# Capstone Project: Viasat Streaming/Non Streaming Network Data Classifier

Arely Vasquez
arv020@ucsd.edu

Andrey Pristinsky
apristin@ucsd.edu

December 11, 2020

**Abstract**

A Virtual Private Network (VPN) is used by individuals when performing activities such as playing video games, browsing the internet, or even streaming a movie on Netflix. With a VPN, this network data is encrypted, making the activity of the user private and initially difficult to understand. Since streaming network data has a unique fingerprint and behavior compared to non streaming network data, we are investigating whether a machine learning model is able to classify if the network data contains streaming in its activity while in a VPN tunnel. By analyzing the packet sizes in bytes, direction of packets such as whether upload or download, and peaks of packets, we are able to use a Random Forest Classifier in order to classify input network data.

## Introduction

The problem we are investigating consists of whether we can use machine learning to predict when a network user is streaming video through a Virtual Private Network (VPN). For example, if someone is playing video games or browsing the internet over a VPN tunnel, this will provide data about a user that you would not typically be able to have access to or understand as it is encrypted. Since streaming network data is unique in its behavior compared to non streaming network data, we are able to analyze this problem in depth. In previous work, the same problem was approached by analyzing and comparing the bursts of packets in the network data, building classifiers with embedded in-depth approaches, and understanding the distinct behaviors between streaming activity [1, 3]. Another unique approach for encrypting internet traffic was to transform flow data into a picture similar to image recognition, a FlowPic, which was shown to have impressive accuracy [2].

With that being said, the data for this problem was captured by running a program called network-stats, provided by Viasat, which allowed us to capture data while connected to the internet and streaming videos in 5 minute intervals. While collecting this data, we changed a  variety of variables throughout the videos such as the resolution, speed, and the different providers and formats. Different providers included Netflix, Youtube, Hulu, Amazon Prime and more. All the data collected for this analysis was collected while connected to a VPN tunnel.

The observed data that we acquired from running the Viasat network-stats showed us the amount of data in packets being sent back and forth between the computer and the target destination. Specifically, the observed data consists of the number of bytes being transferred, the number of packets that contained those bytes, the protocols used such as UDP (User Datagram Protocol) or TCP (Transmission Control Protocol), the internet providers' encrypted address of both the source and destination, and the time of when the data was sent back and forth.

The most significant feature observed was the number of bytes being transferred between the source and destination since large spikes of bytes being sent or received can help indicate that a streaming service is being used, such as when the video buffers. This observed data is appropriate for addressing the problem because through machine learning, the usage of these features should be enough to provide a conclusive analysis as long as we gather enough data to train our machine learning algorithm appropriately. If certain features aren't up to par, we can use feature selection to obtain the best features and provide the highest possible accuracy for addressing the problem of using machine learning to predict when a user is streaming videos over VPN.

## Methods

The experimental design that was conducted consisted of initially cleaning the data, analyzing the data, feature engineering, feature selection and ultimately creating a Machine Learning Model in order to classify if an instance of VPN usage contained streaming video or not.

To begin with, the data was initially cleaned by separating every instance of a packet being transferred and its packet size in bytes. From there we were able to analyze the individual packet sizes and the behavior it entailed, such as the direction of packets, the timing in which they arrived, and the occurrences of certain packet sizes. With that in mind, by plotting and visually understanding the behavioral pattern between streaming and non streaming network data, we then looked at the frequency of packet sizes from different ranges as seen in Figures 1 and 2. There were some ranges that appeared distinct from streaming and non streaming data. These ranges included frequency of packet sizes in 0-200 bytes, 200-400 bytes, and 1200-1500 bytes.

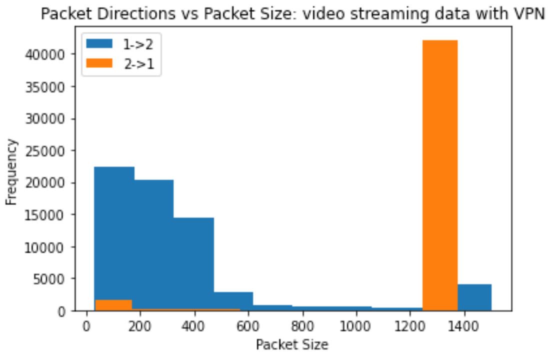


**Figure 1:** Frequency of packet sizes (in bytes) for streaming data. This data collection included streaming a video from Youtube. The blue represents the frequency of uploads. The orange represents the frequency  of downloads.

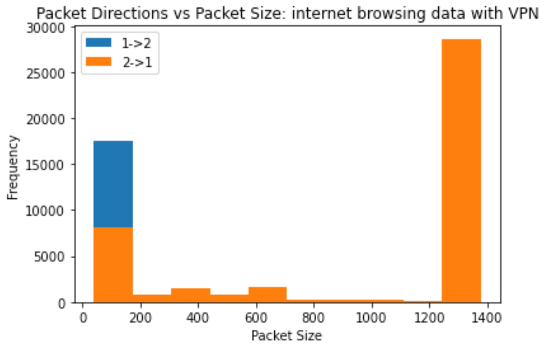


**Figure 2:** Frequency of packet sizes (in bytes) for non streaming data. This data collection included internet browning. The blue represents the frequency of uploads. The orange represents the frequency of downloads.

First, we found that there is a greater frequency of larger packet sizes compared to the overall packet sizes in streaming data as seen in Figure 1. A majority of the packet sizes are all 1200+ bytes in streaming data. When looking at the occurrences of packet sizes in the ranges mentioned previously, this analysis was a strong identifier that streaming usually occurs when there are more occurrences with the 1200-1500 byte range vs web browsing when there is a greater frequency within the 0-200 byte range.

We turned this into two features, where one direction was more focused on the lower range of bytes while the other direction was focused on the larger range of bytes. By doing so, these two features are impactful in predicting streaming vs non-streaming since the occurrence of certain packet sizes are a key identifier in classifying the data accordingly. The other key identifier was another version of the ratio of packet sizes. This included looking at the ratio of packet sizes that were between 0-300 bytes and comparing that number to the overall number of packets. This gave us an understanding around what percent of packets were on the smaller size range.

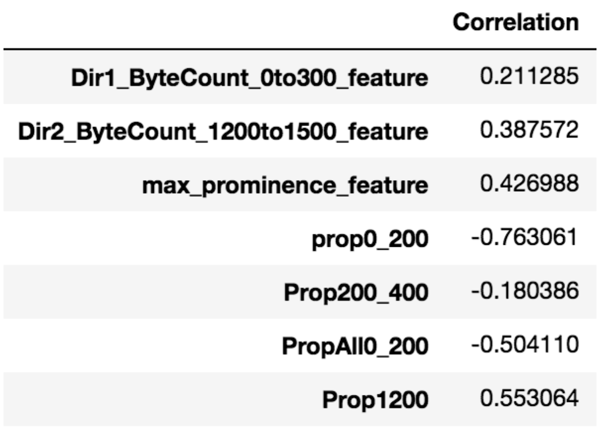
In addition to analyzing the byte sizes of packets, we decided to transform the data using spectral analysis. After resampling every 500 milliseconds, we used Welch's method to sample the data at a frequency of 2Hz. Now that the data was in the frequency domain, we decided to apply scipy's find peaks function to identify all of the spikes of data transfer occurring. From this, we noticed that the peaks in video packet sizes appeared much more prominent than the peaks in the no video packet data. These differences were found based on the byte packet sizes coming in from the 2 to 1 direction, which led us to believe that analyzing the prominence of peaks would be a useful tool to have as a feature. Therefore, we created a feature that selects the max prominent peak present within a dataset since there appears to be a significant difference in the prominence of peak sizes between streaming and no streaming.

From this discovery in our exploratory data analysis, we were able to construct features that were a strong indication of whether the data included video streaming with this knowledge and understanding. After constructing key features that are strong indicators in identifying streaming network data, we then tested and studied the various machine learning algorithms in order to identify the best classifier. Using the features above and a Random Forest Classifier, we constructed the streaming and non streaming classifier.

## Results

The final model consisted of a Random Forest Classifier. The model performance was measured by looking at the test data accuracy. The test data accuracy we acquired for our model reached 89.3%.

In terms of the features incorporated in the model, the correlation coefficient of each feature was taken into account. Chart 1 below shows the correlation coefficients of each feature to the target variable. The strongest negatively correlated feature is the proportion between upload and download packets from range 0-200 with a -.076 correlation coefficient. Whereas the strongest positively correlated feature consists of the proportion of packet sizes of 1200+ bytes between upload and download with a 0.55 correlation coefficient.



| | Correlation |
|---|---|
| Dir1_ByteCount_0to300_feature | 0.211285 |
| Dir2_ByteCount_1200to1500_feature | 0.387572 |
| max_prominence_feature | 0.426988 |
| prop0_200 | -0.763061 |
| Prop200_400 | -0.180386 |
| PropAll0_200 | -0.504110 |
| Prop1200 | 0.553064 |

**Chart 1**: Correlation Coefficient of each feature and the target variable

From the sklearn model selection package, we utilized GridSearchCV on the model. This enabled us to select the best parameters for the classifier in the model in order to maximize performance and accuracy. The grid search process looked at a variety of different parameters with a cross validation of 5. With that being said, only the main parameters were taken into account such as criterion, max_depth, min_samples_split, and lastly n_estimators.

Another way of measuring performance of the classifier was by conducting a confusion matrix. Chart 2 displays the confusion matrix for the test data. We can see that from the 28 test datasets, that 25 were predicted accurately while 3 were misclassified as False Positive.

| | **Predicted: No** | **Predicted: Yes** |
|---|---|---|
| **Actual: No** | TN = 3 | FP = 3 |
| **Actual: Yes** | FN = 0 | TP = 22 |

**Chart 2:** Confusion matrix for Test Data

## Discussion

From the results of the final classifier, the strongest positively correlated feature consists of the proportion of packet sizes of 1200-1500 bytes between upload and download. This indicates how a large part of the distinct fingerprint between streaming data and non streaming data is the large packet sizes that are present in the streaming data. On the other hand, the strongest negatively correlated feature is the proportion between upload and download packets from range 0-200. This is a strong indication that non streaming behavior is different than streaming. The larger the ratio is of small packets, the more likely that the data did not contain any case of streaming. As well, by looking at the data in the frequency domain, maximum prominence was identified as the second strongest positively correlated feature. There was a significant size difference of the max prominence values between streaming vs no streaming, that definitely helped our classifier achieve the 89% accuracy rating. While the other features also helped in accurately predicting the class of the network data, packet sizes and maximum prominence have the largest influence.

Compared to prior research and work on this problem, one of the main characteristics of streaming activity that made it distinct from non streaming activity, were the bursts of packets [1]. With the understanding of this from prior work, we were able to go more into depth with looking at the maximum prominence and creating a feature of this. We also looked at the proportions of packet sizes that were also a large indicator of streaming which made our analysis different.

An example of how our classifier can be used in a real life scenario and utilized outside of the classroom includes that of an ISP (Internet Service Provider), such as Viasat, to use this classifier to identify if a customer is streaming while using their internet service while also utilizing a VPN. This can allow Viasat to have a better understanding of their customers and their customer needs from them. In a larger sense, this can help Viasat by optimizing and enhancing their services, as well as increasing their customer satisfaction.

One of the underlying issues of building this model was understanding the ethical concerns of it. When internet users use a VPN, they primarily use it for the reason that it keeps their internet data private and encrypted. The purpose of the model is to decrypt the data to indicate whether a user is streaming while on a VPN. Although the primary use of this model is to help the customer by informing the ISP of the customer's own data, there are still some ethical considerations when approaching the purpose of this classifier.

The final classifier that we developed achieved a test accuracy performance of 89.3% . With the training accuracy being 100%, this is a bit worrisome knowing that the model is most likely overfitting. Although the accuracy rates are sufficient, you can almost say that they are suspiciously too high. This goes hand in hand with the limitations of the model.

A limitation of our model includes how the training data of the model was not diverse enough in terms of non streaming data. For example, most of our non streaming instances included only browsing the internet. Although there were only a few instances where this included loading small ads, smaller content videos. For example, scrolling through Tiktok or Instagram is not considered streaming, but it does take more bytes to load up small videos that are posted on peoples' feeds. With that being said, we need to address the limitations of the model and how it might not perform as well when predicting instances like these. Not to mention, there was an imbalance between streaming and non streaming in the training data. This is noticeable on the confusion matrix when the model is only trained on 22 non streaming datasets and 60 streaming datasets. In order to improve the credibility of our model, we would need to collect more data in order to build a well-rounded model.

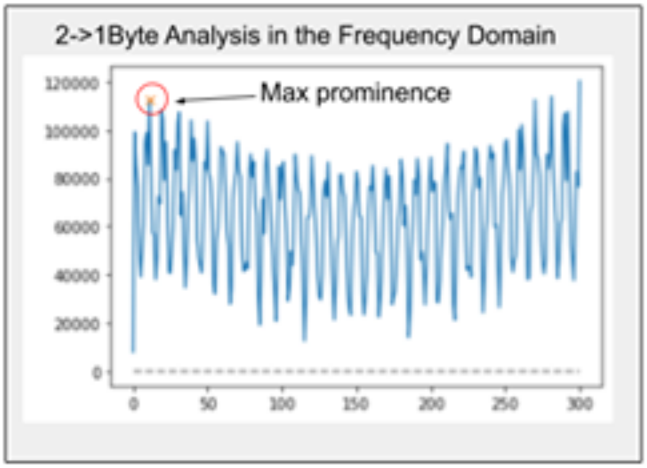
After seeing that it is possible to take the unique signature of streaming network data vs non streaming network data, it is possible to further this classifier and go beyond this problem. Some future work and an extension to this problem can be being able to detect what streaming provider the VPN user is streaming from. This could be Netflix, Hulu, Youtube, or any other streaming provider. Another possibility would be to be able to differentiate between a live stream and a regular streaming instance. Different instances have different patterns that make them unique, so by analyzing that from network data, this could be a possibility to continue to further the initial project and continue to improve an ISP.

## References

[1] Roei Schuster, Vitaly Shmatikov, Eran Tromer. Beauty and the Burst: Remote Identification of Encrypted Video Streams. 2017.
[2] Tal Shapira, Yuval Shavitt. FlowPic: Encrypted Internet Traffic Classification is as Easy as Image Recognition. In *IEEE* 2019. [3] Guorui Xie, Qing Li, Yong Jiang, Tao Dai, Gengbiao Shen, Rui Li, Richard Sinnott, Shutao Xia. SAM: Self-Attention based Deep Learning Method for Online Traffic Classification. In *NetAl* 2020.

## Appendix

*a. Maximum Prominence*



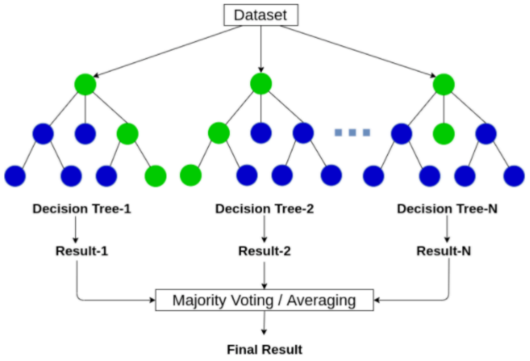
**Appendix Figure 1:**

The image above shows a single dataset in the 2 ->1 direction for the Byte sizes in the frequency domain.

Maximum prominence refers to the point circled, which is the largest prominence detected for a peak out of all the peaks and prominences present. This is what was used for the creation of one of our features. Through scipy.welch and scipy.find_peaks, we were able to reach this conclusion. For more information on how these scipy functions work:

- https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.welch.html
- https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.find_peaks.html

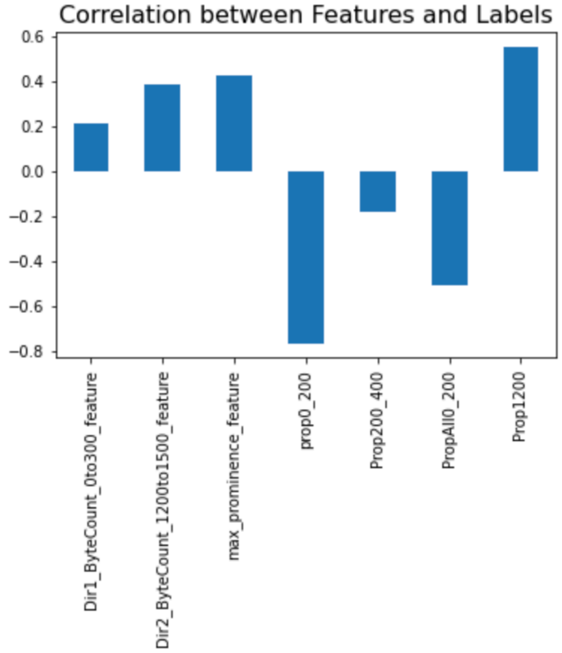
### *b. Random Forest Classifier*



**Appendix Figure 2:** Random Forest Classifier

The image above shows the general algorithm and process for the machine learning model of Random Forest Classifier. Multiple decision trees are constructed in the model and taken into account when outputting the result. For more information on the Random Forest Classifier: [https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html) [https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/](https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/)

### *c. Final Classifier Model Coefficient Correlation*



**Appendix Figure 3:** Correlation coefficient between each feature and the target variable.