Feature Extraction & Modeling Methodology for Next Generation Network Traffic

PIs: *Mo-Yuen Chow* *Arne Nilsson* *H. Joel Trussell*

Department of Electrical and Computer Engineering
NC State University
Raleigh, NC 27695-7911
Email: {nilsson, chow, hjt}@eos.ncsu.edu

ABSTRACT

Traffic classification is a fundamental problem in achieving differentiated services for Next Generation Internet. In this work we propose a statistically based scheme to classify Internet traffic flows over a time frame, based on the application to which the packets belong. The scheme uses statistical information and thus it does not involve reading any packet headers to determine the application. This 'implicit' classification builds a foundation for estimation and prediction of traffic mix, which is a long-term goal of this research project. The classification results mentioned in this paper indicate that this approach is promising.

I. INTRODUCTION AND MOTIVATION

Over the last few years, traffic on the Internet has proliferated, both in terms of amount of traffic, and in variety of applications. The introduction of voice, video and other real-time applications has changed the way the Internet is used. This has triggered the need for a change in traffic handling on the Internet. In particular, there is increasing demand for service differentiation. The Diffserv architecture[8] of the IETF is one such step towards fulfilling this demand. However, for any such service, the very basic problem one encounters is that of classification. The traffic on the Internet can be classified using various parameters, viz. source and/or destination IP address (or prefix), type-of-service, application, etc. A number of classification schemes are discussed in [1][5][6][7]. In this work, we classify the Internet traffic based on the application to which the packets belong. An easy approach to do so is by extracting the port number information from the layer 4 (TCP/UDP) header [4]. However, there are certain problems with this approach. With the introduction of the Network Address Port Translation (NAPT) concept, the port numbers are no longer a trustworthy source for determining the application type. In a free environment like the Internet, it is not mandatory for applications to use specific port numbers[2]. It is very likely that other applications would spoof port numbers in order to gain better service. Also, future network security algorithms might encrypt even the packet headers, along with the data contained in the packet. These bottlenecks motivate our approach in classifying and modeling network traffic. In this paper, we classify IP traffic into different application types on the basis of packet attributes obtainable at layer 3 (i.e. IP layer). These attributes are intrinsic characteristics of the packet itself. Also, it should be noted that we do not classify the traffic on a per-packet basis, rather the traffic flows are classified.

The long term goal of this research project is to estimate the contents of a traffic mix in a given time frame without reading the layer 4 header, and from it predict the traffic composition for successive times. The steps taken to solve the estimation and prediction problems start with determining the characteristics of applications. In this paper, we discuss only the classification methodology that we have used.

2. DATA USED IN ANALYSIS

Our requirement was to use packet attributes that cannot be spoofed by the source. The ones that we considered using were the source and destination IP address pairs, packet size and inter-arrival time. But as our goal was to classify the traffic based on the application, the IP address-pair option was dropped and we decided on using packet size and inter-arrival time.

The second question was that of the level at which to classify. We decided to classify the flows, i.e. classify the traffic over a time frame. This allows the use of statistical data that is obtained over the time frame. It is impossible to classify individual packets based on only packet size and inter-arrival time.

The data for this project is collected from the North Carolina State University backbone network using TCPDUMP software. The overall details of the data are as below:

- Each file has approximately 500,000 packets, representing varying time durations - the minimum 118 s and maximum 999 s. The average over is 372 s.
- The data is divided into two sets: Training set, and Test set.
- Each of these 2 sets has 8 sub-sets.
- Each of the 16(8x2) sub-sets contain 3 files, similar to the ones described above. The 3 files are obtained with an average time gap of 15 minutes between each of them.
- Each of the 8 data sets is selected with average time intervals of 968 minutes between them (approximately 16 hrs).

The classes into which we will categorize are the applications. We calculated the contribution of each application in the overall traffic mix across the entire training set. Based on these results, we decided to classify the traffic into 7 classes, viz. dns, ftp, http, nntp, reserved, smtp, telnet. The result is given in Table 1. Here, a packet is considered to be belonging to an application if the source and/or destination port number in the packet header is of that application.

We see that the results are quite similar to the traffic on the Internet backbone [3]. Once a successful classification scheme is devised, the same can be extended to include more classes.

This work was supported by a grant from the Center for Advanced Computing and Communication

Hence, if real-time traffic, i.e. voice and video, is treated as a separate class, it would be possible to differentiate it from other data traffic.

3. CLASSIFICATION

As mentioned before, our approach is to classify over a period of time. We studied the distribution of the features we selected, i.e. packet size and inter-arrival time, over periods of time spanning one sub-set. One approach was to use clustering methods to determine if the data would fall naturally into distinct sets, then use these sets for the classification. A second approach uses neural nets with the same data for classification.

The use of histograms for classification leads naturally to the estimation of mixture densities, which is the long-term goal of this work.

3.1. Packet size distribution

A histogram is computed to estimate the distribution of packet sizes for each application. For the initial analysis, a bin width of one was used. It was observed that for most applications, the number of packets in the small packet-size range, i.e. between 60-80 bytes, was large. In fact, it is so large that if a graph used linear scaling on Y-axis, the lines for other packet sizes become almost invisible. Thus, logarithmic scaling is used to display the histograms.

Based on the observation of fine resolution histograms, a low-resolution histogram was constructed for further processing. This histogram uses 50 non-linearly distributed bins. The bins are narrow in the small packet-size region, where the packet-size distribution is dense, and increase toward the large packet-size region. The plots of average packet size distribution across the entire training set are shown in Figure 1. The histogram is normalized and logged before plotting in order to magnify the large packet-size region.

3.2. Inter-arrival Time Distribution

We used the same approach as in packet-size distributions to compute the histograms for inter-arrival time distributions. For this data we used 49 non-uniformly distributed bins. The plots of average inter-arrival time distribution across the entire training set are shown in Figure 2. As before, the histogram is normalized and logged before plotting in order to magnify the large inter-arrival time region corresponding to high bin numbers.

3.3. Using both Packet Size and Inter-arrival Time Information

The more information we add, the better the resulting classification should be. Following this principle, we thought of combining the packet size and inter-arrival time information. However, the results would be better only if the two variables are relatively uncorrelated. To investigate this, we calculated the correlation coefficient of packet size and inter-arrival time over the entire training set. Table 2 gives the average correlation coefficients between packet size and inter-arrival time for all applications across the training set.

It is observed that the packet size and inter-arrival time are reasonably uncorrelated. Hence, we experimented with using a combination of the two as a variable. We did this using two

Table 1. Contribution of different applications to the internet traffic

Application	Port Number	Number of Packets	% contribution
dns	53	179960	1.50
ftp(data)	20	933777	7.78
http	80	4290363	35.75
nntp	119	560403	4.67
reserved	0	397384	3.31
smtp	25	540878	4.51
telnet	23	416830	3.47
other	-	4681205	39.01

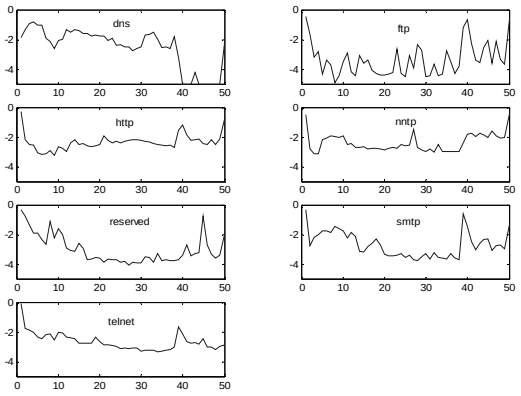

Figure 1: Plots of histograms (normalized and logged) for packet-size distributions (in 50 bins) of different applications.

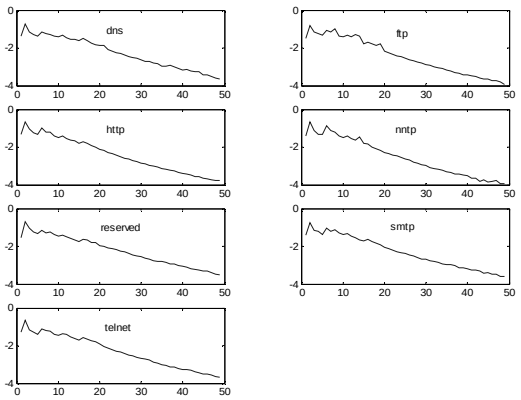

Figure 2: Plots of histograms (normalized and logged) for inter-arrival time distributions (in 49 bins) of different applications.

approaches.

In one, we generated a 2-D histogram matrix of dimension 49x50 for each application, where the rows indicated the packet size distributions, and columns the corresponding inter-arrival time distributions. Then we treated the matrix as an image, and used the Wavelet Toolbox of MATLAB to decompose the image into its wavelet coefficients. We eliminated the wavelet coefficients having the least standard deviations across the 56 vectors to compress the image, and then reconstruct it with the remaining coefficients. In this way,

Table 2. Correlation coefficients between packet size and inter-arrival times for different applications.

Application	Correlation Coefficient
dns	0.0368
ftp	0.1157
http	-0.0006
nntp	0.1361
reserved	-0.0374
smtp	0.0928
telnet	0.0041

we could reduce the dimension of the variable vector from 2450 (49x50), to 250. In Figure 3 we show an example of comparison of the two images. The first plot is the original figure, and the second is the one reconstructed using 250 wavelet coefficients.

In the second approach, we simply concatenate the packet-size distribution and inter-arrival time distribution vectors to form a single vector of dimension 99.

3.4. Clustering

We tested a variety of clustering methods for this purpose. For clustering, we used the packet size distribution, inter-arrival time distribution, and the combination of both as features. These distributions for all the 7 applications across the 8 data sets of the training set were used as instances of these features. Hence, for each feature, we have 56 data-points to cluster.

The clustering was done using the SAS v.7 software. We investigated four clustering methods: Average Method, Centroid Method, Complete Linkage and Ward's Minimum-variance Method [9]. Of these, the Ward's method gave the best results, and it was the only one to give perfect clustering. In this method, the distance between two clusters is the sum of squares between the two clusters summed over all variables. At each generation, the within-cluster sum of squares is minimized over all partitions obtainable by merging two clusters from the previous generation.

Note: In this analysis, we have used histograms that are logged and normalized as in the previous section. If 3 or more vectors of an application are clustered together and constitute a majority in the cluster, it is considered to be a valid cluster of that application. All others are considered as misclassifications.

The clustering results of the 56 packet-size distribution vectors using the Ward's method are given in Table 3. The clustering based on inter-arrival time alone, and that using a combination of packet-size and inter-arrival time was poor. This might be expected from the similarities of the inter-arrival time histograms in Figure 2.

Since the packet size distributions gave better clustering results, while at the same time using less information as compared to the 2-D case, we decided to go ahead with the packet size distributions as our classification parameter.

3.5. Reducing the Size of Data Set for Training

We tested the performance of the classification scheme by using smaller data sets, and thus obtain a trade-off between data set size and classification efficiency. Instead of using 3 files in a sub-set, we performed a similar analysis using 1 file, i.e. the size of the data used was reduced to $1/3^{rd}$ as that of the original. The clustering results with this reduced data are given in Table 4.

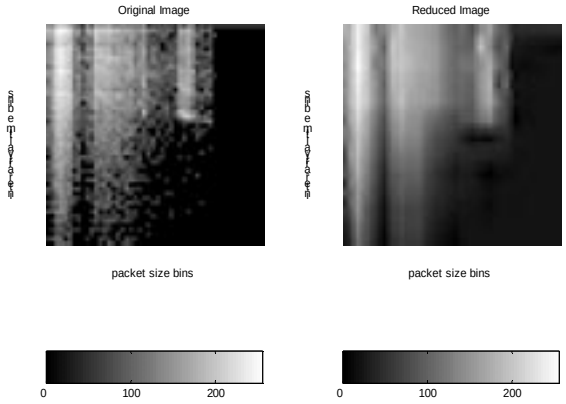

Figure 3: Plots of images (original and compressed) for packet size vs. inter-arrival time distributions of dns

Table 3. Results of clustering the packet size distributions

Cluster Number	Members(Number/Application)
1	8 / http
2	8 / dns
3	8 / nntp
4	8 / telnet
5	8 / smtp
6	8 / ftp
7	8 / reserved

Table 4. Results of clustering the packet size distributions using reduced data sets

Cluster Number	Members(Number/Application)
1	24 / dns
2	24 / http, 1 / ftp
3	22 / nntp, 1 / ftp
4	12 / reserved
5	19 / telnet
6	12 / reserved, 4 / telnet
7	24 / smtp, 2 / nntp, 1 / telnet
8	22 / ftp

Here, we see that according to our criterion of successful clustering, we have 9 misclassifications out of a total of 168(7x24) vectors used. However, considering the fact that we have used only one-third of the data, the result is quite satisfactory.

3.6. Classification Based on Clustering Results

From the study above, it is possible to classify network traffic into various application types using packet size distribution alone. To perform classification using the results of clustering, we added the test vector to each cluster and measured the error using the clustering criterion. The test vector belongs to the cluster in which the total error comes out to be the least.

The results of the above experiment using the reduced data are given in Table 5. It can be seen that a total of 6 vectors are

Table 5. Results of classification – clustering using training set and experimenting with test set.

Experi-mental	Actual						
	dns	ftp	http	nntp	res	smtp	tel
dns	24						
ftp		24					
http			24				
nntp				23			
res					24		
smtp				1		24	5
tel							19

misclassified (1 nntp and 5 telnet). Hence, we get 96.43% accuracy in estimating the application to which a particular packet size distribution belongs.

We then repeat the same experiment in a reverse manner, i.e. we form the clusters using the test data set, and test using the training data set. The results of that are given in Table 6. Here, we see that 9 vectors are misclassified, i.e. 94.64% accuracy.

However, it should be noted that on a packet-count basis, the accuracy is higher – 98.49% in the case of Table 5 and 97.26% in the case of Table 6. This is because all of http vectors are classified properly, and http accounts for a large part of the traffic mix as observed earlier.

3.7. Classification Using Neural Nets

For classification using the Neural Net approach, we use the same histograms of packet size distribution as used in the clustering approach to classification. We have used a two-layer feed-forward backpropogation network with seven neurons in the hidden layer. Both the hidden and output layers use the Log sigmoid transfer function. We trained the neural net with the histograms of the training data set. The training used cross-validation on the test data set as a stopping criterion. The classification results on the test set using this method are given in Table 7. The results of reverse experiment, i.e. training using test data set and classifying using training data set are given in Table 8. We observe that we get slightly better accuracy of classification than with the clustering approach – 98.81% in Table 7 and 97.62% in Table 8. It should also be noted that the neural approach is computationally much faster than the clustering method.

4. CONCLUSION AND FUTURE WORK

We saw that using clustering algorithms and the neural net approach, we are able to classify major applications of Internet traffic based on packet size histograms. From that, it can be concluded that classification of Internet traffic using the features that are intrinsic characteristics of the packets, and features that cannot be spoofed so easily, is possible. This gives a completely new perspective of 'implicit' classification for differentiated services.

Future work will develop methods to estimate traffic mix. The work will identify the time-critical applications and develop adaptive estimation methods.

5. REFERENCES

[1] Mika Ilvesmaki, Marko Luoma, Raimo Kantola: Flow classification schemes in traffic-based multilayer IP switching – comparison between conventional and neural approach, Computer communications 21 (1998), pp. 1184-1194.

[2] Fulu Li, Nabil Seddigh, Biswajit Nandy, Diego Malute: An empirical study of today's Internet traffic for Differentiated Services IP QoS, Proceedings of ISCC 2000.

[3] K. Glaffy, G. Miller, K. Thompson: The Nature of the Beast: Recent traffic measurements from an Internet Backbone, Proceedings of INET '98, June 1998.

[4] Assigned Port Numbers, http://www.iana.org/

[5] Dan Decasper, Zubin Dittia, Guru Parulkar, Bernhard Plattner: Router Plugins: A software architecture for next-generation routers, IEEE/ACM Transactions on Networking, Vol. 8, No. 1, February 2000.

[6] Pankaj Gupta, Nick McKeown: Algorithms for Packet Classification, IEEE Network Magazine, March/April 2001, vol. 15, no. 2, pp. 24-32.

[7] Mary L. Bailey, Burra Gopal, Michael A. Pagels, Larry L. Peterson, Prasenjit Sarkar: PathFinder: A pattern-based packet classifier, Proceedings of the First Symposium on Operating Systems Design and Implementation, pp. 115-123, November 1994.

[8] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Wiess: An Architecture for Differentiated Services, RFC 2475.

[9] SAS/STAT user guide, vol. 1.

Table 6. Results of reverse classification – clustering using test set and experimenting with training set.

Experi-mental	Actual						
	dns	ftp	http	nntp	res	smtp	tel
dns	24						
ftp		22					
http		1	24				
nntp		1		23			
res					18		
smtp				1	6	24	
tel							24

Table 7. Results of Neural Net classification – training using training set and classifying with test set.

Experi-mental	Actual						
	dns	ftp	http	nntp	res	smtp	tel
dns	24						
ftp		23					
http			24				
nntp		1		24		1	
res					24		
smtp						23	
tel							24

Table 8. Results of reverse Neural Net classification – training using test set and classifying with training set.

Experi-mental	Actual						
	dns	ftp	http	nntp	res	smtp	Tel
dns	23						
ftp		22					
http	1		24				
nntp				24			
res					24		1
smtp		1				24	
tel		1					23