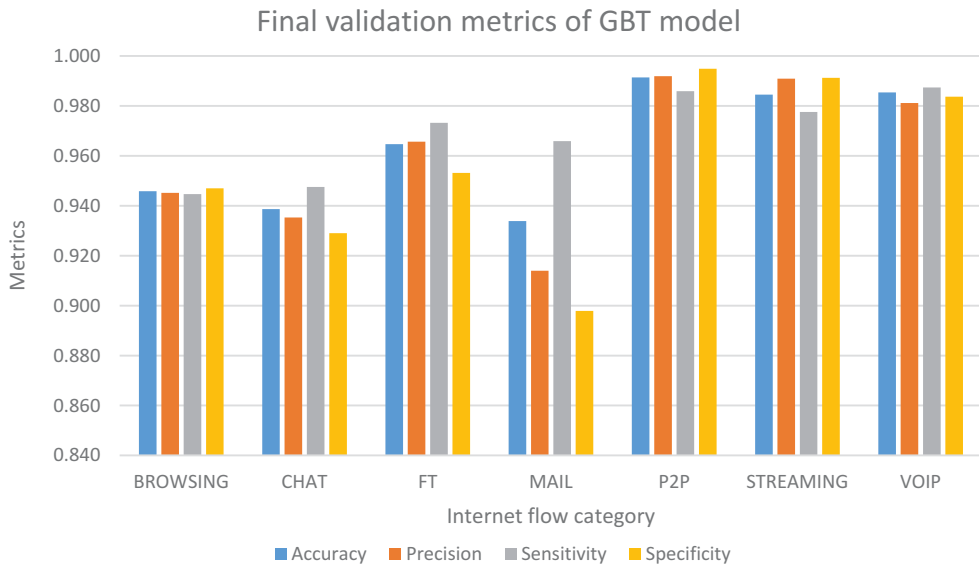


Figure 5. Final validation metrics, GBT model.

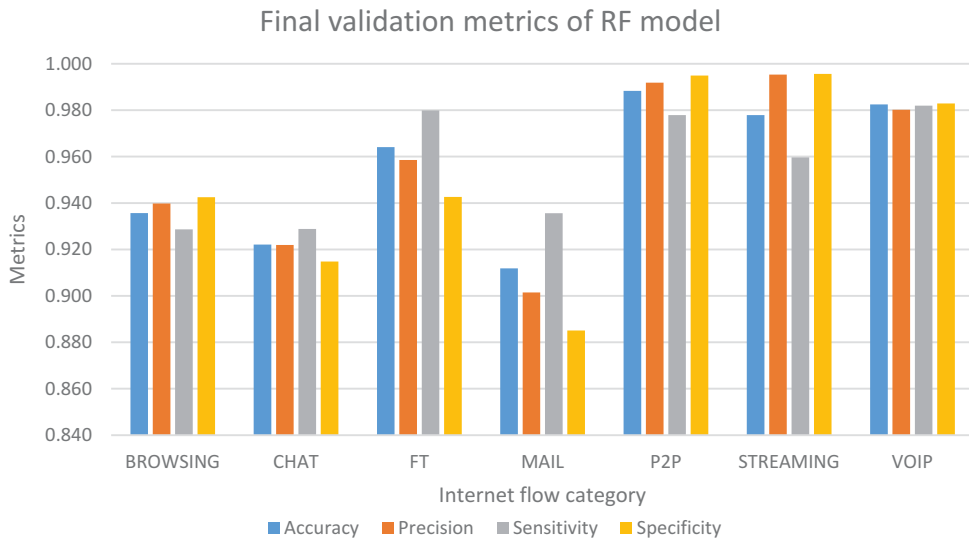


Figure 6. Final validation metrics, RF model.

4.5. Discussion

Since the RF and GBT models outplayed other models (LR, RBF SVM, NB, KNN) by a large margin, the four less effective models, suffering from an underfitting problem, were not entitled to further optimisation. High cross-validation metrics were obtained with both the RF and GBT models and underfitting was not an issue for these models. The fact that the high variance models with high representation power were preferred over simpler

high bias models suggests that the hypothesis functions to be learned by the model were not simple enough to be captured by simple models.

Further optimisations were performed on RF and GBT models because they both exhibited high cross-validation metrics. The hyper-parameter optimisation provided optimal model configurations that improved the model cross validation by a few per cent. Although the addition of the feature selection step to the model did not improve the accuracy, it provided extra insight on the feature contributions to the model prediction. This information can be employed to build simple but fast rule-based classification tools. Feature selection was not added to the final model.

After the optimisation, final tests with unseen test data sets on both models provided slightly lower metrics as compared to the cross-validation metrics obtained in the model optimisation step. These results suggested that the models trained on the training data sets, with the optimised hyper-parameters, were capable of generalising the hypothesis function such that the high cross-validation accuracies were successfully transferred to the unseen data set. This indicated that overfitting was not an issue and further overfitting control was not necessary.

So, we confirm that the time-related feature set is representative enough and the RF and the GBT models perform well on this kind of data. The confidence that they will work well on a future data set with the same features is extremely high. When trained with new data, the RF and GBT models developed in this work should still be capable of predicting future data accurately without significant modification.

5. Conclusions and future work

Six supervised machine-learning models were compared to identify the best model in the VPN and non-VPN network traffic flow classification of data sets of time-related features from various network flow categories. The RF and GBT models with fine-tuned hyper-parameters were developed in this work. These models exhibited excellent predictive performance with no overfitting problem. This study also shows that a small number of time-related features could give over 90% accuracy for each of the network flow categories. These results can be employed in the development of fast rule-based classifiers.

Some recommendations of future work would include collecting network traffic data using comprehensive cybersecurity defensive solutions like Intrusions Detection Systems, Firewalls and Honeynets and using similar machine-learning algorithms to classify the above-mentioned defensive solutions, using the acquired network traffic flow data sets.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Dr. Sikha Bagui is a Professor and Chair of the Department of Computer Science, at The University West Florida, Pensacola, Florida. Dr. Bagui is active in publishing peer-reviewed journal articles in the areas of database design, data mining, BigData, pattern recognition and statistical computing. Dr. Bagui has worked on funded as well unfunded research projects and has over 35 peer-

reviewed publications. She has also co-authored several books on database and SQL. Bagui also serves as Associate Editor and is on the editorial board of several journals.

Xingang Fang, a data scientist and medicinal chemist, teaches programming and conducts research on machine learning, cheminformatics and other topics as a postdoctoral researcher at University of West Florida. Xingang has a PhD in Medicinal Chemistry from Emory University and obtained a MS in Computer Science from UWF after serving 3 years as a Research Associate at The Scripps Research Institute Florida. His past research interests include organic synthesis and drug discovery and he currently focuses on chemical data mining, machine-learning algorithms and applications, Hadoop/Spark applications, computer-aided drug discovery and general scientific computations.

Ezhil Kalaimannan, Ph.D. (ekalaimannan@uwf.edu) is an assistant professor in computer science at the University of West Florida, Pensacola, Florida, working in the areas of cybersecurity and information forensics. He received his master's and doctoral degrees in computer engineering with a specialisation in cybersecurity from the University of Alabama in Huntsville. His research has been largely in the areas of digital forensics, network security and cybersecurity education.

Dr. Subhash C. Bagui received his Ph.D. in statistics from the University of Alberta in 1989. He is currently Professor of Statistics in the Department of Mathematics and Statistics at the University of West Florida. He has authored a book titled, "Handbook of Percentiles of Non-central t-distribution", and published many high quality peer reviewed journal articles. He is currently serving as associate editors/ editorial board members of several statistics journals. His research interests include nonparametric classification and discrimination, clustering, statistical pattern recognition, probability, and experimental designs. He is also a fellow of American Statistical Association (ASA).

Joseph Sheehan has a Master of Science degree from The University of West Florida. He is currently employed as a Software Application Engineer at The University of West Florida.

ORCID

Sikha Bagui  <http://orcid.org/0000-0002-1886-4582>

Xingang Fang  <http://orcid.org/0000-0003-3042-8043>

Ezhil Kalaimannan  <http://orcid.org/0000-0003-2080-7007>

Subhash C. Bagui  <http://orcid.org/0000-0002-1886-4582>

References

- [1] Dainotti A, Pescapé A, Ventre G. A packet-level characterization of network traffic, 11th International Workshop on Computer-Aided Modeling, Analysis and Design of Communication Links and Networks. Trento: IEEE; 2006. p. 38–45. doi:[10.1109/CAMAD.2006.1649716](https://doi.org/10.1109/CAMAD.2006.1649716).
- [2] Schneider P. TCP/IP traffic Classification Based on port numbers. Vol. 2138. Cambridge (MA): Division of Applied Sciences. 1996.
- [3] Paxson V. Growth trends in wide-area TCP connections. IEEE Netw. 1994;8(4):8–17.
- [4] Paxson V. Empirically derived analytic models of wide-area TCP connections. IEEE/ACM Trans Netw (TON). 1994;2(4):316–336.
- [5] Bernaille L, Teixeira R, Akodkenou I, et al. Traffic classification on the fly. ACM SIGCOMM Comput Commun Rev. 2006;36(2):23–26.
- [6] Park J, Tyan H-R, Kuo -C-CJ. Internet traffic classification for scalable QoS provision", In Proceedings of the IEEE International Conference on Multimedia and Expo. 2006. p. 1221–1224.

- [7] Erman J, Mahanti A, Arlitt M, et al. Identifying and discriminating between web and peer-to-peer traffic in the network core. *Proceedings of the 16th International Conference on World Wide Web, WWW 2007*; May 8-12. Banff: ACM. p. 883–892. doi:[10.1145/1242572.1242692](https://doi.org/10.1145/1242572.1242692)
- [8] Williams N, Zander S, Armitage G. A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification. *ACM SIGCOMM Comput Commun Rev.* **2006**;36(5):5–16.
- [9] Nguyen TTT, Armitage G. Training on multiple sub-flows to optimise the use of Machine Learning classifiers in real-world IP networks", *IEEE 31st Conference on Local Computer Networks*, Tampa, FL, 14-16 November 2006; p. 369–376.
- [10] Crotti M, Dusi M, Gringoli F, et al. Traffic classification through simple statistical fingerprinting. *ACM SIGCOMM Comput Commun Rev.* **2007**;37(1):5–16.
- [11] Bonfiglio D, Mellia M, Meo M, et al. Revealing skype traffic: when randomness plays with you. *Proceedings of the ACM SIGCOMM 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Kyoto, 27-31 Aug. 2007.p. 37–48. doi:[10.1145/1282427.1282386](https://doi.org/10.1145/1282427.1282386)
- [12] Draper-Gil G, Lashkari AH, Mamun MSI, et al. Ghorbani, Characterization of Encrypted and VPN Traffic Using Time-Related Features", In the proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP 2016), Italy. pp. 407–414.
- [13] Larsen RJ, Marx ML. *An introduction to mathematical statistics and its applications*. Englewood Cliffs (NJ): Prentice-Hall.**1986**. p. 282.
- [14] Freedman DA. *Statistical models: theory and practice*. New York (NY): Cambridge University Press; **2009**. p. 128.
- [15] Cortes C, Vapnik V. Support-vector networks. *Mach Learn.* **1995**;20(3):273–297.
- [16] Murphy KP. *Naive bayes classifiers*. Vancouver: University of British Columbia; **2006**.
- [17] Altman NS. *An introduction to kernel and nearest-neighbor nonparametric regression*. Am Stat. **1992**;46(3):175–185.
- [18] Polikar R. Ensemble based systems in decision making. *IEEE Circuits Syst Magazine.* **2006**;6(3):21–45.
- [19] Ho TK. The random subspace method for constructing decision forests. *IEEE Trans Pattern Anal Mach Intell.* **1998**;20(8):832–844.
- [20] Friedman JH. Stochastic gradient boosting. *Comput Stat Data Anal.* **2002**;38(4):367–378.
- [21] Patterson DW. *Artificial neural networks: theory and applications*. Upper Saddle River (NJ): Prentice Hall PTR; **1998**.
- [22] LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature.* **2015**;521(7553):436–444.
- [23] Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection, Vol. 14. Montreal: In Ijcai. 1995. p. 1137–1145.
- [24] Snoek J, Larochelle H, Adams RP. Practical bayesian optimization of machine learning algorithms. *Adv Neural Inf Process Syst.* **2012**;2012:2951–2959.
- [25] Bergstra J, Bengio Y. Random search for hyper-parameter optimization. *J Machine Learn Res.* **2012**;13(Feb):281–305.
- [26] Chapelle O, Vapnik V, Bousquet O, et al. Choosing multiple parameters for support vector machines. *Mach Learn.* **2002**;46(1–3):131–159.
- [27] Breiman L. Technical note: some properties of splitting criteria. *Mach Learn.* **1996**;24(1):41–47.