15 December 2023

**APPLYING MACHINE LEARNING TO CONGESTION CONTROL**

**ABSTRACT**

​​In today’s ever evolving world, there are more devices uploading data to the web than ever before. With this growth comes the need to improve networks in order to not fall behind the increase in demand. There are a fair amount of methods researchers have investigated to resolve this issue. One of which is finding new, profound ways to optimize congestion control mechanisms. The topic of congestion control alone has various approaches that could be optimized. In this paper, the topic of packet loss classification will be discussed.

**1 INTRODUCTION**

​​This research project focuses on the intersection of congestion control and machine learning, exploring the viability of employing advanced classification techniques to identify and isolate malicious packet data within network traffic. Specifically, the study aims to evaluate the effectiveness of machine learning algorithms as classifiers for distinguishing between normal and malicious packets, with the ultimate goal of enhancing network security through proactive congestion control as well as performance by mitigating potential traffic.

The significance of this research lies in the potential to not only improve the efficiency of congestion control mechanisms but also to improve network security by identifying and isolating malicious activities at an early stage. Traditional methods of congestion control often rely on reactive measures, responding to network congestion after it has already occurred. By incorporating machine learning into congestion control strategies, we aim to move towards a more predictive and preemptive approach, where malicious packets can be identified and mitigated in real-time, minimizing the impact on network performance.

**2 Related work**

​​At the start of this research journey, I was planning on implementing a new, machine learned, protocol that was based off of TCP network data but overall improving throughput. This was the goal, but as I was looking through the research on congestion control implementations and machine learning approaches, this project would seem to be very time consuming and too muc of a time investment for a month long research project. As I was looking into other types of approaches I found the articles listed below.

Recently, as stated by research at Xidian University, in recent years there are almost 375 whitepapers published each year with “congestion control” in their title[1]. From one of these research papers, I found an article that studied how different ML algorithms classify link errors distinctly from congestion loss[2]. The researchers created a simple TCP network and monitored various packet metrics. The classification process was utilized when a triple duplicate packet was recognized by the system, meaning that the packet was retransmitted twice. When this duplicate packet was identified, its metrics were then analyzed by a machine learning to classify the packet as either benign or malicious.

The study experimented with 8 different machine learning algorithms and came to the conclusion that although a fair amount of these algorithms performed well, the decision tree boosting algorithms performed best for their situation. This IEEE paper was my main source of inspiration, because at the very least, a binary classifier seemed like the simplest place to start this research and build off of the classifying algorithm from there.

My attempt in this project is to prove the efficacy and importance of machine learning in today’s network systems by building off of the knowledge that has been published.

**3 Approach**

​​The approach of this project was broken into 3 main parts, data collection, machine learning model research and application, compiling results and reiterating evaluation process. This seemed like a reasonable and simple approach to a possibly much larger project. My purpose of this was to create a basis or foundation to build off of with more time and deeper investigation.

To start, copious amounts of packet data was preferred. The first approach I considered was creating synthetic data through python simulations using Simpy to create a somewhat realistic dataset, at the very least, the data would be fairly replicable to evaluate with testing data. Some time later, I consulted with a mentor who works in the telecommunications business who is working on machine learning applications in this industry. He noted to me that synthetic data does not promote as vigorous basis for results that I am attempting to prove.

The next approach was researched large public datasets related to networking. After a fair amount of hours looking over public datasets on the IEEE dataset archive, I found a public dataset published by the Canadian Institute for Cybersecurity Intrusion Detection Datasets, CIRA-CIC-DoHBrw-2020 [3]. This data was extremely detailed, including a csv file for benign data and malicious data. This solidified that this was the approach I was attempting. The data also included 30 features on the packet data including median packet length, average round trip time, and actual duration of the packet being sent. This dataset provided over 800,000 entries, each one of these entries being a separate packet.

Next, I needed to determine the best ML model to implement as a basis. At first, I was going to use the Decision Tree Boosting algorithm as the underlying paper utilized. This algorithm seemed to be a complex method to start with. I researched a more simple model and came to the conclusion to begin with a machine learning model known as a One-Class Support Vector Machine also known as a One-Class SVM. This model is a binary classifier, meaning that it attempts to bin data into two labels, positive and negative. The details for the model are provided below:

**Objective Function**



**Decision Function**



* *w* is the weight vector.
* *b* is the bias term.
* *ξi*​ are slack variables.
* *ϕ*(*xi*​) is the mapping of the input *xi*​ into a higher-dimensional space using the chosen kernel function.
* *ν* is a user-defined parameter controlling the fraction of margin errors (anomalies).
* *n* is the number of normal instances in the training set.
* *ρ* is the radius of the hypersphere around the origin (centered at the decision boundary).

If the result is positive, the instance is classified as a normal inlier, if the result is negative, the instance is classified as an anomaly (outlier).

Now that I had a model in mind, I formatted the data, merged the benign and malicious data together, and ran a few hundred entries through the model. The results were not great, around 54% accuracy on my first few runs. But over time, I was able to tweak hyperparameters of the model and change what features the model was evaluating. With these changes, the model was able to achieve 70% accuracy. This is where the model started proving effective and now it was time to apply various models on this data to determine which model works best for this type of process or this type of data.

Once this ML model proved effective, I would test other machine learning models and see how they compare to the One Class SVM.

**4 Results**

**A graph of a graph

Description automatically generated with medium confidence**

​​​​When only the OneClassSVM was being tested on the data, the best results gathered were 70% accuracy, trained on 5,000 entries and tested on 1,000 entries. Although better results were expected, this proved useful as the data had some distinction in patterns that a basic model was able to distinguish between.

Next was to explore other ML classifier models and compare their performance to my initial attempt with the One Class SVM. The other classifiers I examined were: Isolation Forests, Local Outlier Factors, Elliptic Envelope, KMeans, Decision Tree Classifier, Random Forest Classifier, and Logistic Regression.

As seen from the ‘Model Accuracies’ figure, the Elliptic Envelope and One-Class SVM showed the most promise for a malicious packet classifier. Both of these models performed at 79% accuracy, trained on 20,000 entries and tested on 5,000 entries.

**5 Conclusion and future work**

​​Prioritization is vital in time management as it helps individuals focus on what truly matters, allocate resources efficiently, and ensure that important tasks are completed in a timely manner.​

**​​Tips and techniques for success​**

1. ​​Evaluate tasks based on their importance and urgency, focusing on high-priority items that align with your goals and deadlines.​
2. ​​Use techniques like urgent vs. important to classify tasks and prioritize them accordingly.​
3. ​​Regularly reassess priorities and adjust as needed to ensure you're allocating your time and energy to the most crucial tasks.​

**References**

​​

[1] Wei, Wenting, et al. “Congestion Control: A Renaissance with Machine Learning - Iqua.”,

iqua.ece.toronto.edu/papers/survey-conges tion-control-ieee-network.pdf.

[2] Geurts, P., et al. “A machine learning approach to improve congestion control over wireless

computer networks.” Fourth IEEE International Conference on Data Mining (ICDM’04), 2004, pp. 2–

3, https://doi.org/10.1109/icdm.2004.10063.

[3] “Search UNB.” CIRA-CIC-DoHBrw-2020, University of New Brunswick , 2020,

www.unb.ca/cic/datasets/dohbrw-2020.html.

[4] https://github.com/ty-runner/ECE-547-Sims/tree/main/ResearchProject

[5] “One-Class Support Vector Machine.” XLSTAT by Lumivero, www.xlstat.com/en/solutions/features/1-class-support-vectormachine#:~:text=The%20One%2Dclass%20Support%20Vec tor,as%20th e%20class%20of%20outliers