Analyzing Network Traffic in Diverse Network Conditions

Parker Addison, Sahil Altekar, Danial Yaseen
Halıcıoğlu Data Science Institute
University of California, San Diego
{pgaddiso,saltekar,dyaseen}@ucsd.edu

March 2021

ABSTRACT
In the modern, online world VPN usage has grown rapidly. This introduces new challenges for ISPs and the field of network traffic classification. Encrypted traffic classification researchers often self-generate datasets to build classifiers, but these datasets are typically collected from a single set of network conditions. This poses a data representation issue. In our paper we introduce a new tool to collect data in various network conditions and use that data to visualize differences between conditions. We then evaluate the effect of different network conditions on the performance of a classifier trained on a previously collected dataset which lacked diverse network conditions. Finally, we demonstrate that there exist statistical features commonly used for traffic classification which are robust to differences in network conditions and thus preserve model performance.

1. INTRODUCTION
When a device is connected to the internet, data is sent to and received from servers as 'network traffic'. Typically, a device sends a request for data to a server in the form of a packet following the Internet Protocol, and in turn receives one or many packets from the server which contain the requested information. The time it takes for a request to reach and be acknowledged by a server is called latency, and the maximum rate at which multiple packets can be sent or received is called bandwidth. Based on different aspects such as user location, internet service provider, or device hardware, observed latency and bandwidth on different networks can vary considerably. Typically, low latency and high bandwidth results in a smoother user experience and is considered to be a 'good' network condition. High latency and low bandwidth are typically considered 'poor' network conditions.

Internet Service Providers (ISPs) analyze network traffic to discern the performance their customers are achieving and to better understand how their customers prefer to use the web. Traffic classification is a specific branch of network traffic analysis which aims to categorize a set of network traffic into the class of user behavior or application which generated the packets. Traffic classification can be leveraged by ISPs to optimize and improve their services, such as by prioritizing a low-latency connection for a customer they recognize as engaging in video-conferencing. Traffic classification may examine packet-level information such as the source and destination IP addresses, the application protocol number, and even the payload data itself to perform classification. However, with the advent of encrypted protocols like HTTPS and with recent widespread user adoption of virtual private networks (VPNs), these features are obscured or obfuscated via encryption [1]. Encrypted traffic classification is an area of research which utilizes statistical features of traffic flows to perform traffic classification even when the individual packet contents may have been or encrypted or passed through a VPN.

Recently, the company Viasat—an ISP which focuses on internet-over-satellite—partnered with researchers at the University of California, San Diego to develop encrypted traffic classification models capable of detecting whether a VPN user is streaming video over the internet. In order to train the machine learning model, a dataset of streaming and browsing network traffic was manually generated at the university. For many researchers in the field of traffic classification it is common to similarly self-generate a dataset from a small set of computers, a computer lab, or a single area on a campus [2][3][4]. However, because researchers and campuses generally have access to relatively good network conditions, this introduces an issue of data representation—a classifier may be developed and trained on data which only represents good network conditions. When it comes to deploying such a classifier, we cannot reasonably expect all internet users to have the same network conditions. For example, because Viasat provides internet over satellite rather than via terrestrial cables, their customers will have substantially greater latency than users of a standard network setup.

In our paper we examine the performance of the previously developed streaming classifier when evaluated on network traffic data generated in a variety of network conditions, relying on a tool we engineered to facilitate and automate a more representative data generation and collection process.

2. DATA

The previous dataset, which was used to develop and train the streaming classifier, was generated by a total of 19 researchers at UC San Diego each with a unique device. The researchers collected data with strong, stable internet connections. To generate the raw network traffic, each participant watched videos from providers such as YouTube, Netflix, and Hulu, or browsed the internet while avoiding such websites. These data were labeled 'streaming' and 'browsing', respectively. To capture the raw traffic as a usable dataset, Viasat provided the tool, network-stats, which summarizes traffic flows on a per-connection, per-second basis [5]. Each second, network-stats outputs metadata such as the timestamp, source source and destination IPs, applications ports, and communication protocol for each unique IP pair seen that second. For each pair it also produces traffic statistics such as the aggregated and individual counts, sizes (bytes), and arrival/departure times of downloaded/uploaded packets in that second. Participants collected the majority of data while connected to a VPN service provided by the university. The labeled data collected manually using network-stats formed a dataset of slightly over 15 hours in total size after cleaning and removing non-VPN entries. Streaming data constituted 8.2 hours of the dataset and browsing data constituted 7 hours. The dataset was then used to train a classification model to detect streaming behavior from a traffic flow.

To enable our analysis of the classifier in different network conditions, we first created a dataset generation tool, DANE, capable of emulating target network conditions and automating the traffic generation and collection process [6]. DANE was used to collect streaming and browsing data by utilizing web automation scripts which visit YouTube to watch videos or endlessly scrolls through Twitter. The tool likewise utilizes network-stats to perform data collection and establishes a VPN connection to the same university service as was used to collect the previous dataset. Independent to network conditions, the output data from DANE closely resembles the previous dataset. Data was collected using DANE in three target types of network conditions defined below.

Table 1: Network Condition Definitions

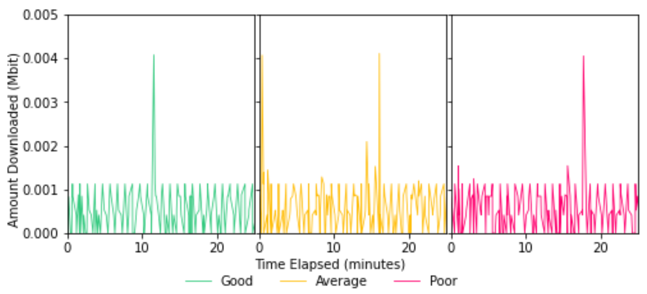
Condition	Latency	Bandwidth
Good	≤ 50 ms	≥ 40 Mbit/s
Average	50–200 ms	8–40 Mbit/s
Poor	≥ 200ms	≤ 8 Mbit/s

Each time the tool was run, we collected the same underlying behavior in parallel for all three conditions—i.e. the same set of YouTube videos or Twitter pages were visited within each network condition, and at the same time. This ensured that any difference in the output data from a single tool run was due solely to the difference in network conditions. In total, approximately 50 hours of data were collected using our tool after cleaning. Streaming constituted nearly 40 hours of the dataset, with 13 hours of streaming data in each condition.

3. METHODS

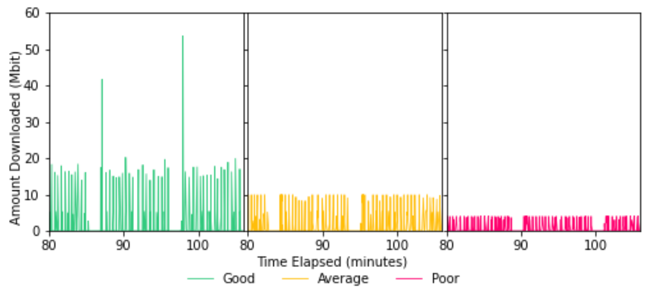
To motivate our analytical focus, we start by visually comparing the newly collected data across the three network conditions. A simple visualization of the amount of data downloaded each second produces a powerful and intuitive way to understand differences in most traffic flow data. Twenty-five minute snippets of our visual inspection are selected and shown below.

Fig 1: Megabits Downloaded in Each Second by Condition (Browsing)



The lack of visual difference between conditions for browsing immediately demonstrates that network conditions have little impact on browsing data. In general, browsing is not a demanding task, as is reflected in the small overall download size of only a couple hundred bytes per second. Most data seen while browsing is small enough to be sent by a handful of packets, so latency and bandwidth have little effect. Notably, the browsing data collected is not fully reflective of real-world browsing data. While real browsing data is 'bursty' in nature—meaning there are periods of little activity interrupted by bursts of high activity, such as when a page is loaded—the data collected appears robotically periodic. This is because the data collection tool utilizes web automation scripts, and replicating human browsing behavior with code is a difficult, unsolved research problem [7]. That said, the behavior was consistent in all conditions, so the lack of a difference suggests network condition must have had little effect. The purpose of our research is to study differences in network conditions, which apparently cannot be achieved by analyzing browsing data. Therefore we focus the rest of our analysis on the differences which arise within streaming data.

Fig 2: Megabits Downloaded in Each Second by Condition (Streaming)



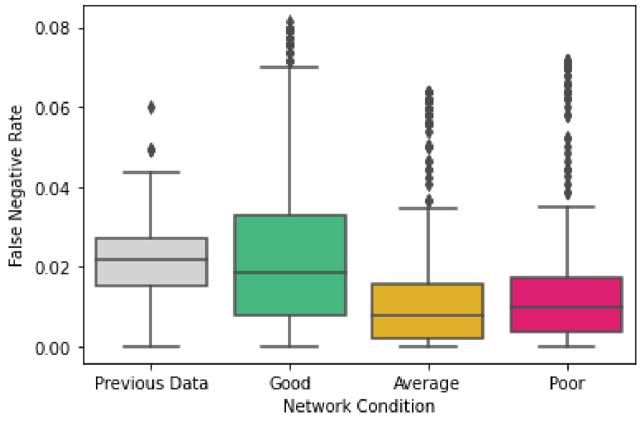
Streaming data shows a clear difference between network conditions. Foremost, the height of each chart is indicative of the bandwidth available. In a good network, well over 40 megabits of data may be downloaded within a single second. However, in an average or poor network limited to a bandwidth of 10 or 5 megabits per second, respectively, the maximum height of any bar is correspondingly limited. In each of the charts, brief periods of inactivity are seen. Most streaming service providers make use of a content buffer which pre-downloads future content as the user is watching. When approaching the end of a video, that buffer will contain the remaining data for the video and no more data needs to be requested, resulting in the inactivity seen above. After the video ends and a new one begins, however, the buffer needs to be quickly refilled, resulting in a period of higher activity. These bufferening events appear different for each network condition type. In good networks, the entire buffer can be filled quickly, resulting in a single spike of activity. In average networks, the same amount of downloaded data is spread over multiple seconds due to the bandwidth limitation, resulting in a denser region of the chart. The effect applies to poor networks, however the visual impact of a buffering event doesn't appear as extreme likely due to video resolution automatically being lowered by YouTube in poor network conditions, which causes far less data to be requested overall.

Having verified a substantial difference in streaming data between network conditions, we now explore the effect of network condition on the performance of a classifier trained to detect streaming behavior in network traffic. We wish to explore the performance of the classifier when trained on the previous data and exposed to diverse network conditions. To do so, we first transform our new dataset using the same cleaning and feature engineering pipeline which was used to train the classifier on the previous dataset. This pipeline involves splitting the multiple hours of data into 90-second segments of packet-level statistics and engineered features. After preprocessing, there are roughly 600 samples of data in the previous dataset and 520 samples of streaming data for each condition in the new dataset. We train the classifier on the previous dataset using a 70/30 train-test split, and then evaluate the classifier on the test set of the previous dataset, as well the entirety of streaming data for each condition in the new dataset. As an evaluation metric, we examine the false negative rate (FNR), which is the proportion of streaming samples which were classified as browsing. Because the classifier utilizes a non-deterministic classification model and there is randomness introduced by the train-test split, we perform 500 independent runs of the full training and evaluation process in order to mitigate the influence of randomness on our analysis.

4. RESULTS

After 500 runs of our training and evaluation process, we examine the distribution of FNR in each condition and the previous dataset's test set by comparing their resulting boxplots. Note that lower FNR indicates better performance. The boxplot shows the 25th, 50th, and 75th quantiles of FNR in each group with whiskers extending to at most 1.5 times the interquartile range (the 75th quantile minus the 25th quantile). Values outside of the whiskers are plotted as outlying points.

Fig 3: False Negative Rate by Condition, 500 runs



The respective medians (50th quantile), means, and variances for FNR in each group are shown below.

Table 2: FNR Median, Mean, and Standard Deviation by Condition

Group	Median	Mean	St. Dv.
Previous	0.022	0.021	0.01
Good	0.018	0.024	0.021
Average	0.008	0.013	0.017
Bad	0.01	0.017	0.02

The median FNR demonstrates a much greater disparity between the conditions and previous dataset than the mean. This is most likely due to a high prevalence of outliers—perhaps due to 'unlucky' training or train-test splits—as can be seen in the boxplots in Figure 3. Regardless, both median and mean FNR exhibit an absolute difference of less than 0.015 between all groups, and less than 0.01 between just the three conditions.

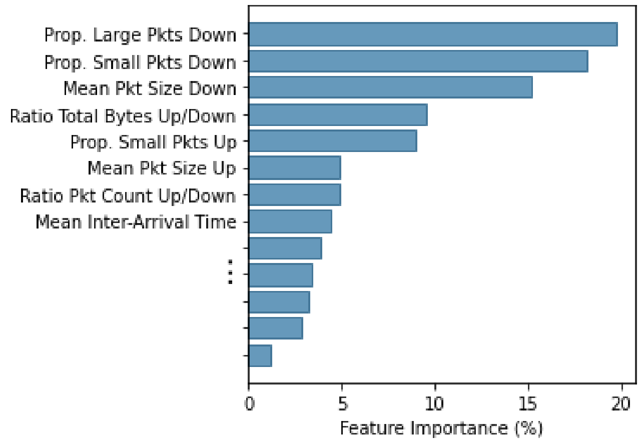
We rely on statistical tests to further analyze the difference between the median and mean FNR in each group. A one-way ANOVA test for differences in means between all groups produces a large F-statistic and corresponding p-value of practically zero; between the three conditions likewise results in a p-value of practically zero. Since there appear to be many outliers, and ANOVA assumes normality—an assumption which we appear to break—we also use a Kruskal-Wallis test which is non-parametric and has considerably less assumptions about the data. A Kruskal-Wallis test for differences in medians between all groups results in a large H-statistic and corresponding p-value of practically zero; between the three conditions likewise results in a p-value of practically zero. These results suggest that differences in FNR between the conditions are highly statistically significant—consistently different—even though the scale of that difference is relatively small.

5. DISCUSSION

Our results confirm our hypothesis that different network conditions result in inherently different traffic data, which in turn exhibits a clear, statistically significant effect on classifier performance. That said, even though the effect is statistically significant, the absolute difference in FNR of less than 0.015 is not a substantial impact.

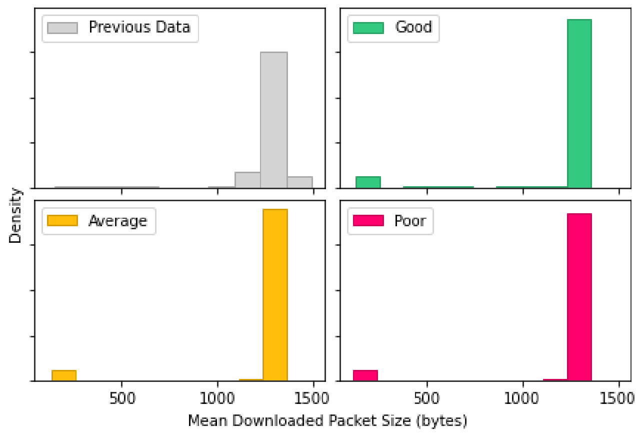
To understand why the classifier still performs well in multiple conditions, we need to examine the features used by the classification model. After 500 runs, the feature importances of the classification can be calculated as the normalized average feature weight.

Fig 4: Feature Importances of Streaming Classifier



Notably, five of the top six most important features are features derived from packet size. Maximum packet size is determined by the maximum transmissible unit (MTU) of a client-server connection which is typically 1500 bytes for the vast majority of consumer-grade networking infrastructure, and is not influenced by network conditions like latency and bandwidth. Because maximum packet size is fixed across network conditions, we hypothesize that the feature distributions of features based on packet size will not be affected by condition. We can explore this hypothesis by directly comparing feature distributions across the three conditions and the previous dataset.

Fig 5: Feature Distribution of Mean Downloaded Packet Size by Condition (Streaming)

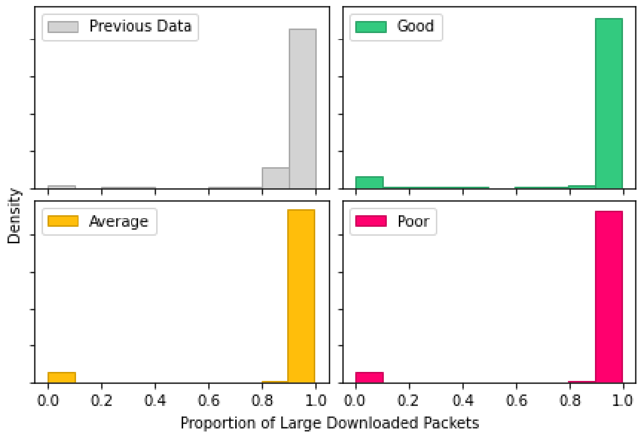


Examining the mean size of downloaded packets—the third most important feature—across all preprocessed streaming data samples, we see a very similar distribution arise in each condition and in the previous dataset. The distributions in the three conditions are nearly indistinguishable, suggesting that indeed network condition has little to no effect on downloaded packet size. The slight difference between the distribution in the previous dataset is most likely attributable to behavioral noise in the previous dataset. Because the previous dataset was collected manually, any active background services or any other web activity the user engaged in while collecting data would also be labeled as streaming, even if a different distribution of data was produced. The newly generated dataset was automated to produce consistent and isolated streaming behavior data.[1]

[1] Admittedly, behavioral noise is perhaps a beneficial component of a traffic classification dataset because it is likely to be exhibited in real-world data. Once again, the automated collection tool is limited by the ability to replicate human-like behavior with coded scripts (i.e., humans are still the limiting factor!).

An examination of the proportion of downloaded packets larger than 1200 bytes across all samples—the most important feature—lends itself to the same conclusion as above.

Fig 6: Feature Dist. of Proportion of Large Downloaded Packets by Condition (Streaming)



Once again we observe that this feature distribution, which is also derived from individual packet sizes, is nearly indistinguishable across the three conditions, and differs minimally from the previous dataset. We will not replicate the analysis of means and proportions for packets in the upload direction nor small packets (less than 200 bytes) in this paper since their results are consistent with what we have already discussed.

Finally we look at two similar features which aren't derived from packet size but rather the total amount of data or packets downloaded in a sample. The distribution of the ratio of total bytes downloaded versus uploaded appears nearly identical across all groups. However, the ratio of the count of uploaded versus downloaded packets starts to show some variation.

Fig 7: Feature Dist. of Ratio of Uploaded vs. Downloaded Bytes by Condition (Streaming)

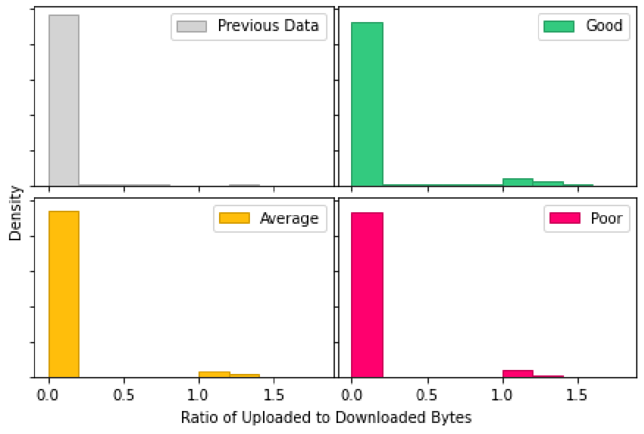
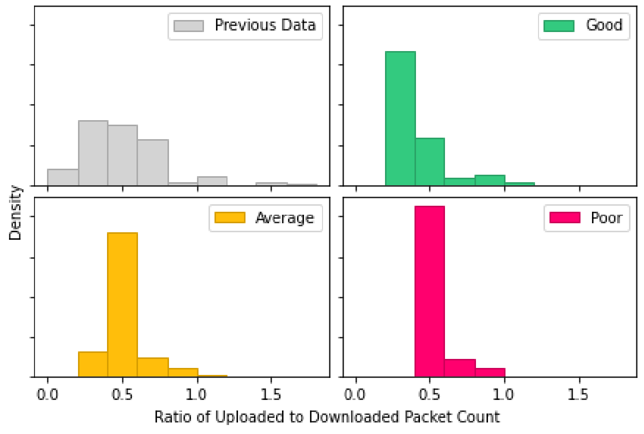


Fig 8: Feature Dist. of Ratio of Uploaded vs Downloaded Packet Counts by Condition (Streaming)



We have already seen in Figure 2 that network conditions directly influence the amount of packets of data which can arrive within a given time frame. Therefore, we expect that the total downloaded bytes and total number of downloaded packets differs by condition. However, most streaming providers send and receive packets using the transmission control protocol (TCP), in which clients periodically upload small packets containing acknowledgements (ACKs) that content from the server has successfully been received. Because large amounts of data need to be downloaded for streaming, and acknowledgements are small, we see in Figure 7 that regardless of the network condition the ratio of uploaded to downloaded bytes in a sample of data is most dense at zero. The ratio of packet counts shown in Figure 8 does not depend on the size of each packet, however, and we start to see slight variance between the network conditions. As this feature accounts for a 5% decision weight in the streaming classifier, such variance between the network conditions likely has a small but noticeable effect on classifier performance which may have contributed to the findings in Figure 3.

The streaming classifier which was developed and trained on the previous dataset relies most heavily on features which we have demonstrated above are robust to changes in network condition. Thus, it is understandable that the effect of network condition on classifier performance is relatively small.

6. CONCLUSION

In this paper, we showed that differences in network conditions can fundamentally change the way network traffic data looks. We measured the effect of network conditions on performance of a pre-trained streaming classifier, and determined that there is a statistically significant difference between performance when the model is evaluated on the different conditions. However, we saw that the overall impact on model performance was still small. This reveals that if a classifier uses features which are robust to changes in network conditions—such as packet size—then the effect on the model performance is minimal.

7. REFERENCES

[1] Cao Z., Xiong G., Zhao Y., Li Z., Guo L. (2014). "*A Survey on Encrypted Traffic Classification*", In: Batten L., Li G., Niu W., Warren M. (eds) Applications and Techniques in Information Security 2014. Communications in Computer and Information Science, vol 490, pp. 73-81. Springer. https://doi.org/10.1007/978-3-662-45670-5_8

[2] Draper-Gil G., Lashkari A.H., Mamun M., Ghorbani A. (2016). "*Characterization of Encrypted and VPN Traffic using Time-related Features*", ICISSP. https://doi.org/10.5220/0005740704070414

[3] Crotti M., Dusi M., Gringoli F., Salgarelli L. (2007). "*Traffic Classification through Simple Statistical Fingerprinting*", In ACM SIGCOMM 2007. https://doi.org/10.1145/1198255.1198257

[4] Miller S., Curran K., Lunney T. (2018). "*Multilayer Perceptron Neural Network for Detection of Encrypted VPN Network Traffic*", In 2018 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA), Glasgow, pp. 1-8. https://doi.org/10.1109/CyberSA.2018.8551395

[5] Laubach C. (2020) "*network-stats*", GitHub repository. https://github.com/Viasat/network-stats

[6] Addison P., Altekar S., Yaseen D. (2021). "*DANE: Data Automation and Network Emulation Tool*", GitHub repository. https://github.com/dane-tool/dane

[7] Y. Yang, N. Vlajic and U. T. Nguyen. (2015). *"Next Generation of Impersonator Bots: Mimicking Human Browsing on Previously Unvisited Sites"*, In IEEE 2nd International Conference on Cyber Security and Cloud Computing, New York, NY, 2015, pp. 356-361. https://doi.org/10.1109/CSCloud.2015.93