Research Result

Machine Learning For Internet Traffic Identification

R. Pitchandi[1], P. Bhavani[2]

[1]*Associate Professor and Head, Department of MCA, Madha Engineering College, Chennai-69*
[2]*Final Year MCA Student, Department of MCA, Madha Engineering College, Chennai-69*

ABSTRACT

This research explores the classification of Internet traffic using machine learning techniques. As the landscape of Internet applications evolves, traditional methods of traffic classification based on port numbers and payload analysis prove less effective. In response, this study employs unsupervised clustering through the AutoClass algorithm and compares it with the supervised Naive Bayes classifier. Both methods utilize flow statistics obtained from TCP/IP headers, avoiding privacy and encryption issues. Experimental results on real-world datasets demonstrate that the AutoClass algorithm achieves an average accuracy of over 90%, outperforming the Naive Bayes classifier by up to 9%. Additionally, the unsupervised approach offers efficiency advantages by clustering connections and expediting classification. This research underscores the potential of unsupervised machine learning for effective Internet traffic analysis, with implications for network management and security.

KEYWORDS

Internet Traffic, Machine Learning, Traffic Classification, Unsupervised Learning, Naive Bayes Classifier, Auto Class Algorithm, Network Security, Network Management, TCP/IP Headers, Flow Statistics, Port-based Identification, Payload Analysis, Encryption, Privacy Concerns, Peer-to-Peer (P2P) Traffic, Online Gaming, Traffic Analysis, Data Privacy, Traffic Cluster Analysis, Classification Accuracy

1. INTRODUCTION

Many areas, including network architecture, network anagement, and network security, depend on accurate classification of Internet traffic. Adapting to the dynamic nature of Internet traffic is one of the main challenges in this area. On the Internet, new applications are being released more frequently; some of these new applications, including peer-to-peer (P2P) file sharing and online gaming, are growing in popularity. Traditional classification techniques, like those based on well-known port numbers or packet payload analysis, are no longer effective for all types of network traffic or are otherwise impractical to use due to data privacy or security concerns. This is because Internet traffic has evolved, both in terms of the number of applications and the types of applications that are used.

Traffic classification using machine learning techniques is a promising method that has lately drawn some attention. These methods presuppose that the apps predictably deliver data most of the time; these predictable patterns can be utilized as a means of identification, allowing the connections to be divided into traffic classes. Flow statistics (such as mean packet size, flow length, and total number of packets) that are only available using TCP/IP headers are required to discover these patterns. As a result, the classification algorithm can classify data without using port numbers or packet payload information.

In this study, we use an unsupervised learning method (EM clustering) for the classification of Internet traffic and compare the outcomes to those of a supervised machine learning technique that was previously used. Unlabeled training data is divided into groupings termed "clusters" based on similarity in the unsupervised clustering approach, which uses an Expectation Maximization (EM) algorithm.

Prior research has demonstrated the excellent accuracy of the Nave Bayes classifier for categorizing Internet traffic [2]. Zander et alparallel.'s study focuses on developing the classification model using the EM clustering method [4]. By creating a classifier utilizing the EM clustering method, we add to their work and demonstrate that it performs better than the Naive Bayes classifier in terms of classification accuracy. As a function of the size of the training data set, we also examine how long it takes to develop the classification models for each strategy. Additionally, we investigate the clusters identified by the EM technique and discover that the bulk of connections are in a smaller subset of all clusters.

2. BACKGROUND AND RELATED WORK

Recent years have seen a lot of progress in the study of traffic classification. The many methods discussed in the literature will be surveyed in this section.

A. Analysis of Port Numbers

In the past, methods for classifying traffic relied on well-known port numbers to identify Internet traffic. The use of fixed port numbers assigned by IANA by many conventional applications made this successful [6]. For instance, port 25 is frequently used by email apps. For some applications, such as the current generation of P2P applications that purposefully try to mask their traffic by using dynamic port numbers or masquerade as well-known programs, this strategy has been demonstrated to be ineffective by Karagiannis et al. in [7]. Furthermore, only programs whose port numbers are known in advance can be recognized.

B. Payload-based Evaluation

The examination of packet payloads [7]–[10] is a further well-researched strategy. The packet payloads are examined in this method to determine whether or not they include the distinctive signatures of well-known programs. These strategies have proven to be particularly effective for Internet traffic, especially P2P traffic. These methods do, however, have some limitations. Payload analysis raises issues with security and privacy first. Second, these methods often need more processing power and storage space. Third, encrypted transmissions cannot be handled by these methods. Furthermore, these methods are unable to categorize previously unidentified traffic and can only detect traffic for which signatures are known.

C. Heuristics for the transport layer

To address the shortcomings of payload-based analysis and the declining efficacy of port-based identification, transport-layer heuristic information has been utilized. To identify this traffic, Karagiannis et al. [7] present a novel method that takes advantage of the distinctive characteristics of P2P apps when they are transferring data or establishing connections. It has been demonstrated that this strategy outperforms port-based classification and is comparable to payload-based analysis. Additionally, Karagiannis et al. developed a different approach that leverages social, functional, and application characteristics to distinguish between different types of traffic [11].

D. Techniques for Machine Learning

The two main components of machine learning approaches are model development and classification. Initially, a model is created using training data. This model is then sent into a classifier, which categorizes a batch of data.

Unsupervised and supervised machine learning approaches can be separated into these categories. According to the McGregor Et alhypothesis.s it is possible to categorize traffic using an unsupervised method based on connection-level information (i.e., transport layer statistics) [1]. This method makes use of an EM algorithm [5], and McGregor et al. conclude that this strategy is promising. By extending this work in [3] and [4], Zander et al. identify the ideal collection of attributes to incorporate into the classification model by utilizing an EM method named AutoClass [12].

Some supervised machine learning methods, like [13] and [2], also classify traffic based on connection-level statistics. Nearest neighbor and linear discriminate analysis are used by Roughan et al. in [13]. This method has limitations because it does not categorize HTTP traffic and only employs a small set of connection-level information. Moore et al. recommend using Nave Bayes as a classifier in [2] and demonstrate that this method has a high level of classification accuracy for traffic.

3. CLASSIFICATION BY MACHINE LEARNING

Flow statistics are used to categorize Internet traffic in both of the methods examined in this research. For both approaches, the classification models (also known as classifiers) are constructed using this relationship information. An outline of the machine learning methods applied in this work is provided in this section.

A. Approach to supervised machine learning

In this paper, supervised machine learning is carried out using the Nave Bayes classifier. Moore et al. applied the Na ve Bayes classifier and discovered that this method has good accuracy for categorizing Internet traffic [2], presuming that flow attributes are independently and uniformly distributed. The interested reader is directed to [2] for more information after we give an outline of this strategy here.

Based on labeled training data, the Nave Bayes technique calculates the Gaussian distribution of the attributes for each class. Based on the conditional probability of the connection falling into a class given its attribute values, a new connection is categorized.

The Bayes rule is used to assess the likelihood that each attribute belongs to the class:

A is a predetermined class and B is an attribute's fixed value. The likelihood that an object belongs to a specific class A is calculated by multiplying these conditional probabilities together. In this study, we utilized the WEKA software package version 3.4's Nave Bayes implementation [14]. Moore et al. also utilized this software suite for their analysis [2].

B. Approach to unsupervised machine learning

The foundation of the unsupervised machine learning method is a classifier constructed from clusters discovered and identified in a training set of data. Once the classifier has been created, the classification procedure entails the classifier determining which cluster a connection is closest to and then identifying that connection using the label from that cluster.

1) Clustering method: The clustering procedure identifies the groupings in a training set. This task, which divides objects into groups based on similarity, is unsupervised because the algorithm does not know the true classes beforehand. High intra-cluster similarity and high inter-cluster dissimilarity are characteristics of a good set of clusters.

To extract the set of clusters that are most likely to exist from the training data, we employ an implementation of the EM clustering algorithm dubbed AutoClass [12]. A finite mixture model of the attribute values for the cluster's members' objects is used by AutoClass to determine the likelihood that an item will belong to each discrete cluster. This presumption asserts that all attribute values are conditionally independent and that any similarity in attribute values between two objects results from their shared class.

The finite mixture model's parameters for each cluster are unknown when this algorithm is first run. An expectation step plus a maximization step make up the EM algorithm. Pseudorandom numbers are used in the initial expectation stage to make educated predictions about the parameters. The parameters are then continuously estimated during the maximizing step until they reach a local maximum using the mean and variance. After recording these local maxima, the EM procedure is carried out once more. Up till sufficient samples of the parameters have been found, this process continues (we use 200 cycles in our experimental results). The intra-cluster similarity and inter-cluster dissimilarity are used to choose the optimum set of parameters.

2) ***Using the Results of Clustering as a Classifier:*** A transductive classifier is used to convert the clustering into a classifier once an acceptable clustering has been discovered using the connections in a training data set [15]. This method labels the clusters before classifying a new object with the label of the cluster to which it is most related.

The most prevalent traffic category among the links in a cluster was used to identify it. When two or more categories are tied, one of the tied category labels is picked at random. The traffic class name of the cluster to which a new connection is most comparable is then used to categorize it.

Experimental outcomes

The effectiveness of the Nave Bayes and AutoClass algorithms is assessed in this section. The data sets that were used in this investigation are first described. The parameters for gauging the techniques' efficacy are then presented. The experiment's findings are then displayed.

Sets of data

In this work, data from two publicly accessible traces are used. The traffic passing through the University of Auckland's Internet infrastructure is depicted in both traces. Only a subset of each trace is used because of the traces' huge size (Auckland IV and Auckland VI [16]). The traffic from the Auckland IV trace that was monitored from March 16, 2001, at 6:00 AM, to March 19, 2001, at 05:59 PM makes up the Auck-IV sub data set. The Auckland VI trace's Auck-VI sub-data set, which was used in this study, is a subset from June 8, 2001, at 06:00:00, to June 9, 2001, at 05:59:59.

Connection Identification: It is vital to identify the flows inside the traces to get the statistical flow data required for the testing. These packet exchanges, also known as flows or connections, take place in both directions between two nodes. These two nodes can be recognized by their transport layer port numbers and IP addresses, which remain consistent during the connection.

Data from connection-oriented transport layer protocols is present in both traces, although not solely (e.g., TCP). Unconnection-oriented protocols like UDP and ICMP are the source of some traffic. Although some connection-related statistics may be gathered for these, we excluded connection-less traffic from our data sets since our main focus was on TCP-using applications.

SYN/FIN packets that identify connections can be traced back to the TCP/IP header information that was collected for the packets in both traces. Sending SYN and FIN packets initiates and ends a connection, respectively. A connection that had been open for more than 60 seconds without a packet being delivered or received between the nodes was likewise shut down.

Following the discovery of a connection, the following statistical flow characteristics are determined: total number of packets, mean packet size (both in each direction and when combined), mean data packet size, flow duration, and mean inter-arrival time of packets. Our choice to employ these traits was largely influenced by earlier research by Zander et al. [3]. We discovered that the logarithms of the characteristics offer considerably better results utilizing both approaches because many of the attributes have heavy-tail distributions.

Pre-classification of the Test Data Sets is required to validate the outcomes of the Algorithms (i.e., a true classification is needed). There is no payload-based identification method that can be utilized to identify the genuine classes because the traces are public and only contain TCP/IP header information. To identify users, ports are utilized. We believe that port-based identification should still deliver reliable findings for the traces utilized in this work, despite its decreasing efficacy. This is because the P2P traffic's development of dynamic port numbers did not occur until late 2002 [18]; the Auckland traces were gathered in 2001.

For the Auck-IV sub-data set, Table I provides summary statistics of the traffic classes together with the identifying port numbers. All connections with a target port of 80, 8080, or 443 include HTTP data. Because it functions similarly to unencrypted HTTP at the connection level, port 443 holding encrypted HTTP data was included. This makes it possible to distinguish between packets that are encrypted and those that aren't when they come from the same class or application.

The results of computing the number of connections corresponding to each class revealed that HTTP traffic constituted the majority of connections in the two data sets. Except for HTTP, the data sets' high HTTP traffic volume does not test the methods for classifying traffic properly. As a result, 1000 randomly selected connections from each traffic class were included in equal amounts in the training and testing data sets to permit a fair analysis. This makes it possible to accurately compare the accuracy obtained in the test results and evaluate the capacity of both machine learning approaches to categorize not just HTTP traffic but also other types of traffic.

Effectiveness Standards

Three metrics—precision, recall, and overall accuracy—were used to assess the algorithms' efficacy. These metrics are frequently employed in the literature on data mining to assess data clustering techniques [14]. The quantity of correctly identified items in a class is known as the True Positives. The term "False Positives" describes the quantity of objects that have been incorrectly classified as a class. False Negatives are the number of objects belonging to one class that are mistakenly classified as belonging to a different class.

A measure of precision is the proportion of True Positives to True and False Positives. This establishes the quantity of identified objects.

The majority of the test data sets show that the Nave Bayes classifier performs well overall.

The total of all True Positives to the total of all True and False Positives for all classes is used to measure overall accuracy. This gauges the classifier's general accuracy. Keep in mind that recall and precision are per-class measurements.

where the number of classes is n. There is a connection between recall and precision. As a result of the algorithms used to categorize the objects into classes, if the recall for one class is lower, this will also result in poorer precision for other classes. Additionally, because it calculates the average precision across all classes, the total accuracy and precision are related.

Results of the Nave Bayes Classifier

The Nave Bayes classifier is initially trained for each data set using a training set made up of 1000 random samples from each traffic class. The classifier is then tested to assess how successfully it categorizes 10 different test sets consisting of 1000 (different) random samples of each traffic class after this training is finished. The efficacy criteria are computed based on the classification of the test set. Figure 1 displays the minimum, maximum, and average precision and recall results for the Auck-IV sub-data set. The Nave Bayes results utilizing the Auck-VI sub-data set resemble the Auck-VI sub-data set in terms of quality (These results are not shown due to space limitations.).

The precision and recall for six out of the nine classes were, on average, above 80%, according to an examination of the Auck-IV sub-data set results. It fared best for IRC connections, with 95.0 percent precision and 94.5 percent recall, and then for POP3 connections, achieving 87.2 percent precision and 88.6 percent recall. With precisions of 69.7% and 73.4 percent, respectively, it performed the poorest for SOCKS and LIMEWIRE connections. The subpar performance is caused by 10% of LIMEWIRE and FTP data transfers being incorrectly categorized as SOCKS, which lowers their recall levels. The main traffic types for LIMEWIRE that were incorrectly categorized were HTTP and SOCKS.

Results from AutoClass

Results for the unsupervised machine learning method utilizing AutoClass are presented in this section. In this method, the training set of data is first grouped using AutoClass to create groups of objects that are similar to one another for each of the data sets. Utilizing the previously mentioned technique, a transudative classifier is then constructed using these clusters. From the 10 test sets of data, the resulting classifier is then used to forecast which traffic class a new connection will fall into. Results for the Auck-IV sub-data set's minimum, maximum, and average precision and recall are shown in Figure 2.

Figure 2 demonstrates that the precision and recall values are, on average, substantially greater than those produced using the Nave Bayes approach. All classes in Figure 2 have precision and recall values that are more than 80%. Take note that seven out of the nine classes have average recall values that are higher than 90%, and six out of the nine classes have average precision values that are higher than 90%. Despite their poor classification, HTTP and LIMEWIRE still have precision and recall rates of around 80%. As a result of about 10% of SOCKS connections being misclassified as HTTP, HTTP had a lower degree of precision. Because HTTP was wrongly categorized as LIMEWIRE, the LIMEWIRE classification accuracy was low.

Each of the AutoClass-generated clusters was examined in detail. This sheds light on the reasons why some connections are misclassified. We looked at one of the clusters, for instance, where HTTP was mistakenly labeled as LIMEWIRE. 37 connections (or 33 percent) were HTTP and 66 were LIMEWIRE in this cluster of 111 connections (59 percent). For all connections in this cluster, a total of 12 packets were transmitted. Each connection had an average packet size of 106 bytes, with HTTP connections averaging 118 bytes and LIMEWIRE connections 101 bytes. With HTTP taking 0.7 and LimeWire 0.3 seconds, respectively, the average duration was 0.5 seconds. The Auck-VI sub-data set's results for AutoClass are qualitatively comparable to those of the Auck-IV sub-data set.

With precision and recall values for both data sets averaging about 91 percent, the AutoClass technique performs pretty well overall for the data sets.

Algorithms' General Accuracy

The comparison of the Nave Bayes classifier and the AutoClass approach's overall accuracy is shown in Table II. The Nave Bayes classifier has an overall accuracy of 82.5 percent in comparison to AutoClass's average accuracy of 91.2 percent in the Auck-IV sub-data set. As a result, for this collection of data, AutoClass surpasses the Nave Bayes classifier by 9%. This demonstrates that the unsupervised machine learning strategy, which does not require the training data to be labeled beforehand, is at least as effective as the supervised learning approach.

4. DISCUSSION

While the unsupervised cluster technique has better accuracy than the Nave Bayes classifier, we demonstrated in the previous section that both performed well at categorizing the connections. Both algorithms have certain clear advantages over payload-based methods. Because the private information contained in packets is not analyzed, non-payload-based techniques have fewer privacy concerns to take into account, as described in Karagiannis et al. [7]. Because there is less data to be processed while merely dealing with packet headers, there is less storage and processing overhead. Last but not least, the encryption of payloads won't hinder these approaches. The reliance on the training data being representative of the total network traffic is a drawback for both algorithms, though. Retraining of the classifiers is necessary if the training data is no longer representative.

Since the training data do not need to be labeled, the unsupervised cluster technique has certain additional advantages. By looking at the connections that are gathered to form a cluster, for instance, new applications can be found. Usually, clusters are built to support a single application. Consequently, just a small portion of the connections between each cluster's components are present. This could result in significant time savings for the operator of this strategy because the training set's manual classification could take some time.

Runtime Evaluation

Due to the computationally intensive nature of the model creation phase, the runtime of both approaches must be carefully considered. On a Dell OptiPlex GX620 with an Intel Pentium IV 3.4 GHz processor and 1 GB of RAM, all operations for the analysis are carried out. The training set contained anything between 1000 and 128000 data items. When creating the classification models, the Naive Bayes classifier generally took much less time than AutoClass. For instance, AutoClass built the classification model in 2070 seconds as opposed to 0.06 seconds for Nave Bayes with 8000 objects. With an increase in the quantity of objects, both strategies show a linear development pattern. Even though the naive Bayes classifier was quicker, the size of the training set is ultimately constrained by the memory available because both methods must load the whole training set into memory before creating the model.

A cluster's individual AutoClass weight

As would be expected, some of the AutoClass-identified clusters have a lot more connections than others. The number of connections in each of the clusters created by five of the Auck-IV sub-training data sets is examined in this section. This analysis is helpful because it shows that if the AutoClass approach was employed, it would be possible to use fewer clusters without having to identify the traffic class that each cluster relates to.

For five of the Auck-IV sub-datasets, the CDF graphs in Figure 3 display the overall number of connections as a function of the number of clusters. There were on average 123 clusters in the Auck-IV sub-data set. According to this graph, only 2% of all connections are represented by the last 20% of the clusters that were formed. Finding these clusters won't greatly affect how accurate the clustering is in general. These figures also demonstrate that 50% of the clusters can represent 80% of the connections. This means that just half of the clusters need to be examined to determine which traffic class each cluster belongs to identify 80% of the connections and produce the transductive classifier.

5. CONCLUSION

An unsupervised machine learning method (AutoClass) for categorizing Internet traffic was presented in this paper. We compared this strategy to a supervised machine learning approach (Naive Bayes classifier) using qualitative and quantitative results. Our findings demonstrate that AutoClass can attain an average accuracy of more than 90%. We discover that AutoClass outperforms Naive Bayes by up to 9% for the data sets examined in this research.

We also found that the unsupervised clustering technique can speed up the classification of connections. Because just a subset of the connections in each cluster must be manually identified, time can be saved. Not every cluster is required for results to be reasonably accurate. Overall, although significantly lowering the amount of manual configuration, the unsupervised machine learning strategy produced superior outcomes and may be said to be at least as effective. This outcome is quite encouraging. By categorizing connections into groups that can be quickly used to identify the apps providing the data, this method has the potential to become an excellent tool for investigating network traffic in the future.

This work is being pursued in various areas. Applying the unsupervised clustering method to a more current trace that might include peer-to-peer and streaming video traffic is the next thing we need to do. In this study, the only clustering technique used was autoclass, which is based on Bayesian classification theory. There are other additional clustering techniques in the data mining literature [19] that are based on various ideas and methodologies. We are investigating a few of these distinctive clustering techniques right now; the outcomes of our initial research are presented in [20].

REFERENCES

[1] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, "Flow Clustering Using Machine Learning Techniques," in PAM 2004, Antibes Juan-lesPins, France, April 19-20, 2004.

[2] A. Moore and D. Zuev, "Internet Traffic Classification Using Bayesian Analysis Techniques," in SIGMETRICS'05, Banff, Canada, June 6-10, 2005.

[3] S. Zander, T. Nguyen, and G. Armitage, "Self-Learning IP Traffic Classification Based on Statistical Flow Characteristics," in PAM 2005, Boston, USA, March 31-April 1, 2005.

[4] "Automated Traffic Classification and Application Identification using Machine Learning," in LCN'05, Sydney, Australia, November 15- 17, 2005.

[5] A. Dempster, N. Paird, and D. Rubin, "Maximum likelihood from incomeplete data via the EM algorithm," Journal of the Royal Statistical Society, vol. 39, no. 1, pp. 1–38, 1977.

[6] IANA. Internet Assigned Numbers Authority (IANA), "http://www.iana.org/assignments/port-numbers."

[7] T. Karagiannis, A. Broido, M. Faloutsos, and K. claffy, "Transport Layer Identification of P2P Traffic," in IMC'04, Taormina, Italy, October 25- 27, 2004.

[8] P. Haffner, S. Sen, O. Spatscheck, and D. Wang, "ACAS: Automated Construction of Application Signatures," in SIGCOMM'05 Workshops, Philadelphia, USA, August 22-26, 2005.

[9] A. Moore and K. Papagiannaki, "Toward the Accurate Identification of Network Applications," in PAM 2005, Boston, USA, March 31-April 1, 2005.

[10] S. Sen, O. Spatscheck, and D. Wang, "Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures," in WWW2005, New York, USA, May 17-22, 2004.

[11] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINK: Multilevel Traffic Classification in the Dark," in SIGCOMM'05, Philadelphia, USA, August 21-26, 2005.

[12] P. Cheeseman and J. Strutz, "Bayesian Classification (AutoClass): Theory and Results." In Advances in Knowledge Discovery and Data Mining, AAI/MIT Press, USA, 1996.

[13] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield, "Class-of-Service Mapping for QoS: A Statistical Signature-based Approach to IP Traffic Classification," in IMC'04, Taormina, Italy, October 25-27, 2004.

[14] I. Witten and E. Frank, (2005) Data Mining: Pratical Machine Learning Tools and Techniques, 2nd ed. San Francisco: Morgan Kaufmann, 2005.

[15] A. Banerjee and J. Langford, "An Objective Evaluation of Criterion for Clustering," in KDD'04, Seattle, USA, August 22-25, 2004.

[16] Auckland Data Sets, "http://www.wand.net.nz/wand/wits/auck/."

[17] V. Paxson, "Empirically-Derived Analytic Models of Wide-Area TCP Connections," IEEE/ACM Transactions on Networking, vol. 2, no. 4, pp. 316–336, August 1998.

[18] C. Colman, "What to do about P2P?" Network Computing Magazine, vol. 12, no. 6, 2003.

[19] A. K. Jain and R. C. Dubes, Algorithms for Clustering Data. Englewood Cliffs, USA: Prentice Hall, 1988.

[20] J. Erman, M. Arlitt, and A. Mahanti, “Traffic Classification using Clustering Algorithms,” in SIGCOMM’06 MineNet Workshop, Pisa, Italy, September 2000.