Robust Streaming Video Traffic Classification

Jordan Ebel, jebel@stanford.edu

I. INTRODUCTION

In order to provide reliable and fair Internet service, Internet service providers (ISP) use network management tools to classify, report, or restrict the Internet traffic of subscribers. Network management tools are critical to ensuring fairness and quality of service for all subscribers on a network, but the growth of streaming video, encrypted Internet protocols, and other traffic obscuring techniques are straining both networks and network management tools.

Streaming video is a bandwidth intensive class of Internet traffic responsible for an increasing percentage of total Internet usage [1]. Parallel to the growth of streaming video, protocols that encrypt Internet traffic are also growing in popularity [2]. In addition to encryption, consumers and video providers use web proxies, nonstandard ports, and other traffic obscuring techniques to attempt to avoid classification. Clearly, there is a need for high performance traffic classification systems to enable critical network management services, while also preserving consumers' expected levels of privacy, security, and performance.

Current research in the area of Internet traffic classification is mostly centered on deep packet inspection (DPI) and machine learning techniques. Current work involves using automata [3, 4] or Bloom filters [5] to maximize performance when matching header and payload contents to regular expressions. These techniques are accurate and very fast, yet they can be beaten by encrypting a packet's header and payload.

Machine learning techniques are another area of intense research. Williams, et al. in [6] provide a thorough summary of five supervised machine learning algorithms classifying traffic into multiple classes, and conclude good performance can be obtained using relatively simple features. The authors of [7] and [8] both tested unsupervised learning algorithms on classifying Internet traffic and achieved average performance. Li, et al. in [9] achieved excellent performance of 99% accuracy using the C4.5 decision tree algorithm. However, their model involved a very large amount of features, including many expensive to compute flow-based features. In addition, their classifier only worked on TCP packets, and their technique made use of port numbers as features. All of the related machine learning works studied supported a limit set of protocols, made use of port numbers, or involved very large feature sets. None of these works could be ideally applied to a real, modern classification system. Further, all of the related works studied have weaknesses in that the Internet is rapidly evolving, and many of the protocols and traffic patterns studied are now out of date.

This work aims to correct for the weaknesses found in related machine learning traffic classification systems. This work tests the algorithms naïve Bayes, logistic regression, support vector machines (SVM), and K-means clustering against the modern traffic patterns and protocols of the Internet. We achieved better than 95% accuracy using the SVM algorithm with a Gaussian kernel function, and implemented the SVM algorithm in a real time classification system to demonstrate the performance of the system. The input to the system is a flow of Internet packets, and the output is a classification for the packet as a streaming video packet or another packet. Our techniques are agnostic to the transport and application layer of the packets entirely, only making use of simple and easy to calculate features from the transport layer and below. As a result, the techniques presented in this paper are robust to modern traffic obfuscation techniques and are suitable for implementation in an actual classification system.

II. MODELS

We tested the following classification algorithms: (i) naïve Bayes, (ii) logistic regression, (iii) SVMs, and (iv) K-means clustering.

i. Naïve Bayes

The naïve Bayes classifier is a generative supervised learning classifier based on the naïve Bayes assumption and Bayesian probability. The naïve Bayes assumption assumes that given the class, any two features are conditionally independent. The naïve Bayes assumption can be expressed as:

$$p(x_1 \dots x_n|y) = \prod_{i=1}^{n} p(x_i|y)$$

The naïve Bayes assumption can be used to simplify the Bayesian probability, shown for a positive example:

$$p(y = 1|x) = \frac{p(x|y = 1)p(y = 1)}{p(x)}$$

$$p(y=1|x) = \frac{p(y=1)\prod_{i=1}^{n} p(x_i|y=1)}{p(x)}$$

A *maximum a posteriori* classifier is built by selecting the class that maximizes the Bayesian probability.

ii. Logistic Regression

Logistic regression is a discriminative supervised learning classifier that models the probability of the positive class given the input features, using the logistic function. The logistic function, applied to the linear combination of parameters and features, is:

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$$

To fit the parameters, logistic regression assumes the class outcome is the result of a Bernoulli trial and maximizes the log likelihood of the probability. The maximization problems does not result in a closed form expression for the parameters, so numerical optimization techniques such as gradient ascent or Newton's Method can be applied to solve for the parameters.

iii. SVM

Support vector machines are a non-probabilistic supervised learning classifier that finds the maximum margin hyperplane separating the input data in high dimensional space. The classifier will use the hyperplane to make predictions about new examples, based on which side of the hyperplane the example lies. Finding the hyperplane reduces to solving the optimization problem:

$$\min_{\gamma,w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i$$

Subject to the constraints:

$$y^i(w^T x^i + b) \geq 1 - \xi_i, i = 1, \ldots, m$$

$$\xi_i \geq 0, i = 1, \ldots, m$$

L1 regularization is used to allow mislabeled data, penalized by the cost parameter C. By taking the dual of this optimization problem, the algorithm can be expressed in terms of only inner products of features and parameters. This property can be exploited to map the input data into a higher dimensional feature space efficiently using a kernel function, allowing the SVM algorithm to learn in a higher dimensional space and perform nonlinear classification.

iv. K-Means Clustering

K-means clustering is an unsupervised learning classifier that attempts to label input data into k clusters. It is an iterative algorithm that assigns each example to the nearest cluster, then updates the cluster centroid. The examples are assigned to the nearest cluster according to:

$$c^i = arg\min_j \left\|x^i - \mu_j\right\|^2$$

This cluster assignment is based on the squared Euclidean distance, are therefore intuitively represents selecting the nearest cluster. It can be shown that K-means clustering is a variant of coordinate descent, so it is guaranteed to converge. However, the algorithm may converge to local optima, so multiple trials of the algorithm should be completed to find the best clustering result.

III. DATASET

Packet captures capturing streaming video traffic from a variety of video providers, mixed with standard web traffic, were collected on a personal computer. The packets were captured and visually inspected using the Wireshark [10] packet analyzer. Traffic was collected for approximately 60 seconds for each packet capture, resulting in about 80,000 packets in each capture. Both the training and test datasets contained approximately two-thirds streaming video traffic. Figure 1 below shows six streaming video packets collected in one of the training sets. The figure shows the arrival times, IP addresses, and protocol information for each packet.

51727	35.034829	173.194.26.109	192.168.1.252	TCP	1514 [TCP segment of a reassembled PDU]
51728	35.034829	23.246.14.169	192.168.1.252	TCP	1514 [TCP segment of a reassembled PDU]
51729	35.034830	23.246.14.169	192.168.1.252	TCP	1514 [TCP segment of a reassembled PDU]
51730	35.034832	23.246.14.169	192.168.1.252	TCP	1514 [TCP segment of a reassembled PDU]
51731	35.034832	23.246.14.169	192.168.1.252	TCP	1514 [TCP segment of a reassembled PDU]
51732	35.034833	23.246.14.169	192.168.1.252	TCP	1514 [TCP segment of a reassembled PDU]

Figure 1: Steaming video packets captured

Viewing the packet captures provided critical insights into the data that were exploited when extracting features. Streaming video traffic appeared in the datasets in bursts, with a large amount of very similar video packets arriving at nearly the same time. In comparison, standard web traffic was more irregular in characteristics and arrival time. These insights were very important to collect more descriptive features.

IV. FEATURES

The datasets were individually passed through a custom Python script to extract features. The Python script used the dpkt module [11] to examine packet contents and output a feature matrix and category matrix for the dataset. While port numbers and IP addresses were not used in feature extraction, they were used to determine the true packet categories. The technique employed to determine the true categories of packets is an area for further work and improvement. Video providers do not publish their IP address ranges, and packets from different providers varied widely in IP address, port number, and other characteristics. Some amount of packets was mislabeled, which therefore introduced a small amount of error into the system.

The initial set of features included features describing each packet individually. These features included IP length, IP time to live (TTL), IP protocol, and inter-arrival time. The initial set of features resulted in poor classification performance. In order to capture the burst-like nature of streaming video traffic, a window of sequential packets was examined. The mean and variance of characteristics of each packet in the window were collected and contributed significantly to algorithm performance. These features included the mean and variance of inter-arrival times, IP protocols, and IP lengths.

V. RESULTS AND DISCUSSION

Algorithm performance was measured using a combination of performance metrics: test error, training error, recall, precision, confusion matrices, and area under curves of both recall-precision and receiver operating characteristic curves. The quality of the features extracted was measured by visual inspection of the dataset in principle component space, and the percentage of total data variance that was captured by the first two principle components. High quality feature sets would show clear clusters of examples in principle component space, clear tradeoffs of features, and low amounts of total variance captured by the first two principle components.

Initially, a principle component analysis (PCA) was completed to gain insight into the dataset and the set of initial features. Figure 2 below is a biplot of a subset of the training dataset and the features in principle component space. The biplot shows a lack of clear clusters of the positive (streaming video) and negative (other traffic) examples in the data, pointing to a high bias issue. The principle component analysis revealed nearly 75% of the variance in the dataset was explained by the first two principle components. The biplot also revealed an overlap of closely related features, such as total packet length and IP packet length. Overlapping features do not provide useful information and were removed from future feature matrices.

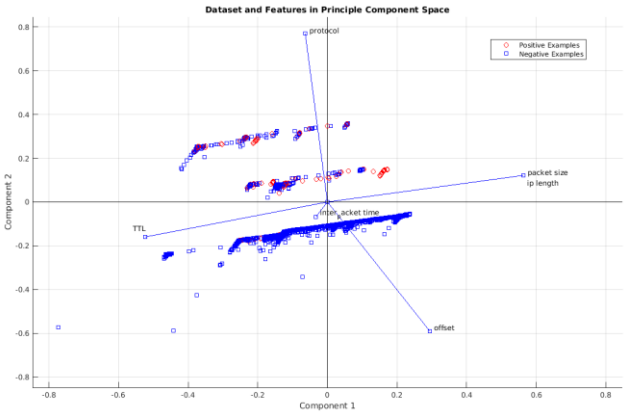

Figure 2: Biplot of training dataset

The performance of the supervised learning algorithms was tested using the limited set of features. Figure 3 below shows the test and training error of each supervised learning algorithm studied. All algorithms displayed subpar performance when the full training set was used. This plot also supports the high bias diagnoses as seen from the biplot, since each algorithm displayed unacceptable test and training performance that did not improve as more training examples were added.

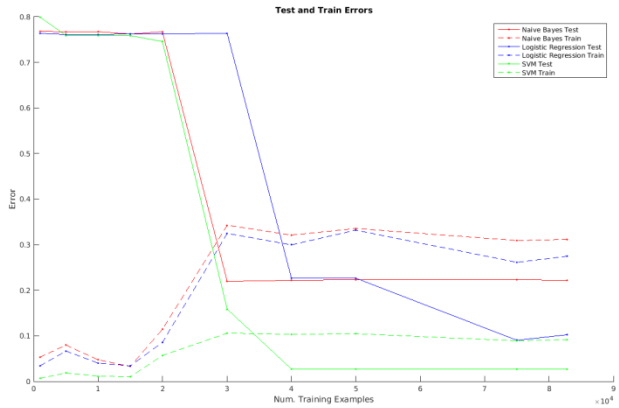

Figure 3: Test and training errors using limited feature set.

Figure 4 below is an updated biplot showing the dataset and features, including the additional features describing a window of packets. The biplot immediately reveals a much clearer grouping of positive examples and negative examples. As expected, the biplot shows related features, such as protocol and the mean protocol over a window of packets, as being closely related. The biplot also reveals interesting tradeoffs between packet size vs. TTL, and protocol vs. offset. These two tradeoffs run nearly parallel to the first and second principle components respectively, and therefore represent the primary sources of variances in the data. The tradeoff between protocol and offset served as a primary distinguishing characteristic between the positive and negative examples.

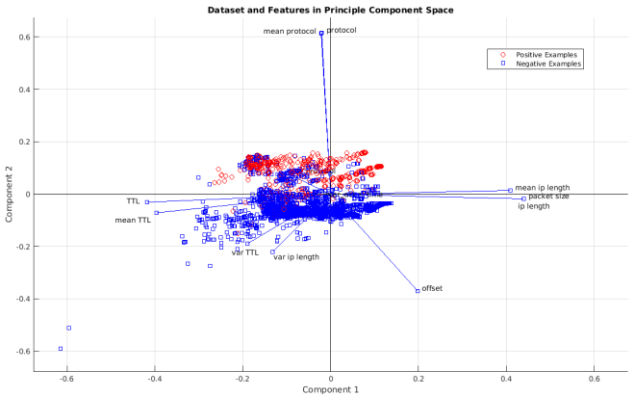

Figure 4: Biplot of training dataset with additional features

With the additional features collected, PCA revealed only about 50% of the variance of the dataset was explained by the first two principle components. The additional features significantly improved the variance of the dataset and correspondingly improved the performance of each algorithm.

Figure 5 below plots the test and training errors of each algorithm, including the additional set of features. The plot reveals each algorithm achieved substantial performance improvements when including the additional features. The test performance for each algorithm was especially improved, indicating progress was made treating the high bias issue. The SVM algorithm performed very well, with both test and training errors at acceptable limits.

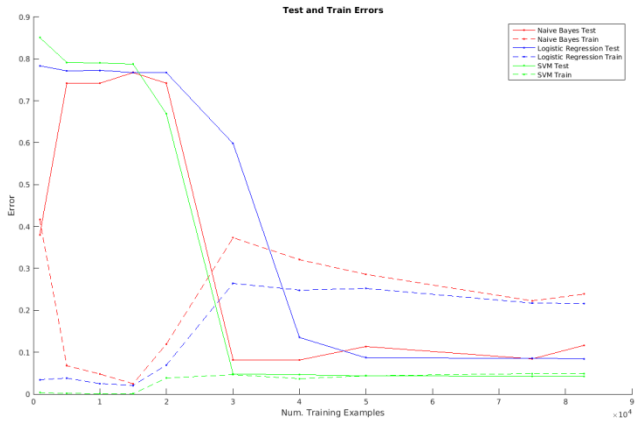

Figure 5: Test and training errors including additional features

To further optimize performance, an analysis was completed on the size of the window of packets. Four-fold cross validation was performed on the training dataset to determine the optimal number of packets to include in the window. Figure 6 depicts the cross validation error displayed by the SVM algorithm for a variety of window sizes. This test suggests a window size of three packets provided optimal performance for the SVM algorithm.

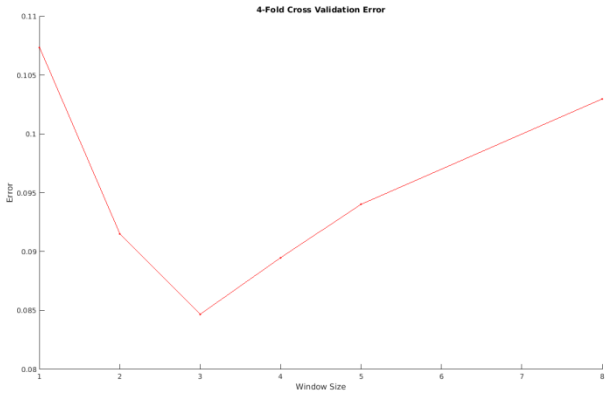

Figure 6: Cross validation error using different window sizes

The performance of the supervised learning algorithms was further tested by plotting receiver operating characteristic (ROC) and recall – precision curves. Figure 7 below shows the ROC curves achieved by each supervised learning algorithm. The SVM algorithm was the best performer, with area under the curve (AUC) of 0.9472, versus .8410 and .7730 for logistic regression and naïve Bayes, respectively. At a relatively low false positive rate of 10%, the SVM algorithm was able to achieve a perfect true positive rate, and therefore correctly identify every streaming video packet.

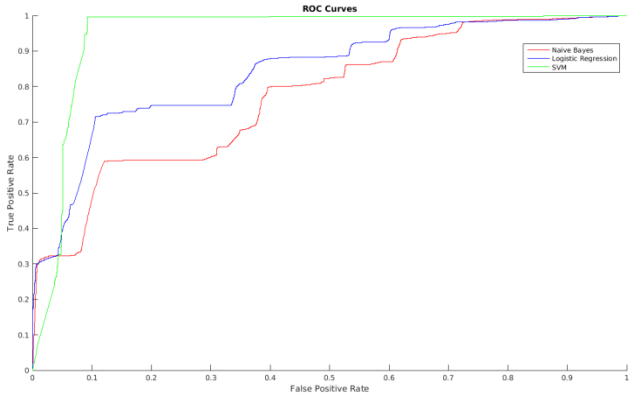

Figure 7: ROC curves of supervised algorithms

The recall-precision plot revealed a similar performance relationship. Figure 8 shows the SVM as the top performing algorithm, with AUC totals of 0.9420, versus 0.9054 for logistic regression and 0.5933 for naïve Bayes. SVM showed high precision across a broad range of recall rates, suggesting the SVM algorithm was more selective, even while classifying increasing amounts of packets as positive examples. The other two classifiers showed very good precision at low recall rates only, indicating they could selectively classify well only when limiting their overall number of positive examples predicted. Since the dataset contained more positive examples than negative examples, it is more useful for an algorithm to be precise even as the algorithm predicts more positive examples, as displayed by the SVM algorithm.

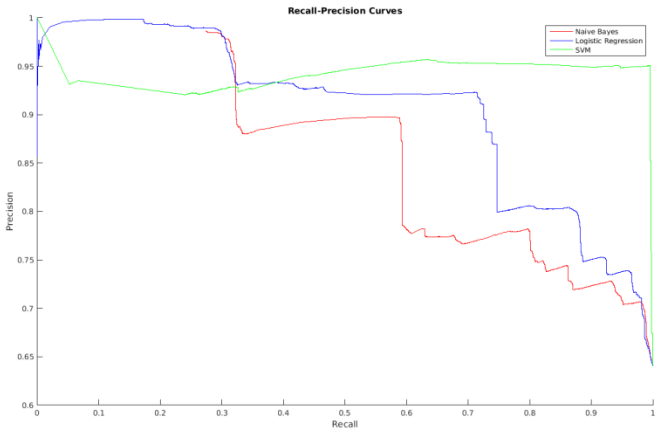

Figure 8: Recall-precision curves of supervised algorithms

The results from the plots were supported by the confusion matrices of the algorithms, shown in Figure 9. The confusion matrices show the naïve Bayes and logistic regression algorithms operating at very high recall, where they correctly classify most positive examples. However, they also incorrectly classify many negative examples as positive. Only

the SVM algorithm demonstrates good performance at a high level of recall. These results directly support the findings from the recall-precision curves above.

Naïve Bayes	Actual	
Predicted	Positive	Negative
Positive	0.9942	0.8712
Negative	0.0058	0.1288

Logistic Regression	Actual	
Predicted	Positive	Negative
Positive	0.9142	0.6574
Negative	0.0858	0.3426

SVM	Actual	
Predicted	Positive	Negative
Positive	0.9508	0.0212
Negative	0.0492	0.9788

Figure 9: Confusion matrices

The superior performance of the SVM algorithm can be attributed to the likely non-linearly separable property of the data. Logistic regression and naïve Bayes produce linear classifiers, whereas SVM with a Gaussian kernel, as was used in this analysis, results in a nonlinear classifier that is better suited for classifying complex, non-linearly separable data. Future improvements of this work include more experimentation with regularization of the data and the result on the performance of the linear classifiers. The test error curves above also show an interesting pattern: the naïve Bayes classifier initially outperformed the logistic regression classifier, until more training examples were included and logistic regression overtook naïve Bayes. This result is directly supported by research by Jordan and Ng, who found the same two regimes of performance [12].

Based on the results of the supervised learning classifiers, a real time system using the SVM algorithm with a Gaussian kernel was built to classify live traffic. The feature extraction functionality was combined with the scikit-learn [13] and scapy [14] modules to capture, extract features, and classify real time traffic. The system is shown operating on live traffic in Figure 10. Further work is required tuning this algorithm to match the performance of the SVM presented in this paper.

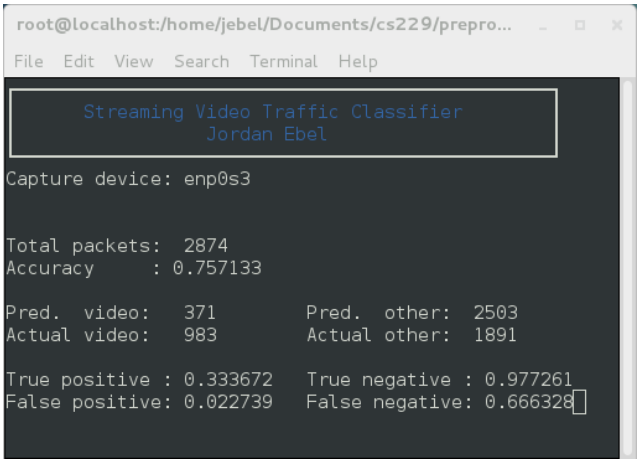

Figure 10: Real time traffic classifier in operation

To gain additional insight into the data, the K-means clustering unsupervised learning algorithm was tested to sort the data into two classes. A plot of the output of the clustering algorithm in principle component space is shown in Figure 11. The plot reveals a relatively clear grouping of the data into actual classes (indicated by color), yet the clustering algorithm failed to accurately group the examples into their respective classes (indicated by marker type). Since K-means clustering assigns examples to the nearest cluster center, it expects clusters to be similar in size. This expectation was not true in the dataset tested. Further work should be completed in preprocessing the dataset and running additional diagnostic tests to tune the clustering performance.

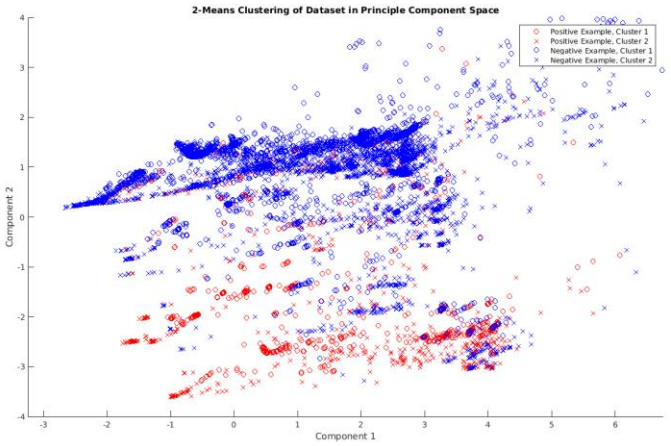

Figure 11: Clustering in principle component space

VI. CONCLUSION

Classifying traffic is a critical capability that enables effective Internet network management. There is a need for modern network management tools that effectively deal with growing bandwidth due to streaming video and the increasing prevalence of traffic obfuscation techniques. This work detailed the use of PCA to analyze and build a rich feature set describing packet captures. The performance of classification algorithms naïve Bayes, logistic regression, SVMs, and K-means clustering were tested. The SVM algorithm was the top performing algorithm, and was implemented as a part of a real time traffic classification system. The system was agnostic to the transport and application layers of packets, and was therefore robust to traffic obfuscation techniques. The system fits the requirements of an effective, practical, and efficient modern Internet traffic classifier.

VII. REFERENCES

[1] Lopes, Marina (2014, June 10). "Videos may make up 84 percent of internet traffic by 2018: Cisco." *Reuters*. Available at: http://www.reuters.com/article/2014/06/10/us-internet-consumers-cisco-systems-idUSKBN0EL15E20140610

[2] Finley, Klint (2014, May 16). "Encrypted Web Traffic More Than Doubles After NSA Revelations." *Wired Magazine*. Available at: http://www.wired.com/2014/05/sandvine-report/

[3] S. Kumar, J. Turner, et al. Advanced Algorithms for Fast and Scalable Deep Packet Inspection. Presented at ACM/IEEE Symposium, 2006. [Online]. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.475.5261&rep=rep1&type=pdf

[4] F. Yu, Z. Chen, et al. Fast and Memory-Efficient Regular Expression Matching for Deep Packet Inspection. Presented at ACM/IEEE Symposium, 2006. [Online]. Available at: http://www.diku.dk/hjemmesider/ansatte/henglein/papers/yu2008.pdf

[5] S. Dharmapurikar, P.Krishnamurthy, et al. Deep Packet Inspection using Parallel Bloom Filters. Presented at 11th Symposium on High Performance Interconnects, 2003. [Online]. Available at: http://www.arl.wustl.edu/~todd/hoti.pdf

[6] N. Williams, S.Zander, et al. A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification. *ACM SIGCOMM Computer Communication Review*, 2006. [Online]. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.108.247&rep=rep1&type=pdf

[7] S. Zander, T. Nguyen, et al. Automated traffic classification and application identification using machine learning. Presented at The IEEE Conference on Local Computer Networks, 2005. [Online]. Available at: http://security.riit.tsinghua.edu.cn/mediawiki/images/8/8e/CLCN2005_Automated_Traffic_Classification_and_Application_Identification_Using_Machine_Learning.pdf

[8] J. Erman, M. Arlitt, et al. Traffic Classification Using Clustering Algorithms. Presented at 2nd Annual ACM Workshop on Mining Network Data, 2006. [Online]. Available at: http://www.ce.uniroma2.it/courses/MMI/memopaper2.pdf

[9] W. Li and A. W. Moore. A Machine Learning Approach for Efficient Traffic Classification. Presented at 15th International Symposium on MASCOTS, 2007. [Online]. Available at: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&tp=&arnumber=4674432&contentType=Conference+Publications&searchField%3DSearch_All%26queryText%3Dtraffic+classification+C4.5

[10] See https://www.wireshark.org/

[11] See https://dpkt.readthedocs.org/en/latest/

[12] A. Jordan, A. Ng. On discriminative vs. generative classifiers: A comparison of logistic regression and naïve bayes. *Advances in neural information processing systems*, 2002. [Online]. Available at: http://papers.nips.cc/paper/2020-on-discriminative-vs-generative-classifiers-a-comparison-of-logistic-regression-and-naive-bayes.pdf

[13] See http://scikit-learn.org/stable/index.html

[14] See http://secdev.org/projects/scapy/