CNT 5410: Computer and Network Security
Final Project Report: Hybrid VAE for Network Anomaly Detection

Dheeraj NVS
vnaganaboina@ufl.edu

February 28, 2024

1 Introduction

In the realm of network security, the identification and mitigation of anomalies within computer networks have emerged as critical concerns due to the ever-evolving cyber threat landscape. Anomalies, denoting deviations from typical behavioral patterns observed within a network, pose significant risks, making anomaly detection pivotal for safeguarding network integrity and data confidentiality.The goal of this project is to introduce an innovative approach, termed the 'Hybrid Variational Autoencoder (VAE) for Network Anomaly Detection,' aiming to enhance anomaly detection capabilities within computer networks. Motivated by the pressing need to proactively detect and address anomalies, our project delves into the challenges posed by increasingly sophisticated cyber threats.The core problem lies in the diversity and complexity of these threats, encompassing intrusions, system failures, and malicious actions, which demand a meticulous examination of network data. Traditional methods often struggle to effectively identify subtle deviations in various network facets such as traffic patterns, device behavior, user interactions, and system performance.

To address this challenge, our solution harnesses the power of a Hybrid Variational Autoencoder (VAE) integrating LSTM (Long Short-Term Memory) and CNN (Convolutional Neural Network) layers. This innovative approach is designed to capture intricate spatial and temporal correlations inherent in network traffic data, thereby significantly improving the detection and mitigation of network anomalies.The interest and significance of this problem stem from the dire need to proactively safeguard networks against sophisticated cyber threats. Anomaly detection acts as a pivotal strategy, enabling organizations to preemptively identify concealed risks and vulnerabilities within their networks. Early detection and timely response to anomalies are imperative to fortify network security in the face of evolving cyber threats.

2 Background & Related Work

In terms of background for this project it is crucial to have a solid understanding of several background concepts that span both machine learning and network security domains. Some core concepts include:

- Variational Auto Encoder
- Convolutional Neural Network
- Long Short Term Memory
- Anomaly in Network Traffic
- Understanding of the features in Dataset
- Cybersecurity concepts such as types of attacks
- Machine Learning Evaluation Metrics
- Data Preprocessing

Having a brief idea on these topics would help in the implementation of the Hybrid VAE in terms of Anomaly detection in Network Traffic for the chosen dataset. To address unknown anomaly attacks and changing network conditions, Machine Learning techniques are employed. This method collects data from a network under normal circumstances and computes training data from that data. The data is then given a label. This data set is used to determine whether or not the network is subject to anomalous attack if any undesirable circumstance develops in the network. Some interesting works in this domain are mentioned below:

One state of the art model paper to be noted is on Isolation forest with a sliding window is one machine learning technique for identifying abnormalities in time series Liu et al. [1] introduced isolation forest, often known as iForest. It creates an ensemble of binary trees that isolate data points called isolation trees, or iTrees. As anomalies have a higher likelihood of isolated than non-anomalous spots, and they are probably nearer an iTree's root [2]. illustrates the process of creating an iTree on a sample data structure. Shorter path length points are therefore seen by this technique as strong candidates for abnormalities. DBSCAN, or Density-Based Spatial Clustering of Application with Noise: is a further clustering-based anomaly detection technique [3]. Compared to previous clustering techniques like CBLOF or STSC [4], it further examines the data's density. The data points are categorized by the approach into three groups:1. Core points 2. Border points 3. Anomalies The user must enter two parameters in order to classify the points: and µ, where is the minimum number of points required for each normal cluster and µ is the distance to declare the neighbors of the examined point. Finding each point's neighbors is a prerequisite to classifying a point. DBSCAN was employed by Çelik et al. [5] to find anomalies in a univariate time-series dataset that included daily average temperature records over 33 years. SVM: A Machine Learning-based network anomaly detection method [6], where the authors made use of SVM, simulated annealing, and decision trees. LSTM: As LSTM networks can learn temporal relations and capture them in a low-dimensional state representation, they are intended to identify contextual anomalies. These relationships may include both short- and long-term dependencies, as well as stationary and non-stationary dynamics. According to [7], LSTM networks are especially well-suited for simulating time-variant systems and multivariate time series. Therefore, one can use the difference between the actual system outputs and the expected outputs that the network predicts to find anomalies. Excellent anomaly detection capabilities have been demonstrated by LSTM-based techniques, as demonstrated by the foundational research of [8].

Although some pertinent work has been done in this domain the shortcomings are not being able to focus and capture the spacial and temporal features of the dataset. And another issue is with the limited availability of labeled datasets, developing privacy-preserving anomaly detection methods that balance the need for security with the protection of user privacy. Previous research in network anomaly detection predominantly employs single architecture-based models or traditional statistical methods. Our approach combines the strengths of both LSTM (Long Short-Term Memory) and CNN (Convolutional Neural Network) layers to capture intricate spatial and temporal relationships within network traffic data. This method is novel because we leverage the two deep learning techniques LSTM and CNN within a Hybrid VAE architecture for effectively capturing complex network anomalies. This proposal introduces a novel approach to enhance network security through the integration of these technologies, ensuring a robust defense against emerging threats.

3 Approach: Dataset(s) & Technique(s)

In the field of cybersecurity, the task of anomaly detection in network traffic demands sophisticated solutions that can decipher intricate patterns and relationships. The approach adopted for this problem includes data preprocessing, desigining and implementing a hybrid Variational Autoencoder (VAE) architecture, targeted model training, reconstruction error calculation, threshold determination, and a comprehensive evaluation using diverse metrics.

- Dataset: The dataset we intend to use needs to have spacial and temporal relationship inorder to carryout our hybrid model implementation hence we are using the UNSW-NB15 dataset. The UNSW-NB15 dataset aims to replicate contemporary network conditions by incorporating various subtle attacks commonly encountered in modern networks. This dataset contains labeled network traffic data with various types of attacks, a variety of features, including source and destination IP addresses, port numbers, protocol types, and timestamps. The dataset comprises ten distinct categories of network traffic. We performed feature engineering on the dataset to capture temporal and spatial relationships.

- Data Cleaning and Preprocessing: Split the data into train and test since we are performing the cleaning and Exploratory Data Analysis on the train set. The test set is again split into two parts of which one is used for testing and the other for validation. The data cleaning is done by checking for any null values, followed by filling the null values with mode of that feature from the train dataset, checking for duplicates, fixing values of the columns, cleaning binary columns, addressing data inconsistencies, managing outliers, filling missing data, understanding the class distribution, datatype of the features, analysing the correlation to find highly correlated pairs of features.

- Feature engineering: A new feature is added network bytes which is a summation of sbytes and dbytes. Also, log1p transformation is applied on numerical features to obtain numerical stability. This transformation can be useful for handling skewed data distributions and improving the performance of the model. The original columns are replaced with their log-transformed counterparts in the dataset.

- Categorical feature encoding: One hot encoding is performed on the categorical features in the model pipeline, The original categorical features are dropped from the training and test datasets, and the one-hot encoded features are concatenated. This effectively replaces the original categorical columns with their one-hot encoded representations. To ensure consistency in the feature sets between the training and testing datasets, additional columns are dropped. Also the feature correlation is explored and some of the highly correalted features are dropped.

- Standardization: Performed Min-Max scaling on the training data using MinMaxScaler. The explanation for chosing MinMaxScaler is derived from the fact that since we are using Convolutional layers in our hybrid model. Since Convolutional Neural Networks (CNNs) are sensitive to the scale of input features, normalizing the input data to a specific range, such as 0 to 1, is a common practice to ensure that the model learns effectively.

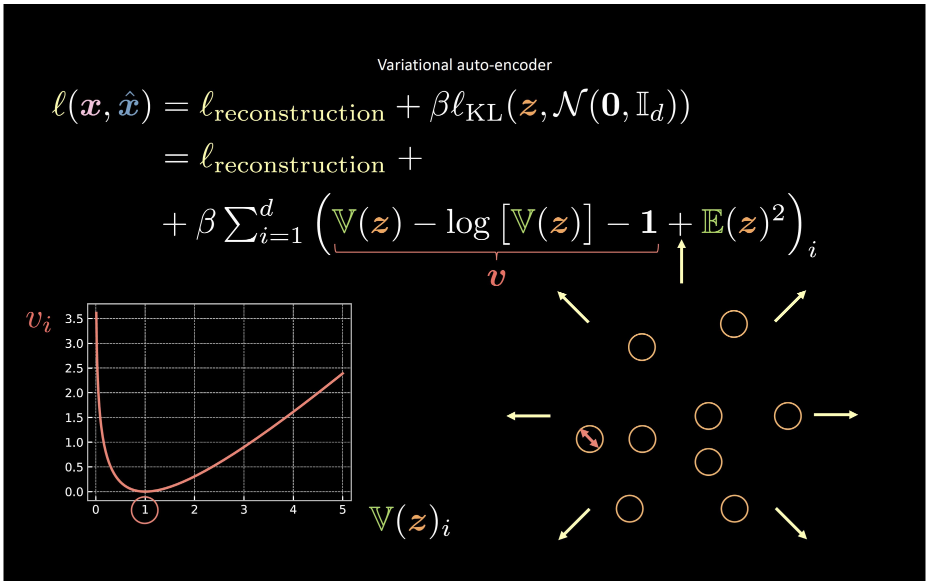
- Building Data pipeline: This CustomDataset class is designed to load and provide access to data stored in a CSV file for PyTorch. The class is built to fit the structure expected by PyTorch's DataLoader, making it suitable for use in training and evaluation pipelines with PyTorch models. Additionally, PyTorch DataLoader objects are initialized for both datasets with the appropriate DataLoader parameters to create two DataLoader objects to iterate over batches of data during the training and validation phases.

- Hybrid VAE architecture: The core part of the proposed solution lies in the design of a hybrid VAE architecture that integrates Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) components within the encoder to capture both spacial and temporal relationships efficiently, promoting a comprehensive comprehension of the intricate relationships found in the network traffic data. The architecture details are mentioned below.

  1. Encoder: The encoder part consists of convolutional layers (conv1 to conv4) followed by a bi-directional LSTM layer with 64 hidden units (lstm) and linear layers (fc1, fc21, fc22) for VAE. The Convolutional layers are followed by ReLU activation function and MaxPooling layers to reduce the dimensions of the input data. We have used a Dropout of 0.5 as a regularization technique.
  2. Reparameterization: The reparameterize method applies the reparameterization trick to sample from the latent space.
  3. Decoder: The decoder includes linear layers (fc3, fc4) and a bi-directional LSTM layer followed by transposed convolutional layers (deconv1 to deconv4).
  4. Forward Method: The forward method orchestrates the complete forward pass through the encoder, reparameterization, and decoder.

- Training: Adam optimizer is initialized for the parameters of the model with a learning rate of 1e-3. The loss function chosen is a combination of Binary Cross Entropy which calculates the reconstruction error and KL Divergence is used to differentiate the generated distribution from the actual distribution as a regularization technique. The batch size is set as 1024 and the model is now trained and validated on normal network traffic until the reconstruction error is minimized.

- Testing: The test set is combined of normal and anomalous points unlike the training set. It contains both features and labels where 0 represents normal network traffic and 1 represents anomalous traffic. A testing loop is created such that each point in the test set is reconstructed and the reconstruction loss is calculate. A threshold for loss is set over which the data points are classified as anomalous. The threshold we set is the 25th percentile of the loss value of the anomalous points.



Variational auto-encoder

$$\ell(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \ell_{\text{reconstruction}} + \beta \ell_{\text{KL}}(\boldsymbol{z}, \mathcal{N}(\mathbf{0}, \mathbb{I}_d))$$
$$= \ell_{\text{reconstruction}} +$$
$$+ \beta \sum_{i=1}^{d} \Big(\underbrace{\mathbb{V}(\boldsymbol{z}) - \log\left[\mathbb{V}(\boldsymbol{z})\right] - \mathbf{1}}_{\boldsymbol{v}} + \mathbb{E}(\boldsymbol{z})^2 \Big)_i$$

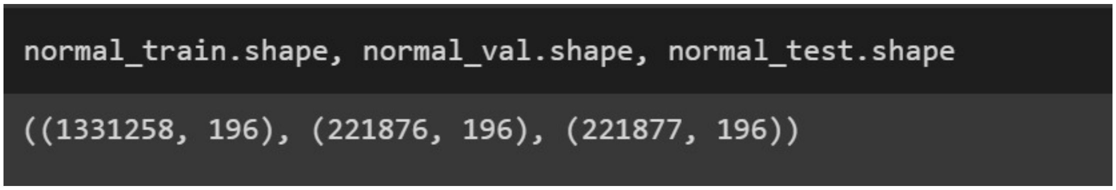
Figure 1: Loss function

- SHAP-SHapley Additive exPlanations: This is a model agnostic approach to understand feature importance after the training is completed. We have tried to implement and apply this to possibly reduce the number of features in the dataset and retrain the model with lesser noise, but due to limitation of computational resources we were unable to execute it succesfully.
- Performance and Evaluation: The robustness of the proposed solution is underscored by a meticulous evaluation strategy. Performance metrics such as precision, recall, F1-score, Area Under the Curve (AUC), and Confusion Matrix collectively paint a comprehensive picture of the anomaly detection system's capabilities. This multifaceted evaluation ensures a nuanced understanding of the model's strengths and limitations, allowing for informed decisions on its deployment.

Novelty of this solution is attributed to its innovative approach to integrating spacial and temporal information. The hybrid VAE architecture, characterized by the the layers of LSTM and CNN components, represents a novel paradigm in anomaly detection. This design choice empowers the model to comprehend complex patterns that arise both spatially and temporally, enhancing its efficacy in discerning anomalous network behavior.Moreover, the decision to leverage the UNSW-NB15 dataset, a real-world repository of diverse network traffic scenarios, adds a layer of authenticity to the solution. The inclusion of labeled data featuring various types of attacks enhances the model's ability to generalize across different threat landscapes. Scalability is evident in the deployment of high-performance GPUs for model training.

4 Results

The following are a compilation of the results obtained while building the hybrid VAE model to detect network anomaly. The model is successfully designed, trained, tested and validated. The consistency of the features is maintained in all the three coherts train, test and validation datasets. The total number of features is 196 excluding label. All the datasets are stratified based on the label so that the balance in labels is maintained to generate the final test dataset.

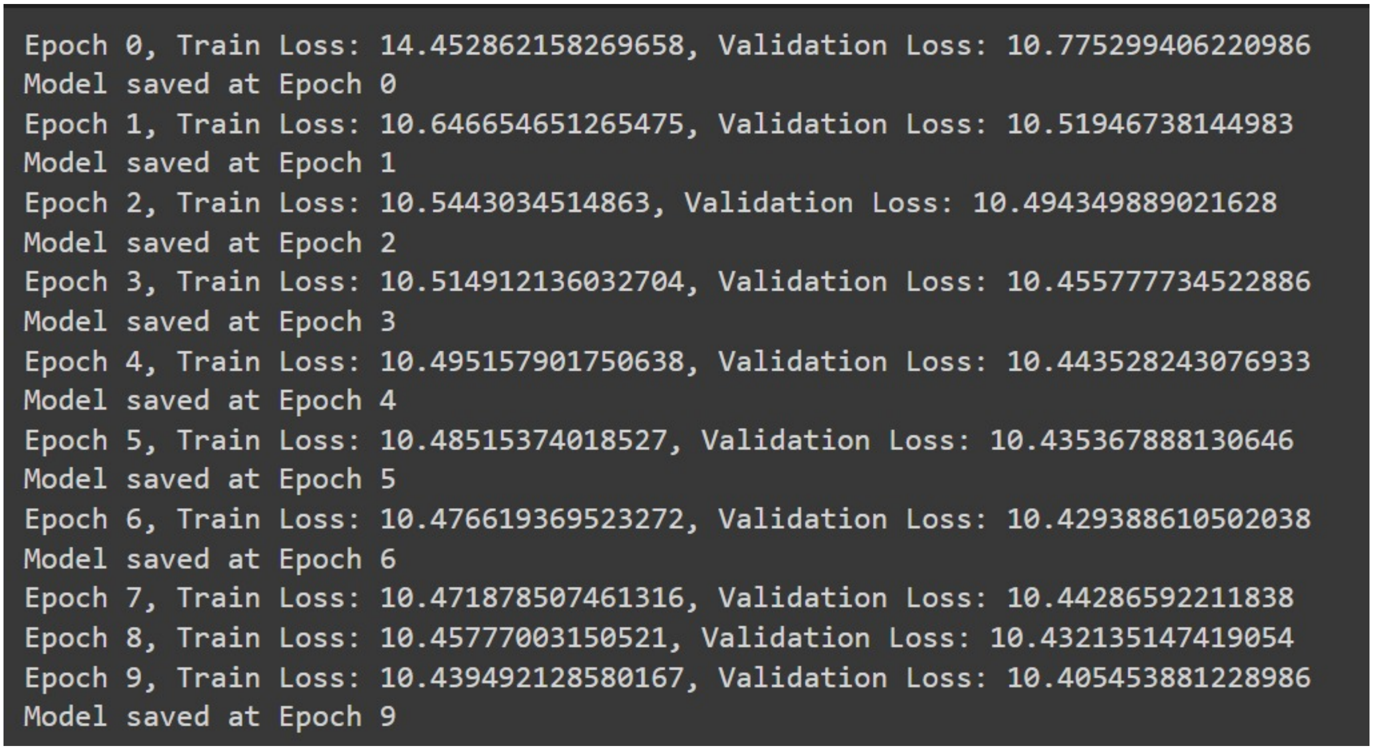


```
normal_train.shape, normal_val.shape, normal_test.shape

((1331258, 196), (221876, 196), (221877, 196))
```

Figure 2: Data shape

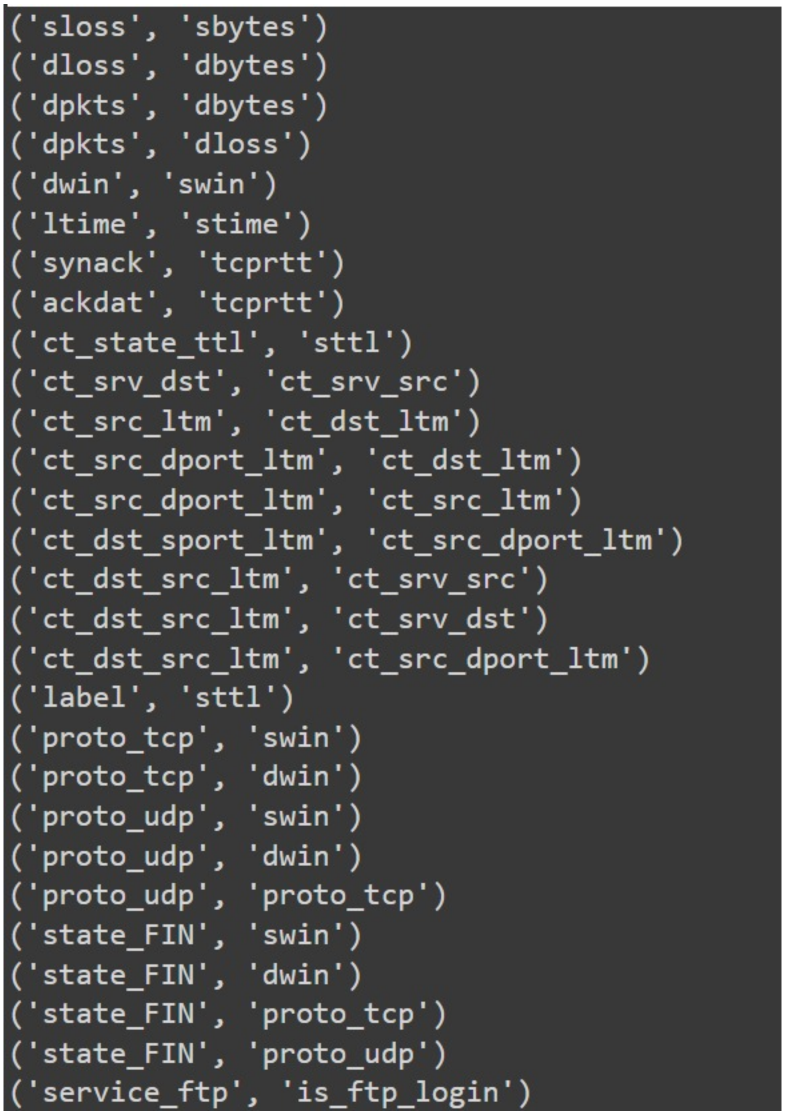
After 10 epochs we can observe that there is no overfitting but there is also no significant improvement in the performance as well. So we can consider the above loss values to be the final reconstruction losses for our implementation



```
Epoch 0, Train Loss: 14.452862158269658, Validation Loss: 10.775299406220986
Model saved at Epoch 0
Epoch 1, Train Loss: 10.646654651265475, Validation Loss: 10.51946738144983
Model saved at Epoch 1
Epoch 2, Train Loss: 10.5443034514863, Validation Loss: 10.494349889021628
Model saved at Epoch 2
Epoch 3, Train Loss: 10.514912136032704, Validation Loss: 10.455777734522886
Model saved at Epoch 3
Epoch 4, Train Loss: 10.495157901750638, Validation Loss: 10.443528243076933
Model saved at Epoch 4
Epoch 5, Train Loss: 10.48515374018527, Validation Loss: 10.435367888130646
Model saved at Epoch 5
Epoch 6, Train Loss: 10.476619369523272, Validation Loss: 10.429388610502038
Model saved at Epoch 6
Epoch 7, Train Loss: 10.471878507461316, Validation Loss: 10.44286592211838
Epoch 8, Train Loss: 10.45777003150521, Validation Loss: 10.432135147419054
Epoch 9, Train Loss: 10.439492128580167, Validation Loss: 10.405453881228986
Model saved at Epoch 9
```

Figure 3: Loss function

The following are the highly correelated features in the dataset and some of the highly correalted features are dropped.



```
('sloss', 'sbytes')
('dloss', 'dbytes')
('dpkts', 'dbytes')
('dpkts', 'dloss')
('dwin', 'swin')
('ltime', 'stime')
('synack', 'tcprtt')
('ackdat', 'tcprtt')
('ct_state_ttl', 'sttl')
('ct_srv_dst', 'ct_srv_src')
('ct_src_ltm', 'ct_dst_ltm')
('ct_src_dport_ltm', 'ct_dst_ltm')
('ct_src_dport_ltm', 'ct_src_ltm')
('ct_dst_sport_ltm', 'ct_src_dport_ltm')
('ct_dst_src_ltm', 'ct_srv_src')
('ct_dst_src_ltm', 'ct_srv_dst')
('ct_dst_src_ltm', 'ct_src_dport_ltm')
('label', 'sttl')
('proto_tcp', 'swin')
('proto_tcp', 'dwin')
('proto_udp', 'swin')
('proto_udp', 'dwin')
('proto_udp', 'proto_tcp')
('state_FIN', 'swin')
('state_FIN', 'dwin')
('state_FIN', 'proto_tcp')
('state_FIN', 'proto_udp')
('service_ftp', 'is_ftp_login')
```

Figure 4: highly correlated features

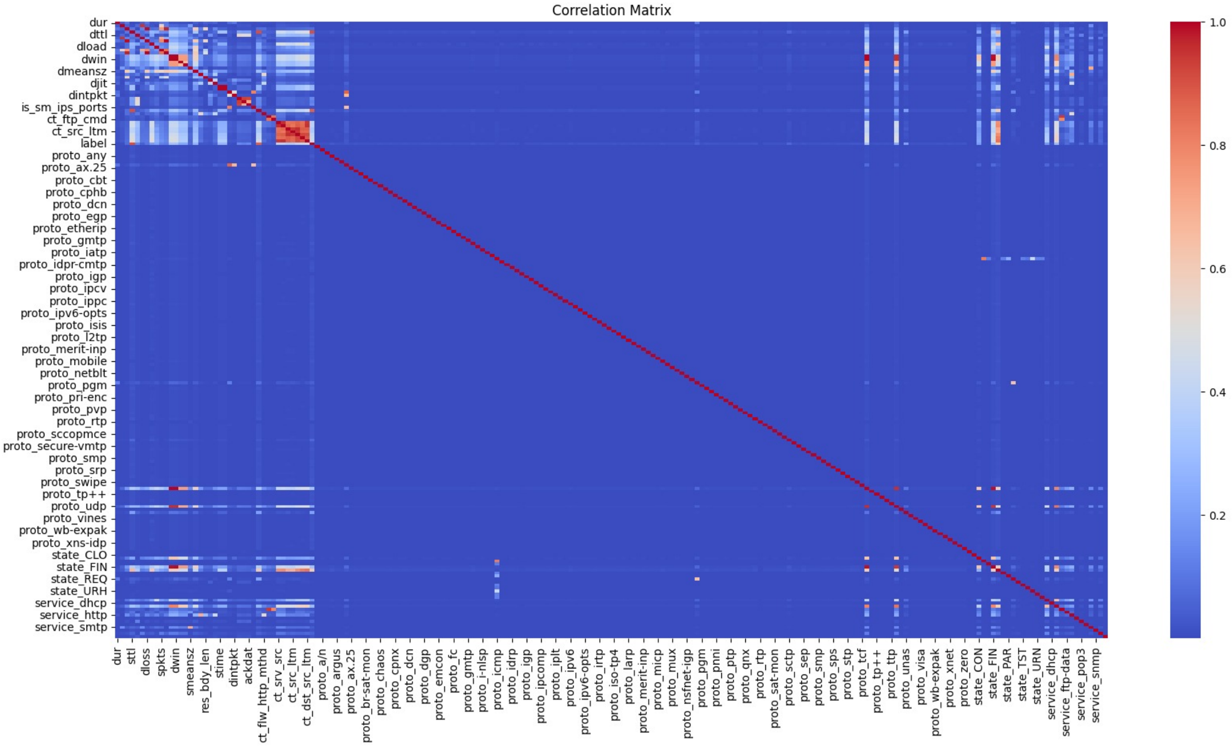
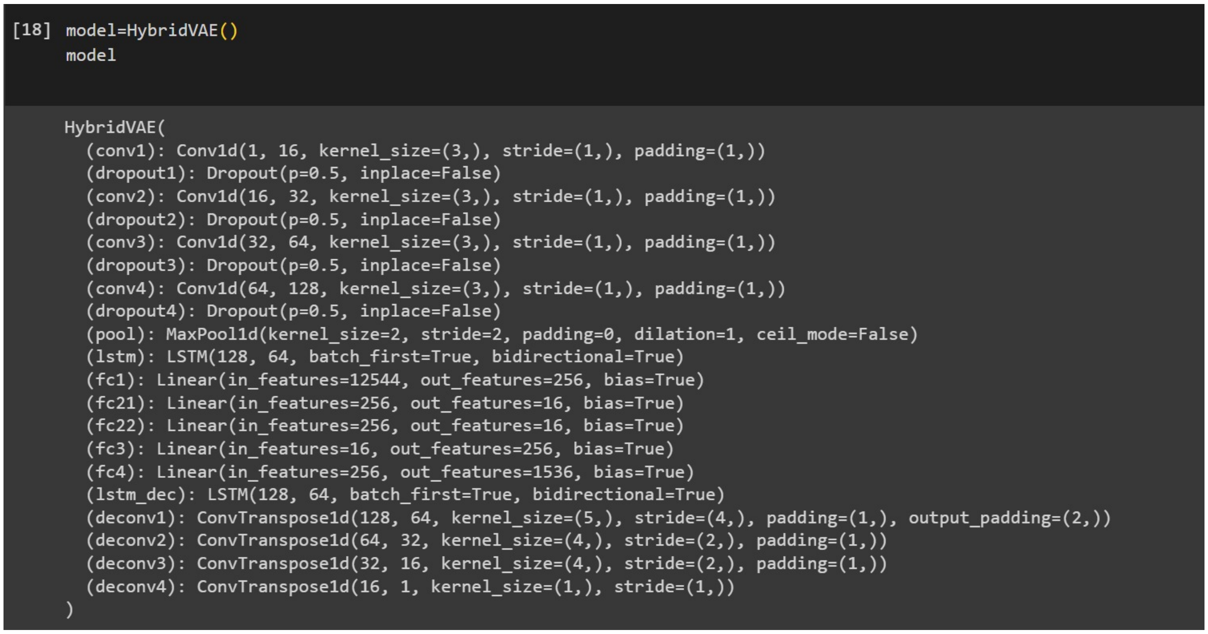


Figure 5: Correlation matrix

Model Architecture: The Hybrid VAE architecture is built with CNN and LSTM layers in the encoder to capture the spatial and temporal features. LSTM and De-convolutional layers are used as part of the decoder.

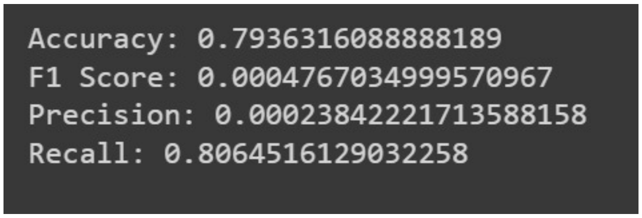


```
[18] model=HybridVAE()
     model
```

```
     HybridVAE(
       (conv1): Conv1d(1, 16, kernel_size=(3,), stride=(1,), padding=(1,))
       (dropout1): Dropout(p=0.5, inplace=False)
       (conv2): Conv1d(16, 32, kernel_size=(3,), stride=(1,), padding=(1,))
       (dropout2): Dropout(p=0.5, inplace=False)
       (conv3): Conv1d(32, 64, kernel_size=(3,), stride=(1,), padding=(1,))
       (dropout3): Dropout(p=0.5, inplace=False)
       (conv4): Conv1d(64, 128, kernel_size=(3,), stride=(1,), padding=(1,))
       (dropout4): Dropout(p=0.5, inplace=False)
       (pool): MaxPool1d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
       (lstm): LSTM(128, 64, batch_first=True, bidirectional=True)
       (fc1): Linear(in_features=12544, out_features=256, bias=True)
       (fc21): Linear(in_features=256, out_features=16, bias=True)
       (fc22): Linear(in_features=256, out_features=16, bias=True)
       (fc3): Linear(in_features=16, out_features=256, bias=True)
       (fc4): Linear(in_features=256, out_features=1536, bias=True)
       (lstm_dec): LSTM(128, 64, batch_first=True, bidirectional=True)
       (deconv1): ConvTranspose1d(128, 64, kernel_size=(5,), stride=(4,), padding=(1,), output_padding=(2,))
       (deconv2): ConvTranspose1d(64, 32, kernel_size=(4,), stride=(2,), padding=(1,))
       (deconv3): ConvTranspose1d(32, 16, kernel_size=(4,), stride=(2,), padding=(1,))
       (deconv4): ConvTranspose1d(16, 1, kernel_size=(1,), stride=(1,))
     )
```

Figure 6: Hybrid VAE architecture

Performance and Evaluation obtained are Precision, Accuracy, Recall, F1-score, ROC and Confusion Matrix.



```
Accuracy: 0.7936316088888189
F1 Score: 0.0004767034999570967
Precision: 0.00023842221713588158
Recall: 0.8064516129032258
```

Figure 7: Performance Metrics

The following is the Confusion matrix obtained is below. True Positives: 403273, False Positives: 104705, False Negatives: 6 and True Negatives: 25.

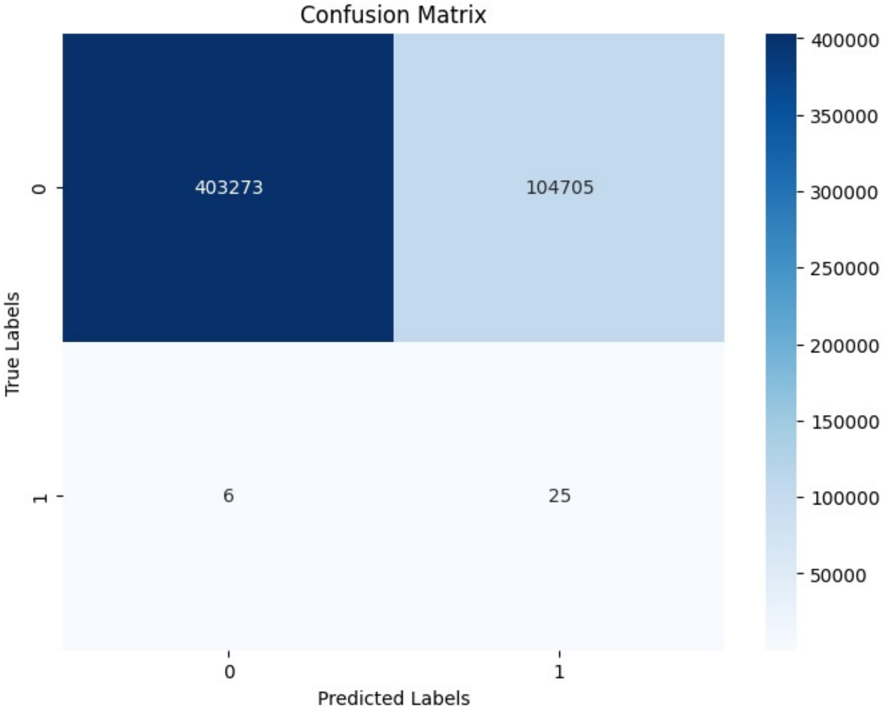
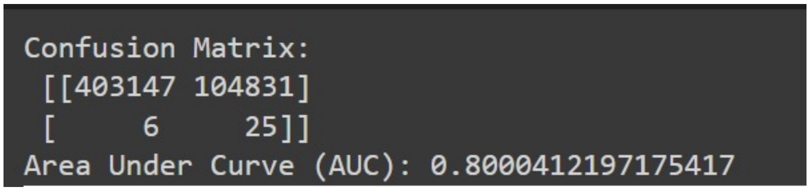


Figure 8: confusion matrix



```
Confusion Matrix:
 [[403147 104831]
 [     6     25]]
Area Under Curve (AUC): 0.8000412197175417
```

Figure 9: confusion matrix

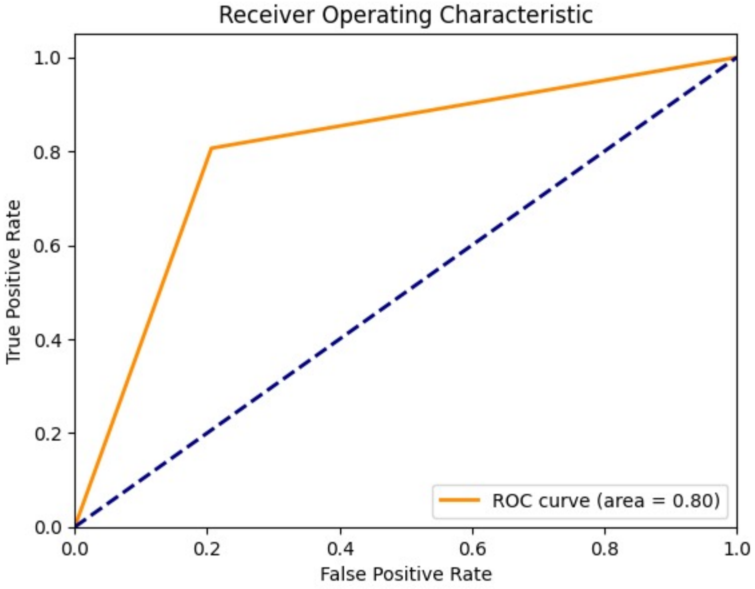


Figure 10: ROC curve

The ROC curve above demonstrates a model with a good ability to distinguish between the positive and negative classes, as indicated by the curve being well above the line of no-discrimination. The area under the ROC curve (AUC) is 0.80, suggesting that the model has a 80% chance of correctly discriminating between a randomly chosen positive and negative instance. Overall, the model shows a strong predictive performance, but there's room for improvement since an AUC of 1.0 would represent a perfect model.

5 Conclusions

The hybrid model is able to predict anomalous network traffic with 80% accuracy, but we have observed that the False positive rate of the model is also very high which leads to very low precision. One thing we were able to understand is that the network traffic data has both spacial and temporal features. Representing the data in a higher dimension and using layers like Conv2D or Conv3D, like channels of an image or depth of a volumetric object could potentially decrease the reconstruction loss. The depth of the model and the batch size can be further increased to improve the latent space representation. The accuracy of the model can further be increased by careful feature selection by manual methods or by running a feature importance algorithm before training the final network.

Limitation:

1. Calculating the input and output dimensions of each layer in the hybrid VAE mode manually can be time consuming and error bound. Also every time we want to change the data representation we would need to change the model architecture accordingly.

2. Although applying the deep explainer of the SHAP-API on sophisticated architectures like VAE is less interpretable we tried to implement it but were unable to execute it due to the lack of enough computational resources.

3. The current architecture is generting many False Positives which is evident in the Precision value, this means the model is still not able to completely differentiate between normal and anamalous points

Future work:

1. Employing better feature engineering techniques and representing the data in a 3-dimensional format where the depth dimension corresponds to the temporal features of the data.

2. Using more sophisticated architecture with 3D Convolutional layers to enhance the capturing of spatial relationships.

3. Implementing SHAP to understand the feature importance and using the SHAPEly values to filter uneccessary features that could add noise to the data representation.

4. Extending this architecture to other datasets like NSLKDD which have inherent temporal dimensions.

References

[1] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*, pp. 413–422, IEEE, 2008.

[2] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, pp. 1–39, 2012.

[3] M. Ester, "Density-based clustering.," *Data clustering: algorithms and applications*, vol. 31, p. 111, 2013.

[4] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern recognition letters*, vol. 24, no. 9-10, pp. 1641–1650, 2003.

[5] M. Çelik, F. Dadaşer-Çelik, and A. Ş. Dokuz, "Anomaly detection in temperature data using dbscan algorithm," in *2011 international symposium on innovations in intelligent systems and applications*, pp. 91–95, IEEE, 2011.

[6] S.-W. Lin, K.-C. Ying, C.-Y. Lee, and Z.-J. Lee, "An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection," *Applied Soft Computing*, vol. 12, no. 10, pp. 3285–3290, 2012.

[7] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich, "A survey on long short-term memory networks for time series prediction," *Procedia CIRP*, vol. 99, pp. 650–655, 2021.

[8] P. Malhotra, L. Vig, G. Shroff, P. Agarwal, *et al.*, "Long short term memory networks for anomaly detection in time series.," in *Esann*, vol. 2015, p. 89, 2015.