Enhancing Network Anomaly Detection: A Two-Stage Approach with Neural Networks for Enhanced Security and Beyond

Mel Avina-Beltran, Kent Cassidy, Zahira Ghazali, Jose Navarro, Connor Bondoc
ECS 170, University of California, Davis

September 13, 2023

I. Introduction

Our final project in ECS 170 focuses on the reimplementation of autoencoders for anomaly detection. This project aims to showcase the application of artificial intelligence (AI) principles in the context of unsupervised learning to identify anomalous patterns in network traffic. The primary motivation behind this project is to enhance cybersecurity measures by effectively detecting network anomalies, thereby mitigating potential threats.

II. Background

To fully comprehend our project, it is crucial to acquire a fundamental understanding of several key concepts. Firstly, anomaly detection is central as it involves the identification of patterns within a dataset that diverge from the anticipated or typical behavior. Especially in the context of cybersecurity, anomalies can be indicative of attacks or system failures. Generally, neural networks, which draw inspiration from the structure of cell behavior within the human brain, are a foundational computational model used across a wide spectrum of cybersecurity applications and are particularly useful at targeting anomalies.

Artificial Intelligence plays a pivotal role in enhancing cybersecurity throughout its various domains. In the realm of threat detection, it is leveraged for identifying potential security breaches and anomalies in real time, ensuring proactive defense mechanisms where a human administrator is unable to. AI aids in user behavior analysis, scrutinizing patterns and deviations to detect unauthorized access or suspicious activities. Moreover, it empowers automated response systems, enabling swift reactions to security incidents. AI's contributions to cybersecurity extend to supervised machine learning, where it excels in tasks such as malware detection, spam email filtering, and intrusion detection by learning from labeled data. Various unsupervised machine learning techniques, including anomaly detection and cluster analysis, enhance cybersecurity by identifying irregularities and grouping similar entities within data, facilitating a more comprehensive understanding of potential threats.

An autoencoder is a unique neural network variant specifically designed to replicate its input as closely as possible at its output. This fundamental characteristic of abstract mimicry makes it distinct from other neural network architectures. Autoencoders are primarily used for unsupervised learning tasks, where they excel at capturing and representing the essential features or patterns within data. By encoding the input into a lower-dimensional representation and then decoding it back to its original form, autoencoders enable dimensionality reduction, feature extraction, and reconstruction. This capability finds applications in various fields, including image processing, data compression, and anomaly detection, making autoencoders a

valuable tool in machine learning and artificial intelligence.

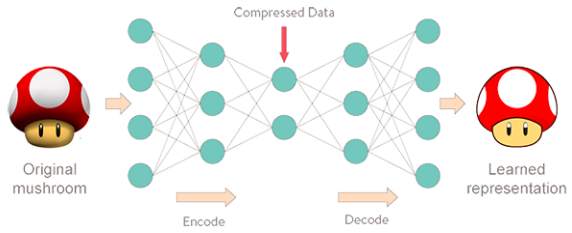


Figure 1: Example Autoencoder [1]

Understanding autoencoders is crucial, especially in the context of unsupervised learning for cybersecurity. Autoencoders offer adaptability to evolving cybersecurity trends and reduce the reliance on labeled data, making them invaluable tools for anomaly detection and threat mitigation. In practice, their application involves training the autoencoder on normal data, allowing it to learn the encoding and reconstruction of this data. This knowledge can be harnessed for various cybersecurity tasks, including denoising data, identifying anomalies, and triggering alerting and response mechanisms when potential threats are detected. This practical utilization underscores the significance of autoencoders in bolstering cybersecurity defenses.

III. Methodology

In this section, we delve into the methodology employed for our project:

- *Dataset Selection*—We initially planned to reimplement the model created by a UCI research study that utilized the N-BaIoT dataset, which contains network traffic data from nine devices infected with Mirai and BASHLITE malware. Our goal was to create a convolutional neural network (CNN) or another type of supervised learning model to predict anomalous or benign network traffic using the features.

  However, we later understood that the original research study had utilized the N-BaIoT dataset to create an unsupervised learning model - specifically an autoencoder with a threshold for anomaly detection. As such we decided to try and recreate this model leading to our implementation of an autoencoder using this data.
- *Autoencoders for Anomaly Detection*—We implemented an autoencoder neural network model to encode and decode benign traffic data. In this way, the neural network could then recreate benign traffic data with low reconstruction error but would recreate anomalous data with very high reconstruction error. We could then use a threshold on this level of error to distinguish whether the data was representative of benign network traffic or anomalous network traffic.
- *Models*—We implemented three different models. Each subsequent model after the first was supposed to fix issues with accuracy and loss that the first had and did so to some effect. Each utilized the neural network to reconstruct benign data and then a threshold was created from the level of training loss experienced solely on benign data. This threshold was then applied to two datasets - the Mirai and BASHLITE datasets - which had all the data from the specific attack plus the testing benign dataset comprising the last 6510 rows.
- *Model 1: Unscaled Simple Model*—The model used for Model 1 (seen below) only consisted of two layers and was run on unscaled data. This model was severely underfitted as seen by the Training Loss vs. Validation Loss graph and had loss in the trillions (1e14). Because the autoencoder was trying to recreate the benign data exactly, the training loss and validation loss shown

here are the loss across all features in the reconstruction that is decoded from the neural network. The model was extremely underfit so it wasn't able to properly reconstruct the benign training data which led to poor performance when using the threshold to detect anomalous traffic in both the Mirai and BASHLITE datasets. As such, we wanted a model that was better capable of reconstructing the benign training set so we opted to normalize the datasets to create a better model which led to Model 2.

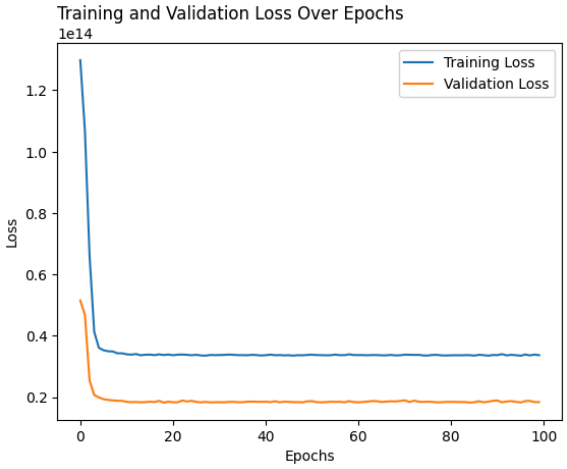


Figure 1: Training Loss vs Validation Loss for Model 1

- *Model 2: Scaled Simple Model* — The model used for Model 2 only consisted of two layers and was trained on benign data that had been normalized using a min-max scaler. This model, opposite to the first, was overfitting as seen by the Training Loss vs. Validation Loss graph and also had a shockingly low accuracy at predicting the validation set while training but was able to predict the scaled training benign data with high levels of accuracy. However, its lack of ability to predict the test benign data meant that, it too, couldn't reconstruct the benign data correctly which led to poor performance when using the threshold to detect anomalous traffic in both the Mirai and BASHLITE datasets. Since Model 1 appeared to have better performance than Model 2, we opted to try to fix the underfitting present in Model 1 to increase both its level of fit and accuracy which led to Model 3.

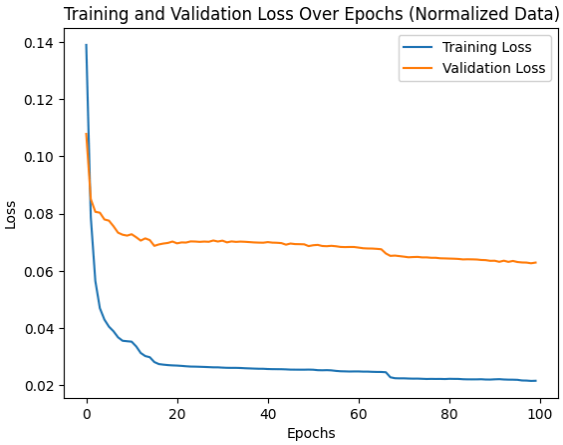


Figure 2: Training Loss vs. Validation Loss for Model 2

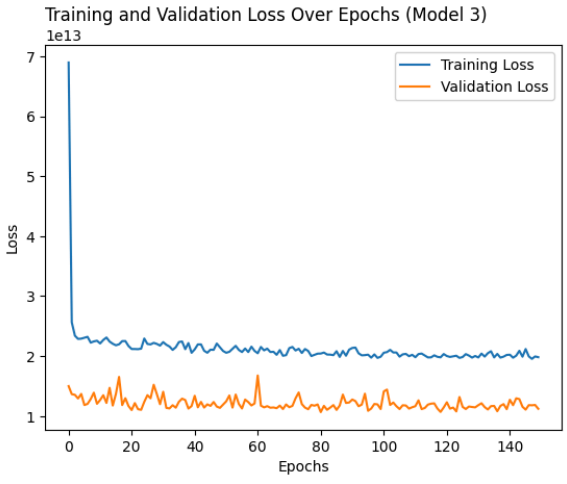
- *Model 3: Unscaled Complex Model* — The model used for Model 3 consisted of several layers and was trained on benign data. This model, similarly, to the first was severely underfitted though to a lesser degree as seen by the Training Loss vs. Validation Loss graph (1e13 as opposed to 1e14) due to the changes we made to help reduce the underfitting.

  We added more layers to the model as well as increased the number of epochs to increase the amount of time spent in training. While it's not also shown in the notebook or here, we did try tweaking the loss function (using mean squared error or Hubert loss instead of mean absolute error), added and removed normalization layers, altered the optimization function used before reverting it back, as well as changed up the activation function used

for the layers or added layers with different activation functions.

While increasing the number of epochs and layers did seem to increase the accuracy, none of the other changes caused a sizable improvement in accuracy or fit. The activation function that appeared to work best was “relu”. While we had higher expectations for this model it did still have a poor accuracy rate when using the threshold. The model did seem to get a better level of accuracy in terms of reconstructing the benign data so we tried to make alterations to the value used for the threshold in order to increase the level of accuracy in distinguishing between benign and anomalous network traffic.

Since the threshold was based on the training loss (the average training loss plus one standard deviation) and the training loss was so high we altered the threshold by decreasing it by one standard deviation or a percentage of the standard deviation which then improved performance. It was also because of this that we realized that, to further improve accuracy, we have to go back to the initial model and find a way to reduce



training loss further for the model to perform the classification better.

Figure 3: Training Loss vs. Validation Loss for Model 3

IV. Results

The results of our project can be summarized as follows:

- *Model Performance*: While our initial models faced challenges, increasing the number of layers and the amount of time spent in training improved the models overall performance at reconstructing the benign data. However, due to the level of training loss experienced by the models, it was difficult to set a threshold that allowed for greater level of accuracy in distinguishing between benign and anomalous data.
- *Model Accuracy*: We tested our model on two datasets: BASHLITE and Mirai. Both datasets had anomalous network traffic except for the last 6510 rows which were the test benign data. The accuracy of each of the models were as follows:

Model #	Mirai Accuracy	BASH Accuracy
Model 1	11.18%	15.32%
Model 2	89.04%*	85.09%*
Model 3	11.2%	15.32%

*For Model 2 it classified all network traffic as anomalous; considering the Mirai and BASHLITE datasets only had 6510 rows with benign data where it had to predict True, this model was able to classify all the anomalous cases “correctly” due to its inability to classify benign data. With a smaller dataset for Mirai and BASHLITE cases but the same number of benign cases, the

accuracy went down to 8.4% accuracy for BASHLITE and 27.6% accuracy for Mirai.

V. Discussion

Right now the current loss for all three models comes from loss across all the features as the autoencoder is trying to reconstruct the benign data exactly. It thus may be a good idea for future improvements to reduce the number of features or remove features that have extremely large values that may be contributing large amounts of loss. We also learned about the importance of unsupervised learning models in the field of cybersecurity especially in the concept of anomaly detection. While we were unable to try more changes to the models due to time restraints, a prospective fix for Model 2 would be to include an ensemble or even drop columns as mentioned previously.

VI. Conclusion

In conclusion, our project highlights the potential of autoencoders in the realm of anomaly detection within network traffic. While we encountered challenges along the way, we successfully transitioned from supervised to unsupervised learning and improved our model's performance through normalizing the data, increasing the length of time spent in training, and increasing model complexity. With a more advanced model some future applications could be used in fraud detection in financial transactions, medical anomalies when analyzing patient records or even imaging, and lastly anomaly detection in NLP for things like spam detection. This experience overall has deepened our understanding of AI principles in the context of cybersecurity.

VII. Team Contributions

The team's contributions were diverse and instrumental in advancing our project. Zahira took the lead by spearheading the creation of the Colab notebook, optimizing all models, and establishing the foundational project structure. Mel significantly enriched the project by incorporating valuable context through detailed comments and explanations within the code, enhancing accessibility for all team members. Jose played a pivotal role in ensuring the project's functionality by executing the code, and ensuring accurate figure displays and results, a critical step in our project's progress. Kent conducted comprehensive research, delving into the literature on general anomaly detection and contemporary techniques, and also undertook the task of organizing, refining, and ensuring the accuracy of the final submission. Connor's contributions were concentrated on researching autoencoders, exploring their benefits, limitations, and potential future applications. These individual efforts, when combined, culminated in a cohesive and well-rounded team endeavor that propelled our project forward.

References

[1] Baptiste Pesquet, "Autoencoders — Machine Learning Handbook," https://www.bpesquet.fr/mlhandbook/algorithms/autoencoders.html