2018-03-30

A Novel Feature Set for Application Identification

H Oudah

B Ghita *School of Engineering, Computing and Mathematics*

T Bakhshi

A Novel Feature Set for Application Identification

Hussein Oudah, Bogdan Ghita, Taimur Bakhshi
Center for Security, Communications and Network Research
University of Plymouth
Plymouth, United Kingdom

Abstract

Classifying Internet traffic into applications is vital to many areas, from quality of service (QoS) provisioning, to network management and security. The task is challenging as network applications are rather dynamic in nature, tend to use a web front-end and are typically encrypted, rendering traditional port-based and deep packet inspection (DPI) method unusable. Recent classification studies proposed two alternatives: using the statistical properties of traffic or inferring the behavioural patterns of network applications, both aiming to describe the activity within and among network flows in order to understand application usage and behaviour. The aim of this paper is to propose and investigate a novel feature to define application behaviour as seen through the generated network traffic by considering the timing and pattern of user events during application sessions, leading to an extended traffic feature set based on burstiness. The selected features were further used to train and test a supervised C5.0 machine learning classifier and led to a better characterization of network applications, with a traffic classification accuracy ranging between 90-98%.

1. Introduction

In 2017, a report published by Cisco indicated that the global IP traffic was 1.2 ZB per year in the end of 2016. By 2021, this trend might increase up to 3.3 ZB per year [1]. This traffic contains many types of applications, such as voice, video, gaming, web browsing, all with different requirements. Classifying such heterogeneous traffic into applications based on its network footprint is a challenging but necessary task, hence it has received much attention over the past decades on account of its importance in profiling user activity, providing intelligent network management, and detecting network intrusion or traffic anomalies [2]. Different techniques could be utilized for IP traffic classification. The traditional way of identifying this traffic typically focused on using IANA assigned port numbers and DPI [3, 4]. This technique based on matching the port number of the packet header with the port that was given by IANA. However, an increasing number of Internet applications nowadays use dynamic post assignments and tunnelling, which renders port-based traffic classification extremely challenging and prone to errors. DPI is useful and the results of many studies showed high accuracy. Nevertheless, this method requires significant computational resources, presenting scalability issues in achieving real-time traffic identification, and cannot cope with the encrypted traffic as this method deals with the payload rather than header of the packet. Hence, most recent studies such as [5, 6] focused on statistical approach, which is able to characterize traffic associated with an application based upon statistics features. This approach requires access the only to packet headers, improving the profiling performance and dealing with encrypted traffic. However, this method highly relies on the quality of the feature set that needs to sufficiently discriminative in order to distinguish between applications. Subsequently, identifying the optimal feature set for applications reduces the potentially large dimensionality and might be useful to improve the system performance [7]. Moreover, because the method employs machine learning algorithms (MLAs), which are accurate in detecting statistical features of the applications, the training data should be robust for the initial learning of the MLAs to output high performance[8]. The main contribution of this paper is a novel set of traffic burstiness features that improves the identification of network applications through statistical traffic analysis.

The rest of this paper is organized as follows. Section 2 discusses state-of-the-art traffic classification approaches in more detail to provide a comprehensive review of the limitations of existing techniques. Section 3 highlights the proposed traffic classification design and discusses classifier training using derived traffic feature sets. Section 4 evaluates the optimal C5.0 based machine learning classifier while conclusions are drawn in section 5.

Table 1. Existing traffic classification approaches and challenges

Classification Approach	Method	Limitations
Port-Based	IANA assigned port-mappings	Dynamic port-assignments and tunneling
Deep Packet Inspection	Packet content and header analysis	Computational overhead
Host Behavior Analysis	Analyze host behavior and application traffic pattern	Applications with similar behavior are difficult to classify
Statistical Analysis	Identify applications using numerical traffic features	Difficult to obtain high quality ground-truth training data
Combinatorial/Hybrid	Multiple approaches, combination of machine learning techniques	Specific to individual network settings, lack scalability of use

2. Background

Traffic classification is a fundamental requirement for service providers to guarantee QoS, malware detection and management. During the last decade, a significant amount of effort focused of identifying and characterizing Internet traffic. This work could be grouped into four approaches: port-based, packet payload-based, behaviour-based and statistical features-based). Table 1 summarises these methods, while the following subsections will introduce these method in detail.

2.1. Port-based approach

Historically, traffic classification typically used port-based identification, which was simple to implement and yielded high accuracy for certain applications such as SMTP or DNS given their static use of specific port numbers. However, most of the present Internet applications either use dynamic port numbers randomly without any prior assumption or use encryption and tunnelling traffic through well-known port such as 80 443 [9]. For example, Skype and P2P applications use TCP port 80 [10] which would appear to be web browsing when using port-based classification, despite the fact that it could be messaging, file transfer or voice communication traffic. As such, port-based traffic classification is now considered ineffective, providing at most 70% accuracy when tested against other available methods [11]. Other studies, such as [12], highlighted the fact that, although it provides low classification accuracy, this method is still relevant in Internet backbone due to its scalability and minimal computational power required. In other words, port-based classification aids in determining the overall application trends when combined with additional techniques resulting in hybrid approaches. Many recent studies therefore combined port-based with machine learning and statistical analysis of network traffic, resulting in higher accuracy as discussed later in section 2.4.4.

The next subsection reviews the DPI method of traffic classification showing its strengths and limitation.

2.2. Deep packet inspection approach

It has been argued that the low accuracy associated with the port-based method can be solved using DPI. The results showed that the accuracy of this method was very high up to 98% [13]. The evolution of DPI started by recording the signatures of each application or protocol format using reverse engineering or vendor white papers that are describing the behavior of these applications. In [13], DPI was used to identify P2P applications by producing signatures for each P2P application according to available documentations and analysis of packet traces. The recorded signatures were subsequently used in designing filters to identify these applications in real-time traffic. The authors chose five P2P applications to test the filters and the results showed low error rate less than 5%. Moreover, the study claimed that the technique could classify P2P applications by inspection the first 10 packets which makes the approach more scalable for high-speed analysis. However, DPI requires significant computational resources, and may be Furthermore, breaches the user privacy due to the inspection of the packet contents that might reveal personal data, hence appearing illegal concerns related to regulations and agreement policies.

2.3. Host-behavioral based approach

This technique is based on the idea that hosts generate different communication patterns at the transport layer; by extracting these behavioural patterns and activities, applications could be classified. For instance, the authors in [14] proposed a novel scheme to identify P2P applications based on heuristics that were extracted from P2P traffic such as IP ratio, port pair differences and the failed connection

ratio. Also, they used statistical features of flows that were generated from the host to refine the identification accuracy, which was 96.3%. Other studies [15, 16] showed acceptable performance (over 90%). However, this approach can detect the application type, but it cannot correctly identify the application names, classifying both Yahoo or Gmail as email [12].

2.4. Statistical features-based approach

In contrast, very good accuracy over 95% was achieved by applying the forth approach using statistical traffic features, such as packet size, number of packets , inter-arrival time of the packets, and flow duration. This approach has many advantages regarding encrypted traffic as it uses header information of the packet rather than payload. Hence, it is likely to be light-weight and fast in application detection, especially in the real time environment [5] or for peer-to-peer applications [17]. However, selecting features, which must be adapt to the application and traffic variability, is the significant point to build a classifier [18]. Given this classification, the approach outlined in this paper strengthens this method (statistical) by considering the arrival times of packets and flows as discriminating features among applications. The authors in [19] analysed burstiness in network traffic by using a measure called Index of Variability. The hypothesis that timing can be used to discriminate between applications was also put forward in [20], which highlighted that applications generate different behaviour based on statistical features relating to the timing of packets arriving. In addition, the machine leaning algorithms enable the approach to be more efficient and reliable in traffic identification [21]. Therefore, the statistical method can be categorized based on the ML algorithms being used and as follow: Supervised, unsupervised and semi-supervised and they will be expanded in the following subsections.

2.4.1. Supervised learning techniques. This technique relies primarily on a quality of training data with the ability of the algorithm to discriminate between applications. This data represents the signatures of the application which is used to build the classifier model. The technique is efficient and produces high accuracy, however, it cannot identify new applications which are seen for the first time [22]. A number of recent studies [23, 24] have used supervised learning techniques in tandem with flow records to classify traffic. For example, Zhang in [23] enhanced a non-parametric supervised approach (NN) to achieve better classification by merging the correlation information into the classification process. Similarly, Hajikarami in [24] proposed a two layer system which was fast and accurate in classifying predefined applications and capable of classifying new applications from unknown samples. A Correlation-based Filter (CFS) was used to extract the optimal classification features for each group and C4.5 decision tree was used for classification. The results showed high accuracy up to 99.55%. Nonetheless, the statistical method based on these algorithms requires a large amount of training data and any changing in the training data, such as changes in user behaviour, might affect the results negatively [25].

2.4.2. Unsupervised learning techniques. To address the issues associated with availability of training data, some studies have employed unsupervised ML techniques [26, 27]. Since this technique does not need prior knowledge of pre-labeled data, it can be used to explore new applications. Alizadehin [26] used unsupervised Gaussian Mixture (GMM) to classify applications based on statistical information to detect traffic anomalies. The traffic was processed using Sequential Forward Selection (SFS) for selection of the optimal features. The authors selected four applications (Mail, p2p, Web Browsers, and Skype). The method showed an ability to identify the type of traffic successfully except for Skype. Also, Keene in [27] tried to reduce the computation time in K-NN algorithm using the principal component analysis (PCA) feature selection method. The results showed that the proposed method outperformed other approaches in processing time while the accuracy was relatively low. Overall, the accuracy of unsupervised techniques remains lower as compared to supervised ML classifiers.

2.4.3. Semi-supervised learning techniques. In addition to static supervised and unsupervised algorithms, a growing number of studies have also utilized semi-supervised learning algorithms employing both labelled and unlabelled data [28]. Studies such as [29, 30] proposed semi-supervised algorithms which combine two or more ML algorithms to detect new applications. Zhang in [29] studied the problem of zero-day applications using machine learning algorithms. The study utilized unsupervised technique to label the unknown applications and supervised technique for training the model. Statistical features were extracted from flows to be the input to the proposed classification system. The experimental results showed that the system outperformed other methods (semi-supervised clustering, one-class SVM, random forest and correlation-based classification). Similarly, Lin in [30] studied the problem of unknown protocols in traffic when the traditional methods misclassified the unknown samples which led to reduction in classification accuracy. The authors used three real-time databases (WIDE-10, WIDE-12, and CND) that were collected within different time period. The classification algorithm used semi-supervised Learning (UPCSS) that was powerful in

discriminating the new samples; hence the efficiency of the classifier increased even with limited training samples. Two metrics were used to evaluate the accuracy of the proposed method i.e. overcall precision and F-measure. The results showed that the proposed method outperforms other methods including Random Forest, Eman and RTC. However, Generation of high quality of labelled or unlabeled data with reliable features remains a challenge.

2.4.4 Hybrid approach. It can be noticed that each method has its strengths and limitations. Therefore, different traffic classification algorithms can be integrated together to obtain high identification accuracy. For example, studies such as [12, 31] proposed combining different methods to achieve superior accuracy. Park in [12] proposed new technique called functional separation method to classify traffic. The authors collected data from the end-hosts using traffic collecting agent, and a functional separation method partitioned each application according to its function. In this stage, port-based classification was used to group the application functions according to the port number similarity. Later, payload-based and communication patterns were used for each group to check the inter-group application similarity. Similarly, Lu and Xue in [31] utilized two approaches (port and payload) to identify Internet traffic. The study used co-clustering method and basic parameters (source/destination IP and destination port number) to characterize the host behavior. The proposed technique first divided the flows into TCP and UDP, and used the payload to classify all the flows into known and unknown traffic. These flows were later combined and the co-clustering method used to cluster the traffic into host communities using port numbers. Finally, each host community was clustered according to destination IP addresses. The experiment was performed using the data collected from the large scale ISP for two days, and the results showed high accuracy. Nevertheless, these studies suffer from the complexity of analysis of using more than one approach

To this end, it can be noticed that each method has its strengths and limitations; hence, a valid experiment is necessarily required in order to select the suitable approach that can be used to classify the traffic network in an effective manner.

2.5. C5.0 machine learning algorithm

C5.0 is a decision tree algorithm, which is an improvement of previous C4.5 [32]. Both the algorithms have the same properties, except new technologies were added to the new algorithm such as boosting that yields an optimal classification [27]. The source code of this algorithm was made publically available, and also incorporated into data analysis tools such as R programming language. In decision trees, the first challenging task is to recognize which parameters to split data upon. C5.0 uses entropy to measure the segments of data that includes only a single class. The entropy of a sample of data refers to how the class values are mixed. If the entropy is equal to 0, that means the sample of data is completely homogenous, while, if it is 1, that means the segment of the data are non-homogenous. The definition of entropy is specified by the following equation:

$$Entropy(s) = \sum_{i=1}^{c} -pi\ log2(pi)$$

Where s,c, and pi refer to given segment of data, number of different class level, and proportion of values falling into class level i. This algorithm was employed in previous studies [5, 21] which reported good performance. In [21], the authors utilized C5.0 algorithm to distinguish between HTTP and non-HTTP traffic and produced low error rate 6%. Moreover, they used the same method to classify HTTP traffic in real time environment. Although, the error rate was high (17%), the authors claimed that the error were in training and test set. Also, in [5], the authors used C5.0 algorithm to identify fifteen types of traffic with the aid of clustering K-NN algorithm. The results showed high accuracy after using boosting features of C5.0 (96.67%). Moreover, this approach has a capability to select the best parameters from the feature set and provide a best solution under different network circumstances.

3. Proposed classification method

The research community proposed many solutions and methods for traffic classification, in detail, we illustrated the limitations and strong points in each one. It seems that the statistical method is more popular than the others because of the suitability of applying it in the real-time environment and its ability to deal with the encrypted traffic. However, this method relies mainly on the correct features that describe accurately the Internet traffic. The current investigation of this work focuses on analysing the timing characteristics, more specifically burstiness, in the traffic generated by an application as a discriminator. This is based on the fact that Internet applications behave inherently different, generating different amounts of data, creating various numbers of connections, and producing different timing patterns between the generated flows. For example, when a user watches Netflix, it is obvious that different patterns would be generated if compared to the same user checking E-mail or browsing social applications (e.g., Facebook or Instagram). The concept of traffic bursts can be described as a group of consecutive packets with shorter inter-packet gaps than packets arriving before or after the burst of packets; the minimum number of packets to form a burst are two

packets [33]. Accordingly, burstiness is a measure of the variability of packet arrival spacing for a traffic flow over time. Figure 1 shows how the group of packets forms a burst based on inter-packet arrival times, based on a set threshold (Burst_threshold) that defines the maximum inter-arrival time for two

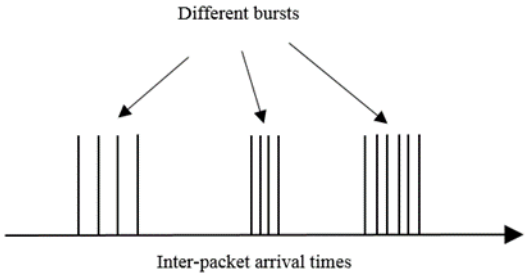

Figure 1. The burstiness concept

packets to belong to the same burst. Initially, some features can be calculated from this figure such as number of bursts per flow, size of bursts and number of packets in bursts. A study was carried out as part of this study to determine the value of Burst_threshold as shown in figure 2. The figure shows the inter-packet arrival times for six applications (i.e., BBC news, Facebook, Google search, Skype, Yahoo mail, YouTube) in msec. Most distributions of the inter-packet arrival times fall under 1 second. Accordingly, the Burst_threshold was set to 1 second. The concept of burst, which is proposed by this work, was implemented in tcptrace tool. The pseudocode in algorithm 1 summarises the estimation of bursts; this code was written in C script as the tcptrace source code. The process of bursts calculation could be illustrated as follow. Firstly, the inter-arrival time between successive packets is computed, if it is less than burst_threshold, a new burst is formed, and some values would be accumulated such as current burst and current session. Otherwise, if the inter-packet arrival time is greater than burst_threshold, it means that the previous burst was finished and new one would be formed and so on. This process was carried out for each direction of the flow and by calculating the number of bursts that were formed per direction.

Algorithm 1: Estimation of packet bursts time

```
Burst threshold= 1s
initialise burst and idle time parameters
while packets arriving
do
 calculate interarrival_time
 if interarrival_time < burst_threshold
  current_burst ++
  current_session ++
 else
  burst_counter ++
  current_burst = 1
 fi
done
```

Also, the parameters for each direction were computed such as number of packets in the total bursts for this direction, size of these packets and duration of the burst. The possible features that could be extracted as output from the pseudocode can be classified into two groups. The first group is related to the burst features that are calculated between packets arrival time within flows. In addition, this group contains two types of burst features which were calculated for either all packets or for only data packets in flow. To distinguish between two types of the first group, we denoted the burst features for only data packets as "data", Table 2 displays the proposed features. The first group features were calculated by writing a script inside tcptrace tool. The second group is related to the burst features that were calculated between flows

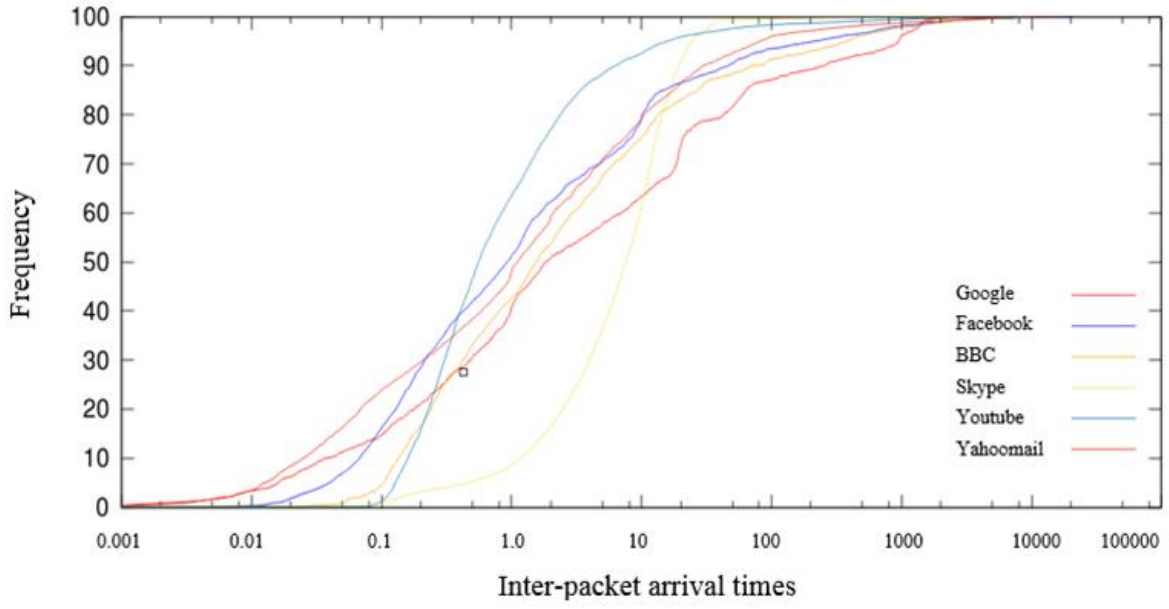

Figure 2. The distribution of inter-packet arrival times (msec)

arrival time. These features are number of bursts between flows and number of flows in bursts. The second group is calculated through writing a script in R. Both of these groups were fed to the C5.0 classifier as will explain in the next section.

Table 2. Proposed parameters for each direction

Features	Description
Burst_no_a, Burst_no_b Burst_data_no_a, Burst_data_no_b	Number of bursts that contain all packets or only data packets
Pkt_count_a, Pkt_count_b Pkt_data_count_a, Pkt_data_count_b	Number of packets in bursts for all packets or only data packets
Burst_size_bytes_a, Burst_size_bytes_b	Size of bursts in bytes
Burst_duration_a, Burst_duration_b Burst_dur_data_a, Burst_dur_data_b	The duration of bursts for all packets or only data packets
Inter_arrival_time_burst_a Inter_arrival_time_burst_b	The time duration of bursts divided by the total packets

The architecture of the proposed system is illustrated in figure 3, showing the components of application scheme. Primarily, the data was captured from six users that accessed six Internet applications, which are frequently used by the users. The raw traffic was analysed using tcptrace tool that has been modified to generate the proposed features. Tcptrace is a tool that manipulates with packets as input and generates flows as output. The tool was exploited by writing the script inside it that computed the burstiness features. Afterword, R scripts were written for pre-processing the output of the previous stage and create additional features derived from the proposed ones, as shown in the figure. Finally, five statistical operation (i.e., maximum, minimum, mean, median, and standard deviation) were applied only upon the basic and packets burst features. The aim of these processes is to summarize the output of all features in one row for each operation. Hence, it reduced the database of sessions that led to decrease in time consuming. The combined set of newly generated features was fed into a C5.0 ML algorithm to derive a traffic classifier. The data were divided into two parts, the first part was for training and to build the classifier model while the second part was for testing. More details are explained in the following sub sections.

4. Evaluation

The proposed method was evaluated by utilizing C5.0 decision trees algorithm. The classifier was built based on data that were captured from six users. Each user was asked to browse six applications (i.e., BBC news, Facebook, Google searching, Skype, Yahoo mail and YouTube,). The reason for selecting these applications as they are considered to be the most well-known applications [34].

Table 3. Summary of the data collection

Application	Flows	Duration (h)
BBC news	32,596	15.6
Facebook	5,620	12.9
Google searching	27,640	8.5
Skype	2,632	9.88
Yahoo mail	48,116	10.22
YouTube	11,233	11.3

Table 3 summarizes the data collection of the conducted experiment. The users accessed separately each application for (30) times and each time was for (2-5) minutes. The users were limited to using only a single application at any session and the dump files were accordingly labelled with the name of the accessed application. The large and separated dataset made the training data more robust which enabling the classifier to learn properly. The collected data were split in the following approximate proportion: 65% of samples were used for training the C5.0 classifier, while testing was done using the remaining 35% samples per application.

4.1. Recorded accuracy

The accuracy was calculated for the basic features (set1) that was proposed by previous studies and for the proposed features (set2). Set 1 included the following features: number of data packets, number of flag packets, size of the first packets, time duration, inter-arrival-time, received packets to transmitted packets, received of data packets to transmitted data packets, received of flags packets to transmitted of flags, transmitted of flags packets to transmitted packets and received of flags packets to received packets. While set 2 included the features that were shown in table 2. Moreover, the accuracy was calculated for set 3 which is the combining of two sets.

Table 4. Accuracy of the classifier for feature sets

Feature set	No boost	Boost 10	Boost 100
Set 1	93 %	97.33%	97.5%
Set 2	94.33%	96.83%	96.83%
Set 3	90.7%	97.96%	97.96%

The accuracy for these sets are shown in Table 4 and range between 90-97.96%, the accuracy for basic features exceeded the accuracy obtained by the proposed features. However, the accuracy reached to the highest level when the both sets were combined together. In other words, the proposed parameters enhanced the classifier ability to discriminate the

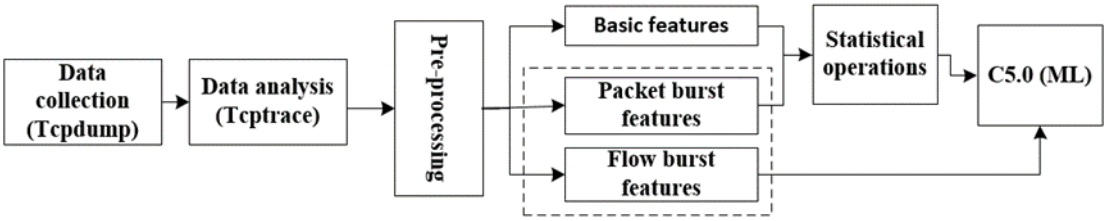

Figure 3. Proposed traffic classification methodology

different traffic that were generated from the applications. The attributes usage (percentage) by the optimal C5.0 in computing the decision tree using feature set 3 is given in Table 5. The table shows the comparison between basic and proposed features. The attributes in interval (100%) reported maximum usage in segregating among the six applications. Also, the parameters in interval (75-99) % showed strongly usage by the classifier. The proposed attributes between packet streams and flows were majority part compared with the basic features which were offered differentiation among applications activities. This is another indicator that the classifier strongly relied on the proposed features because they provide high discrimination between applications.

4.2. Confusion matrix

The accuracy, as presented in the previous section, represents only the ratio of correctly classified instances versus all instances. Therefore, the accuracy does not give us any indication of which class has the error. For further investigating, the performance of the classifier for each class can be represented using confusion matrix. The confusion matrix is a table that displays the visualisation of the classifier performance. It has two dimensions, one for predication instances and the other for the actual instances. Correct classification occurs when the predication instances match with the actual instances. For example, confusion matrix table might be 2*2 or 3*3 as shown in Figure 4, or more than three depends on the number of classes. The row shows the instances in the predicated class while column shows the instances in the actual class. The diagonal of the matrix represents the number of samples that are correctly classified as interest class and called True Positive (TP). The rest of the values in the row of each application are misclassified as the class of interest and called False Positives (FP), and the rest of the values in the column of each application are misclassified as not the class of interest and called False Negatives (FN). The most important point in this measure that it specifies the classifier ability to distinguish one class versus all classes. The performance of the classification model (classifier) on the test data are shown in Table 6 using the confusion matrix. The overall performance of the classifier is considerably high for all applications except Google browsing. Out of a total of sixty samples containing several flows, packet streams and corresponding features, there were four instances classified as (FN) with Yahoo Mail.

Table 5. Attributes Usage in C 5.0 Classifier

Basic features usage (the statistical operations calculated for all flows)	
100%	Mean & median for number of transmitted packets, mean for number of transmitted data packets, mean_flow_duration_b, Max_flow_duration_b, median for the first packets_a & the first packets_b, standard deviation of inter arrival time_b
75-99%	Mean no. of data Packets_b, median for no. of flags packets_a / no. of packets_a, standard deviation for the first Packets_b, Max for inter arrival time_a, mean for the first packet, mean for inter arrival time_b, standard deviation for ratio of no. of packets in both directions, mean for no. of flags packets_b, standard deviation for number of data packets_a, standard deviation for number of packets_a
Proposed features usage (the statistical operations calculated for all flows)	
100%	Mean for number of data burst_a, mean for the inter arrival time_data_b, Max for number of packets in burst_b , Max for data burst_b, Max for data burst duration_b, Max for average of the size of the data burst_b, median for the duration burst_a, median for the inter arrival time_data_a, standard deviation for burst_duration_b, No. of connections for each session, No. of connections in bursts, mean for the ratio of size of burst in both direction, Max for the ratio of the size data burst in both direction

75-99%	Max size of burst_b, Max no. of burst_b, median for the ratio of the burst size in both directions, standard deviation of the no. of packets in burst, mean for the inter arrival time in the burst, Max no. of the data burst_a , standard deviation for the average of the size burst_b, median for the ratio of the data burst size in both directions, Max for the number of packets in burst_a, median for average size of data burst_a, standard deviation for the size burst_a, standard deviation for the size burst_a, standard deviation for the inter arrival time in burst_a, No. of bursts in connections, mean for the size of data burst_b, Max for the inter arrival time in data burst_a, standard deviation for the ratio size burst in both directions, standard deviation for the size of data burst_b, standard deviation for the no. of data packets in burst_a

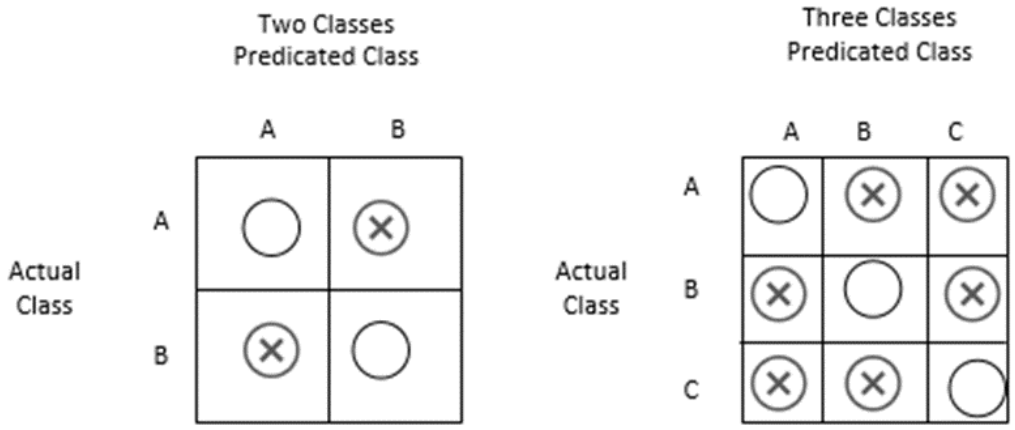

Figure 4. Confusion matrix tables

Table 6. Confusion Matrix table (Feature Set 3)

Application name	**BBC news**	**Facebook**	**Google**	**Skype**	**Yahoo Mail**	**YouTube**
BBC news	**59**	0	0	0	0	0
Facebook	0	**59**	0	0	1	2
Google	0	0	**55**	0	0	0
Skype	0	0	0	**60**	0	0
Yahoo Mail	1	0	4	0	**59**	0
YouTube	**0**	1	1	0	0	58

4.3. Sensitivity and specificity factor

These parameters are a measure of ability of a classifier to identify and discriminate samples of given classes. Sensitivity refers to the derived model's capability to predict the samples that belong to a class or application, while specificity refers to the generated prediction model's capability to mark and differentiate that these samples as not belonging to a given class. Sensitivity therefore avoids the false negative (FN), while specificity avoids the false positives (FP). Accordingly, the sensitivity and specificity can be defined as in the following equations:

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

The optimal balance between these factors relies on the type of application that being used. The relationship between these parameters is a trade-off,

when one parameter increase the other decrease. For instance, when the testing occurs in the airport security, the alarm is set on a low-risk items (low specificity) to be likely identify dangerous items and avoid missing any objects (high sensitivity). Both sensitivity and specificity factors for the built classifier are shown in Figure 5 using feature set 3 with a boost factor of 100. The sensitivity of Google browsing was the lowest (97%) due to misclassification with Yahoo Mail. The overall sensitivity ranged above 97%. Also, specificity factor across all six applications was considerably high ranging between (98-100) percent, depicting high segregation ability of the prediction model.

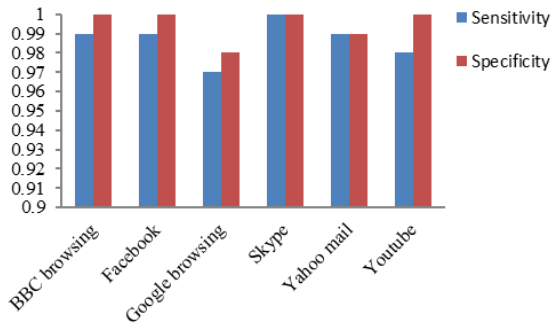

Figure 5. Sensitivity and specificity factors

5. Conclusion

This work proposed an algorithm for applications identification based on a novel feature set. The study exploited the inter-arrival times between packets and flows to generate these features, most specifically burstiness. The selected parameters were evaluated based on data that were collected from six users browsing six applications. Afterwards, the data were analysed using tcptrace tool and fed to the C5.0 classifier for training and testing. The results showed very high accuracy for the proposed method up to 98%. The proposed features set enhanced the ability of the classifier to predict the application type when it added to the previous studies features.

As part of future work, the set of Internet applications considered by the present study will be expanded to include other applications (i.e., online shopping, email, news websites, photo sharing websites, search engines, etc.). Moreover, more experimental work would investigate the visibility of utilizing the inactive time within packets and flows. Different machine learning algorithms would be evaluated to investigate their effects on system performance.

6. References

[1] Cisco, “Cisco visual networking index: forecast and methodology,” 2016–2020 white paper; 2016. Available from: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360 .html.

[2] T. Bakhshi and B. Ghita, “User traffic profiling,” *2015 Internet Technol. Appl. ITA 2015 - Proc. 6th Int. Conf.*, no. November, pp. 91–97, 2015.

[3] M. S. Joe Touch; Eliot Lear, Allison Mankin, Markku Kojo, Kumiko Ono and and A. Z. Lars Eggert, Alexey Melnikov, Wes Eddy, “IANA.” [Online]. Available: http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml. [Accessed: 04-Mar-2016].

[4] J. Sherry, C. Lan, R. A. Popa, and S. Ratnasamy, “BlindBox: Deep Packet Inspection over Encrypted Traffic,” *Proc. 2015 ACM Conf. Spec. Interes. Gr. Data Commun. - SIGCOMM ’15*, pp. 213–226, 2015.

[5] T. Bakhshi and B. Ghita, “On Internet Traffic Classification: A Two-Phased Machine Learning Approach,” *J. Comput. Networks Commun.*, vol. 2016, no. May, 2016.

[6] A. Vlăduţu, D. Comăneci, and C. Dobre, “Internet traffic classification based on flows’ statistical properties with machine learning,” *Int. J. Netw. Manag.*, vol. 27, no. 3, p. e1929, May 2017.

[7] T. Antonio and A. S. Paramita, “Feature selection technique impact for internet traffic classification using Naïve Bayesian,” *J. Teknol.*, vol. 72, no. 5, pp. 141–145, 2015.

[8] J. Cai, Z. Zhang, and X. Song, “An analysis of UDP traffic classification,” *Int. Conf. Commun. Technol. Proceedings, ICCT*, pp. 116–119, 2010.

[9] J. M. Reddy and C. Hota, “Heuristic-Based Real-Time P2P Traffic Identification,” *2015 Int. Conf. Emerg. Inf. Technol. Eng. Solut.*, pp. 38–43, 2015.

[10] S. A. Baset and H. G. Schulzrinne, “An analysis of the Skype peer-to-peer internet telephony protocol,” *Proc. - IEEE INFOCOM*, 2006.

[11] V. C. Español, “Network traffic classification: from theory to practice,” Barcelona University, 2014.

[12] B. Park, Y. Won, J. Chung, M. Kim, and J. W.-K. Hong, “Fine-grained traffic classification based on functional separation,” *Int. J. Netw. Manag.*, vol. 23, no. 5, pp. 350–381, 2013.

[13] S. Sen, O. Spatscheck, and D. Wang, “Accurate, scalable in-network identification of p2p traffic using application signatures,” *Proc. 13th Int. Conf. World Wide Web*, p. 521, 2004.

[14] J. Yan, Z. Wu, H. Luo, and S. Zhang, “P2P traffic identification based on host and flow behaviour characteristics,” *Cybern. Inf. Technol.*, vol. 13, no. 3, pp. 64–76, 2013.

[15] A. Bashir, C. Huang, B. Nandy, and N. Seddigh, “Classifying P2P activity in Netflow records: A case study on BitTorrent,” *IEEE Int. Conf. Commun.*, pp. 3018–3023, 2013.

[16] J. Hurley, E. Garcia-Palacios, and S. Sezer, “Host-Based P2P Flow Identification and Use in Real-Time,” *ACM Trans. Web*, vol. 5, no. 2, pp. 1–27, 2011.

[17] A. Ulliac and B. V Ghita, “Non-Intrusive Identification of Peer-to-Peer Traffic,” in *2010 Third International Conference on Communication Theory, Reliability, and Quality of Service*, pp. 175–183.

[18] A. Hajjar, J. Khalife, and J. Díaz-Verdejo, “Network traffic application identification based on message

size analysis," *J. Netw. Comput. Appl.*, vol. 58, pp. 130–143, 2015.
[19] G. Y. Lazarou, J. Baca, V. S. Frost, and J. B. Evans, "Describing Network Traffic Using the Index of Variability," *IEEE/ACM Trans. Netw.*, vol. 17, no. 5, pp. 1672–1683, 2009.
[20] M. Roughan and S. Sen, "Class-of-service mapping for QoS: a statistical signature-based approach to IP traffic classification," *Proc. 4th …*, pp. 135–148, 2004.
[21] T. Bujlow, T. Riaz, and J. M. Pedersen, "A method for classification of network traffic based on C5.0 machine learning algorithm," *2012 Int. Conf. Comput. Netw. Commun. ICNC'12*, pp. 237–241, 2012.
[22] P. Pinky and S. E. V. Ewards, "A Survey on IP Traffic Classification Using Machine Learning," *Int. J. Eng. Res. Appl.*, vol. 3, no. 1, pp. 2099–2104, 2013.
[23] J. Zhang, Y. Xiang, Y. Wang, W. Zhou, Y. Xiang, and Y. Guan, "Network traffic classification using correlation information," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 1, pp. 104–117, 2013.
[24] F. Hajikarami, M. Berenjkoub, and M. H. Manshaei, "A Modular Two-layer System for Accurate and Fast Traffic Classification," *Inf. Secur. Cryptol. (ISCISC), 2014 11th Int. ISC Conf. (pp. 149-154). IEEE*, pp. 149–154, 2014.
[25] T. Bakhshi and B. Ghita, "Traffic profiling: Evaluating stability in multi-device user environments," *Proc. - IEEE 30th Int. Conf. Adv. Inf. Netw. Appl. Work. WAINA 2016*, pp. 731–736, 2016.
[26] H. Alizadeh, "Traffic Classification and Verification using Unsupervised Learning of Gaussian Mixture Models," *Meas. Netw. (M&N), 2015 IEEE Int. Work. (pp. 1-6). IEEE*, 2015.
[27] Trianggoro Wiradinata and P. Adi Suryaputra, "Clustering and Principal Feature Selection Impact for Internet Traffic Classification Using K-NN," *Proc. Second Int. Conf. Electr. Syst. Technol. Inf. 2015 (ICESTI 2015) (pp. 75-81). Springer Singapore.*, pp. 75–81, 2016.
[28] S. Sun, "A survey of multi-view machine learning," *Neural Comput. Appl.*, vol. 23, no. 7–8, pp. 2031–2038, 2013.
[29] J. Zhang, X. Chen, S. Member, Y. Xiang, and S. Member, "Robust Network Traf fi c Classi fi cation," *IEEE/ACM Trans. Netw. (TON), 23(4), pp.1257-1270*, pp. 1–14, 2014.
[30] R. Lin, O. Li, Q. Li, and Y. Liu, "Unknown network protocol classification method based on semi-supervised learning," *Comput. Commun.*, pp. 300–308, 2015.
[31] W. Lu and L. Xue, "A Heuristic-Based Co-clustering Algorithm for the Internet Traffic Classification," *2014 28th Int. Conf. Adv. Inf. Netw. Appl. Work.*, no. 5, pp. 49–54, 2014.
[32] R. Alshammari and A. N. Zincir-Heywood, "Identification of VoIP encrypted traffic using a machine learning approach," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 27, no. 1, pp. 77–92, 2015.
[33] R. Krzanowski, "Burst (of packets) and burstiness," *66th IETF Meet.*, 2006.
[34] "Top Sites in United Kingdom - Alexa." [Online]. Available: https://www.alexa.com/topsites/countries/GB. [Accessed: 16-Feb-2018].

7. Acknowledgements

This research was undertaken with the support of my sponsor (Iraqi cultural attaché).

