CS489 Independent Study

Final Paper

Network Connection Classifier

Author:
CDT Tyler Reece
Class of 2019
tyler.reece@westpoint.edu

Advisor:
Mr. Kyle King
NSA Visiting Professor
kyle.king@westpoint.edu

United States Military Academy
Department of Electrical Engineering and Computer Science
West Point, New York
December 2018

Abstract

This paper presents several supervised machine learning models for labeling Domain Name Service (DNS) query names as either primary or secondary web connections. As the complexity of a typical web browsing session grows, these models aim to classify connections as either primary or secondary, allowing increased understanding of the genesis of a single connection. The results of this work could be utilized in the future to generate a TCP-only model for classifying connections.

Introduction

The web has grown extremely complex in the last decade. A typical web browsing session in the early days of the Internet consisted of only primary connections, defined as information that results in the genesis of a browsing session. Such primary connections consisted of HyperText Markup Language (HTML) and rudimentary Cascading Style Sheets (CSS). However, the number of secondary connections - connections that were implicitly made in support of a primary connection - has grown exponentially in recent years. Secondary connections include a wide variety of information such as images, JavaScript, video, ad services, custom fonts, Content Distribution Networks (CDNs), and third party analytics. The average web page today loads over 100 objects, relying heavily on the use of CDNs and geographic and logical load balancing.[1] Since secondary connections now outnumber primary connections by a large margin, it can be difficult to understand the genesis of a single connection, or indeed what is happening on a network at all. This problem is compounded by the diagnostic nature of aging network analysis tools and ubiquitous encryption.

Traditional, text-based network analysis tools such as WireShark, tcpdump, and Snort can assist an experienced user answer specific questions about activity on a network. However, macro trends and basic questions about a network can often be difficult to answer. Thus, understanding the genesis of a single connection by separating primary from secondary connections is increasingly necessary for network administrators and users of network analysis tools alike.

In this work, we show how various supervised learning models, including k-nearest neighbors (KNN), support vector machines (SVMs), multilayer perceptrons (MLPs), and decision trees can classify DNS queries as primary or secondary with a high degree of accuracy.

Technical Background

This research focused on the use of supervised learning to train a model to label DNS queries as part of primary or secondary connections. In supervised learning, models observe example input-output pairs to learn a function that maps input to output.[2] In the case of the classification models presented in this paper, the input values are a number of quantifiable integer features extracted from DNS packets (see Methods and Results), and the output is one of binary set of values (primary or secondary).

In a supervised machine learning algorithm, the input data is typically separated into three sections: training, validation (or holdout) and testing. Training data typically makes up around 50% of the data, while validation and testing make up 25% each.[3] This allows a model to observe features and patterns in the training data, before any hyperparameters (any non-feature inputs that affect the performance of the model) are calibrated based on error generated in the model's performance on the validation data. The model's ability to generalize (to be effective across a broad range of inputs and applications) is then evaluated using the testing data. A highly generalizable model avoids overfitting, which occurs when a model captures the noise of the data, fitting specific test cases too well. Overfitting can be caused by an overcomplicated model with an overabundance of features, and prevented by fitting multiple models and utilizing cross-validation to compare accuracy.[4] In contrast, underfitting occurs when a model is unable to capture the underlying trend or pattern in the data, where the model is too simple to work effectively.[4]

This research utilized a total of four types of supervised learning classification algorithms, including k-nearest neighbors (KNN), multilayer perceptron networks, decision trees, and a support vector machines (SVMs). K-nearest neighbors, one of the simplest machine learning methods, classifies unlabeled observations by assigning them to the class of the most similar labeled examples.[5] This concept of "similarity" (also commonly called "distance") of one sample to another can be calculated in any number of dimensions with the Euclidean or Manhattan distance formulas.[5] KNN takes a parameter k, which decides how many neighbors are chosen for comparison in the algorithm. The choice of the k value strikes a balance between overfitting and underfitting, and some authors suggest to set k equal to the square root of the number of observations in the training dataset as a rule of thumb.[5]

A multilayer perceptron (MLP) network is a class of artificial neural network, consisting of at least three layers of nodes: an input layer, a hidden layer, and an output layer.[6] Except for the input nodes, each node is a neuron that utilizes a nonlinear activation function that maps inputs to outputs. AN MLP utilizes backpropagation, a method used to calculate the gradient needed to alter the weights of features and minimize error in the classification of training samples.[6] The MLP network used in this study utilized two hidden layers of size five and ten, respectively.

A decision tree takes a vector of feature values and performs a sequence of tests and builds a classification model in the form of a tree structure, where a leaf node represents a classification or decision, and each decisions have two or more branches. The topmost decision node in a tree (the root node) corresponds to the best predictor in the model.[2] In this case, we are most interested in decision trees that involve Boolean classification, where each example input is classified as true: a primary connection, or false: a secondary connection.[7]

Support Vector Machines are often utilized for categorical classification problems, such as the one in this study. SVMs attempt to find a hyperplane that best divides a dataset into distinct classes. Support vectors (data points nearest to the hyperplane) are those that, if removed, would alter the position of the dividing hyperplane.[8] Hyperplanes are placed so as to generate the largest possible margin (distance between the hyperplane and nearest data point to either side), so that there is the greatest chance of new data being classified correctly[6].

Related Work

Machine learning (ML) techniques have been applied to many applications in recent years, largely due to the unprecedented availability of data, improvements in ML techniques, and advances in computing capabilities.[9] ML approaches are natural solutions to problems that have a large representative dataset with uncertain relationships between features, so they are a natural choice for identifying and predicting trends in network traffic.

Machine Learning techniques have been generally applied to network traffic in two categories: application classification and intrusion detection. Application classification attempts to utilize ML techniques to identify various web applications (FTP, Telnet, SMTP, DNS, HTTP) from features such as port numbers, unencrypted packet headers, and inter-packet arrival time.[8] These techniques face difficulties keeping pace with the continuously growing number of web applications, growing encryption, and with applications that dynamically assign ports.[10] However, ML models utilizing support vector machines have demonstrated up to 99% accuracy using these methods, with similar results from other kinds of models including neural networks and deep learning approaches.[8]

ML techniques have also been utilized in the area of network traffic analysis. Intrusion detection systems (IDSs) aim to identify unusual access or attacks on secure networks.[11] ML techniques have focused on both operating system logs and network traffic to detect possible intrusion attempts.[12] Support vector machines have been shown to have accuracies up to 98%, and though it takes considerably longer to train a neural network model, such approaches have had up to 99.25% accuracy.[11]

Methods and Results

Two datasets were utilized to train and later to cross-validate the models in this study. Each dataset was self-generated by the author, and was captured in .pcap format via WireShark. Each dataset was comprised of approximately 30 different commonly visited websites (see Appendix A for a full listing), as well as visits to 2-3 hyper-links within each main website. The resulting .csv file, when filtered using TShark (a command line version of Wireshark) to include only DNS queries, contained approximately 1200 rows each.

A Python script constituted a feature extractor tool, which iterated through the rows in the .csv file, extracting and quantizing various features that would be utilized by the model. Features included all integer values such as query name length, presence and number of special characters (periods, dashes, and numbers), as well as the presence of specific domain endings (.com, .net, .org, .edu). Each query was then hand labeled as either primary (a value of 1) or secondary (a value of 0) to support the supervised learning approach.

The samples were then split into training and test sets in an 80:20 ratio. Utilizing scikit-learn, a free machine learning library for Python, several models were evaluated on their performance learning from the training data. Accuracy percentages among the same dataset are shown in the table below.

Model	Dataset 1	Dataset 2
KNN	97.68% (k=7)	98.47% (k=9)
MLP	98.95%	91.77%
DT	96.85%	89.48%
SVM	96.00%	98.85%

The models were also cross-validated by training on one dataset, then testing on a separate dataset which contained no similar websites. Accuracy percentages were generally marginally lower for each model. Below, the column headers indicate the order that the datasets were used, with the first as the training dataset and the second as the testing dataset.

Model	(1,2)	(2,1)
KNN	96.71% (k=6)	97.68% (k=8)
MLP	95.55%	98.42%
DT	95.54%	98.42%
SVM	96.62%	97.03%

Analysis

In general, all models performed with a high degree of accuracy both within a dataset and in the cross-validation tests performed across mutually exclusive datasets. K-nearest-neighbors, the simplest approach attempted, actually consistently performed the best throughout all trials in the 96%-98% range, with the optimal k value somewhere between 6 and 9. The SVM performed similarly. While the MLP model performed well, it was inconsistent in its same-dataset test on Dataset 2. The decision tree performed the poorest, and was very inconsistent in its trials, with a range of performance of nearly 7.5% between same-dataset tests.

Conclusion and Future Work

The models presented in this paper show definite promise in their application to classifying network connections. The MLP and

SVM models are particularly intriguing, consistently showing over 98% accuracy in test runs both in the same dataset and through cross-validation against a dataset of completely unrelated browsing content.

However, additional research both on the models presented in this paper and new approaches are necessary. The models presented here show some weaknesses, particularly when attempting to categorize excessively long domain names (above 20 characters in length), likely hinting at the overimportance of the length of a query name as a feature. Similarly, further investigative testing should be completed to find other suspected weaknesses, such as the model's handling of subdomain names, such as mail.google.com, which it currently incorrectly classifies as secondary. Training the model on much larger (10,000 row) datasets and introducing additional features (presence of subdomains and letter frequency) could yield much higher accuracy.

Just as the evolution of the Internet provided the motivation for this project, the changing nature of computer network traffic will continue to shape future research in this area. With the impending emergence of encrypted DNS in the near future, a successful network connection labeling model will ultimately need to utilize features of TCP connections to classify connections. A promising approach that builds upon the work presented in this paper would be to sort TCP packets into streams, and correlate them with their corresponding DNS requests by temporal locality. Then, this correlation could be used to create a large, automatically labeled TCP dataset, which could be used to train a TCP-only model, thus mitigating the problem posed by encrypted DNS.

Finally, once a highly accurate model is obtained, it could be used in real time as a filter for traditional network analysis tools. By only displaying primary connections, high level questions about a network would be much simpler to answer, even for an inexperienced user. Additionally, network administrators would benefit from an increased capability to monitor traffic and understand patterns on a network.

Acknowledgements

The author would like to thank Mr. Kyle King for his inspiration, guidance, and mentorship throughout the research process, as well as all the faculty of the USMA EECS Department for their support over the years.

Appendix A

Dataset 1
Amazon
Google
CNN
Facebook
Yahoo
Wikipedia
MSNBC
FoxNews
Reddit
Nike
Adidas
Instagram
Netflix
Microsoft
Bing
Youtube
LinkedIn
eBay

Dataset 2
Zillow
Adobe
Vimeo
Flickr
GoDaddy
NYTimes
GitHub
MySpace
The Guardian
Huffington Post
Yelp
USA Today
NPR
NASA
Skype
CNBC
US News
Debian
WikiHow

References

[1] Domenech, Gomez, Charzinski, and Sounders, "Growth of Average Web Page Size and Number of Objects - Jan 1995-Nov 2012", Image.

[2] S. Russell and P. Norwig, *Artificial Intelligence: A Modern Approach*, Saddle River, NJ, 2010.

[3] R. Boutaba et al, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, 2018.

[4] Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning*, pp. 222.

[5] Z. Zhang, "Introduction to machine learning: k-nearest neighbors," *Annals of Translational Medicine*, 2016, pp. 218-225.

[6] E. Zanaty, "Support Vector Machines versus Multilayer Perceptron in data classification," *Egyptian Informatics Journal*, 2012, pp. 177-183.

[7] S. Murthy, "Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey," *Data Mining and Knowledge Discovery*, 1998, pp. 345-389, vol. 2.

[8] R. Yuan, Z. Li, and X. Guan, "An SVM-based machine learning method for accurate Internet traffic classification," *Information Systems Frontiers*. Beijing, China, 2010, pp. 149-156, vol. 12.

[9] S. Zander, T. Nguyen and G. Armitage, "Automated traffic classification and application identification using machine learning," *The IEEE Conference on Local Computer Networks 30th Anniversary*, Sydney, NSW, 2005, pp. 250-257.

[10] M. Soysal and E. Schmidt, "Machine learning algorithms for accurate flow-based network traffic classification: evaluation and comparison," *Performance Evaluation*. 2010, pp. 451-467, vol. 67.

[11] S. Mukkamala, G. Janoski and A. Sung, "Intrusion detection using neural networks and support vector machines," Proceedings of the 2002 International Joint Conference on Neural Networks. Honolulu, HI, USA, 2002, pp. 1702-1707 vol. 2.

[12] T. Chih-Fong et al, "Intrusion detection by machine learning: a review," *Expert Systems with Applications*. Taipei, Taiwan, 2009, vol. 36.

The code (including feature extractors, machine learning models, and demo files), datasets, and presentations used in this paper are available for use and can be found on GitHub at the following link: `https://github.com/tylerreece/Reece_CS489`.