

Self-learning IP Traffic Classification based on Statistical Flow Characteristics

Sebastian Zander¹, Thuy Nguyen¹, Grenville Armitage¹

Centre for Advanced Internet Architectures (CAIA)
Swinburne University of Technology, Melbourne, Australia
{szander, tnguyen, garmitage}@swin.edu.au

Abstract. A number of key areas in IP network engineering, management and surveillance greatly benefit from the ability to dynamically identify traffic flows according to the applications responsible for their creation. Currently such classifications rely on selected packet header fields (e.g. destination port) or application layer protocol decoding. These methods have a number of shortfalls e.g. many applications can use unpredictable port numbers and protocol decoding requires high resource usage or is simply infeasible in case protocols are unknown or encrypted. We propose a framework for application classification using an unsupervised machine learning (ML) technique. Flows are automatically classified based on their statistical characteristics. We also propose a systematic approach to identify an optimal set of flow attributes to use and evaluate the effectiveness of our approach using captured traffic traces.

1 Introduction

Over recent years there has been a dramatic increase in the variety of applications used in the Internet. Besides the ‘traditional’ applications (e.g. email, web) new applications have gained strong momentum (e.g. gaming, P2P). The ability to dynamically classify flows according to their applications is highly beneficial in a number of areas such as trend analysis, network-based QoS mapping, application-based access control, lawful interception and intrusion detection.

The most common identification technique based on the inspection of ‘known port numbers’ suffers because many applications no longer use fixed, predictable port numbers. Some applications use ports registered with the Internet Assigned Numbers Authority (IANA) but many applications only utilise ‘well known’ default ports that do not guarantee an unambiguous identification. Applications can end up using non-standard ports because (i) non-privileged users often have to use ports above 1024, (ii) users may be deliberately trying to hide their existence or bypass port-based filters, or (iii) multiple servers are sharing a single IP address (host). Furthermore some applications (e.g. passive FTP) use dynamic ports unknowable in advance.

A more reliable technique involves stateful reconstruction of session and application information from packet contents. Although this avoids reliance on fixed

¹ Work supported by Cisco Systems, Inc under the University Research Program.

port numbers, it imposes significant complexity and processing load on the identification device, which must be kept up-to-date with extensive knowledge of application semantics, and must be powerful enough to perform concurrent analysis of a potentially large number of flows. This approach can be difficult or impossible when dealing with proprietary protocols or encrypted traffic. The authors of [1] propose signature-based methods to classify P2P traffic. Although these approaches are more efficient than stateful reconstruction and provide better classification than the port-based approach they are still protocol dependent.

Machine Learning (ML) automatically builds a classifier by learning the inherent structure of a dataset depending on the characteristics of the data. Classification in a high dimensional attributes space is a big challenge for humans and rule-based methods, but stochastic ML algorithms can easily perform this task. The use of stochastic ML for traffic classification was raised in [2], [3] and [4]. However, to the best of our knowledge no systematic approach for application classification and evaluation has been proposed and an understanding of possible achievements and limitations is still lacking. We propose a detailed framework for self-learning flow classification based on statistical flow properties that includes a systematic approach of identifying the optimal set of flow attributes that minimizes the processing cost, while maximizing the classification accuracy. We evaluate the effectiveness of our approach using traffic traces collected at different locations in the Internet.

2 Related Work

Previous work used a number of different parameters to describe network traffic (e.g. [1], [5], [6]). The idea of using stochastic ML techniques for flow classification was first introduced in the context of intrusion detection [2]. The authors of [7] use principal component analysis and density estimation to classify traffic into different applications. They use only two attributes and their evaluation is based on a fairly small dataset. In [3] the authors use nearest neighbour and linear discriminate analysis to separate different application types (QoS classes). This supervised learning approach requires an a-priori knowledge of the number of classes. Also, it is unclear how good the discrimination of flows is because in [3] the sets of attributes are averaged over all flows of certain applications in 24-hour periods. In [4] the authors use the Expectation Maximization (EM) algorithm to cluster flows into different application types using a fixed set of attributes. From their evaluation it is not clear what influence different attributes have and how good the clustering actually is.

3 ML-based Flow Classification Approach and Evaluation

As initial input we use traffic traces or capture data from the network. First we classify packets into flows according to IP addresses, ports, and protocol and compute the flow characteristics. The flow characteristics and a model of the flow attributes are then used to learn the classes (1). Once the classes have been learned new flows can be classified (2). The results of the learning and classification can be exported for

evaluation. The results of the classification would be used for e.g. QoS mapping, trend analysis etc. We define a flow as a bidirectional series of IP packets with the same source and destination address, port numbers and protocol (with a 60 second flow timeout). Our attribute set includes packet inter-arrival time and packet length mean and variance, flow size (bytes) and duration. Aside from duration all attributes are computed in both directions. We perform packet classification using NetMate [8], which supports flexible flow classification and can easily be extended with new flow characteristics. For the ML-based classification we use autoclass [9], an implementation of the Expectation Maximization (EM) algorithm [10]. EM is an unsupervised Bayesian classifier that automatically learns the ‘natural’ classes (also called clustering) inherent in a training dataset with unclassified cases. The resulting classifier can then be used to classify new cases (see [4], [9]).

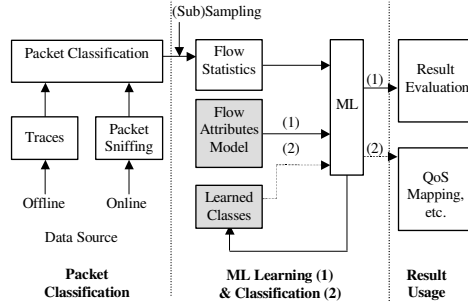


Fig. 1. ML-based flow classification

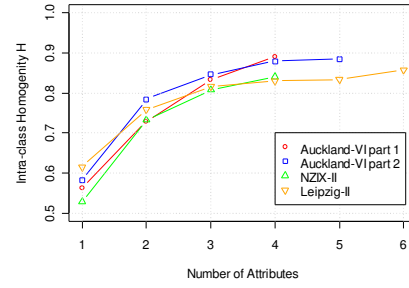


Fig. 2. Intra-class homogeneity

For the evaluation we use the Auckland-VI, NZIX-II and Leipzig-II traces from NLNR [11] captured in different years at different locations. Because the learning process is slow we use 1,000 randomly sampled flows for eight destination ports (FTP data, Telnet, SMTP, DNS, HTTP, AOL Messenger, Napster, Half-Life), which results in a total of 8,000 flows. Finding the combination of attributes that provides the most contrasting application classes is a repeated process of (i) selecting a subset of attributes, (ii) learning the classes and (iii) evaluating the class structure.

We use sequential forward selection (SFS) to find the best attribute set because an exhaustive search is not feasible. The algorithm starts with every single attribute. The attribute that produces the best result is placed in a list of selected attributes SEL(1). Then all combinations of SEL(1) and a second attribute not in SEL(1) are tried. The combination that produces the best result becomes SEL(2). The process is repeated until no further improvement is achieved. To assess the quality of the resulting classes we compute the intra-class homogeneity H . We define C and A as the total numbers of classes and applications respectively. If N_{ac} is the number of flows of application a that fall into class c and N_c is the total number of flows in class c H_c is defined as:

$$H_c = \max\left(\frac{N_{ac}}{N_c} \mid 0 \leq a \leq A-1\right) \quad (0 < H \leq 1) \quad (1)$$

For each trial H is the mean of H_c for $0 \leq c \leq C-1$ and the objective is to maximize H to achieve a good separation between different applications. For the evaluation we assume a flow’s destination port defines the application. This may be incorrect (as stated initially) but we assume it is true for a majority of the flows. Unfortunately

public available traces do not contain payload information usable for verification.

For each trace (and for two different parts of Auckland-VI) the best set of attributes found is different and the size varies between 4-6 (see Fig.2.). We rank the attributes according to how often they appear in the best set: forward packet length mean, forward/backward packet length variance, forward inter-arrival times mean and forward size (75%), backward packet length mean (50%), duration and backward size (25%). Clearly, packet length statistics are preferred over packet inter-arrival time statistics for the ports we use. The average maximum H is 0.87 ± 0.02 but H greatly differs for different ports (e.g. 0.98 ± 0.01 for Half-Life but only 0.74 ± 0.14 for HTTP).

4 Conclusions and Future Work

We have proposed a framework for ML-based flow classification based on statistical flow properties, identified a systematic approach of identifying an optimal set of flow attributes and evaluated the effectiveness of our approach. The results show that some separation of the applications can be achieved if the flow attributes are chosen properly. We plan to evaluate our approach with a larger number of flows and more applications (e.g. audio/video streaming). We hope to get traces that contain payload information usable for verifying the actual applications. We also plan to experiment with more attributes (e.g. idle time, burstiness) and possibly use payload information in a protocol independent way. Furthermore the precision of the resulting classifier and the classification performance has not yet been evaluated.

References

1. S. Sen, O. Spatscheck, D. Wang, "Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures", WWW 2004, New York, USA, May 2004.
2. J. Frank, "Machine Learning and Intrusion Detection: Current and Future Directions", Proceedings of the National 17th Computer Security Conference, 1994.
3. M. Roughan, S. Sen, O. Spatscheck, N. Duffield, "Class-of-Service Mapping for QoS: A statistical signature-based approach to IP traffic classification", ACM SIGCOMM Internet Measurement Workshop 2004, Taormina, Sicily, Italy.
4. A. McGregor, M. Hall, P. Lorier, J. Brunskill, "Flow Clustering Using Machine Learning Techniques", Passive & Active Measurement Workshop 2004, France, April, 2004.
5. K. Lan, J. Heidemann, "On the correlation of Internet flow characteristics", Technical Report ISI-TR-574, USC/Information Sciences Institute, July, 2003.
6. K. Claffy, H.-W. Braun, G. Polyzos, "Internet Traffic Profiling", CAIDA, San Diego Supercomputer Center, <http://www.caida.org/outreach/papers/1994/itf/>, 1994.
7. T. Dunnigan, G. Ostrouchov, "Flow Characterization for Intrusion Detection", Oak Ridge National Laboratory, Tech Report, <http://www.csm.ornl.gov/~ost/id/tm.ps>, November 2000.
8. NetMate, <http://sourceforge.net/projects/netmate-meter/> (as of January 2005).
9. P. Cheeseman, J. Stutz, "Bayesian Classification (Autoclass): Theory and Results", Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, USA, 1996.
10. A. Dempster, N. Laird, D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm, Journal of Royal Statistical Society, Series B, Vol. 30, No. 1, 1977.
11. NLNLR traces: <http://pma.nlanr.net/Special/> (as of January 2005).