NETWORK INTRUSION DETECTION USING VARIOUS AUTOENCODER METHODOLOGIES

VIT University, Chennai
Dr. Pattabiraman
Mukkesh Ganesh, Akshay Kumar
pattabiraman.v@vit.ac.in
g.mukkesh2017@vitstudent..ac.in
akshaykumar.satheesh2017@vitstudent.ac.in

Abstract— Network security is one of the most critical fields of computer science. With the advent of IoT technologies and peer-to-peer networks, the significance of mitigating security threats has never been higher. Network Intrusion Detection Systems are used to monitor the traffic in a network to detect any malicious or anomalous behavior. Anomalous behaviour includes different types of attacks such as Denial of Service (DoS), Probe, Remote-to-Local and User-to-Root. If an attack/anomaly is detected, custom alerts can be sent to the desired personals. In this paper, we will be exploring the effectiveness of various types of Autoencoders in detecting network intrusions. Artificial Neural Networks can parse through vast amounts of data to detect various types of anomalies and classify them accordingly. An autoencoder is a type of artificial neural network which can learn both linear and non-linear representations of the data, and use the learned representations to reconstruct the original data. These hidden representations are different from the ones attained by Principal Component Analysis due to the presence of non-linear activation functions in the network. Reconstruction error (the measure of difference between the original input and the reconstructed input) is generally used to detect anomalies if the autoencoder is trained on normal network data. Here, we used 4 different autoencoders on the NLS-KDD dataset to detect attacks in the network. With just reconstruction error, we were able to achieve a highest accuracy of 89.34% by using a Sparse Deep Denoising Autoencoder.

***Keywords*— Network Intrusion Detection; Artificial Neural Network; Autoencoders; Anomaly Detection;**

I. Introduction

An anomaly or an outlier is the point in the data that are significantly different from the rest of the dataset. In most cases, these anomalies are very few in number and hence have a large class imbalance. Detection of anomalies have a lot of application in a variety of fields, from cyber security to credit card fraud detection.

In the field of network security, it is crucial to identify rogue packets. These packets could lead to network attacks, which could potentially decrease the availability of systems. Millions of dollars can be lost due to website downtimes. Hence, the fast detection of these packets is critical in nature.

Autoencoders are a type of neural network which can learn the encodings of various inputs and learn to reconstruct them from a smaller/larger dimension with minimal error. These Autoencoders can be trained to represent only the normal activities. Hence, the error between the reconstructed inputs and the actual inputs will be high in the case of anomalous data points. This can be used to detect anomalies in the network.

In this paper, we have taken different types of autoencoders and trained them on the NSL-KDD-CUP network intrusion dataset. The models were trained only on the normal packets and the reconstruction error was used to classify the unseen packets as attack/normal packets. The encoding parts can also be used as a dimensionality reducer and be stacked together with other ML/DL algorithms to get more discriminative models.

II. Literature survey

[1] Have worked on the Network Intrusion Detection system by using the deep learning paradigm known as Self-taught Learning (STL). They created a two-stage sparse autoencoder and trained it on unlabeled data and then used the learned features and passed through a classifier to identify network intrusions.

[2] The paper talks about using the Fuzzy Rough Clustering algorithm to detect network anomalies and gave results comparative to the K-means result making the Fuzzy Rough Clustering algorithm more efficient.

[3] The paper gives a comparative study on the common machine learning algorithms such as random forest, SVM, Navie Bayes etc. that are trained on 41 features. All the features weren't used in this paper.

[4] This paper talks about using variational autoencoder (VAE) and uses the reconstruction probability as the anomaly score to determine the error of the autoencoder. Contrary to most of the autoencoder methods, Variational AutoEncoders are actually generative models.

[5] In this paper, the authors have combined the variational autoencoder (VAE) with the data that is split task wise while training the model. This model is an effective system to mitigate catastrophic forgetting.

III. Dataset

For this analysis, we chose to work with the NLS-KDD dataset [6]. The training data has 1,25,973 data points whereas the test data has 22,543 datapoints. It is the benchmark dataset for intrusion detection based on machine learning approaches. We grouped the 38 different attack types into 4 different attacks. Namely,

- Denial of Service Attack (DoS)
- User to Root Attack (U2R)
- Remote to Local Attack (R2L)
- Probing Attack

The remaining packets are classified as normal.

However, the distribution of packets in the training and testing data are not the same to simulate a real-world scenario. It is illustrated by the figures 1 and 2.

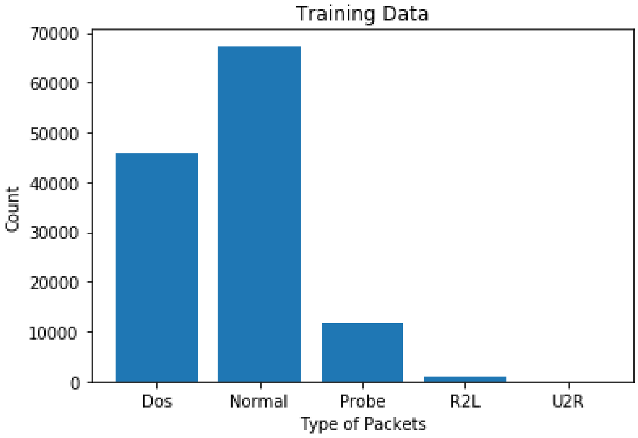

Figure 1*: Training Data Distribution.*

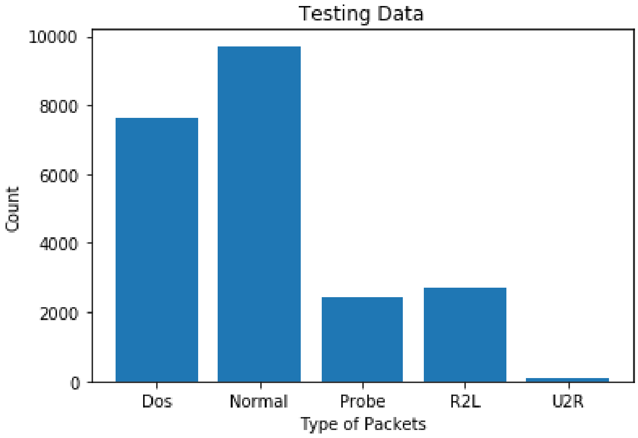

Figure 2*: Testing Data Distribution.*

The U2R samples are virtually inexistent in the training set (0.04%) and hence, leads to vary poor classification accuracy on trained classifiers. Similarly, the percentage of R2L packets differ significantly in these two datasets. However, by using the Autoencoder method, we can overcome this hurdle, since we are looking at only the normal packets.

Label encoding, dummy variables and minmax scaler were used on appropriate columns. The final number of columns in the dataset was 122.

10% of the training dataset was held for validating the models, and the trained models were tested on the test set. For training the autoencoders, only the normal packets from the training data was used.

IV. Prediction Models

A. Undercomplete Deep Autoencoders

An autoencoder is a neural network that can efficiently learn encodings of data using supervised learning. An undercomplete autoencoder has lesser neurons than the input layer in its hidden layer(s). If more than one hidden layer is present, then it is a deep autoencoder. An undercomplete autoencoder's structure looks like Fig 3.

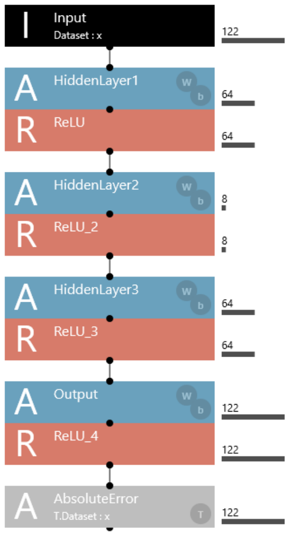

Figure 3*: Autoencoder Architecture.*

Reconstructed inputs are present in the output layer. Loss is calculated by finding the mean of the square of the difference between the reconstructed values and the original values. The loss is then backpropagated through the network and the weights are adjusted accordingly.

By training the autoencoder on just the normal data, we can use the reconstruction error (with set thresholds) to detect the abnormal transactions.

B. Denoising Autoencoder

Its architecture is very similar to that of a vanilla autoencoder. The main difference is that the inputs are corrupted to ensure more generalization. The type of corruption can vary (e.g., Dropout Layer, adding noise, etc.) In our architecture, Dropout was added after the Input Layer to randomly corrupt/drop the input neurons for every epoch.

Figure 4*: Denoising Autoencoder Architecture.*

Here too, the loss is calculated with respect to the Input layer and not the corrupted inputs. This has been shown to increase the model's generalization. Once again, we use the reconstruction error to detect abnormal transactions.

C. Sparse Autoencoders

A sparse autoencoder usually contains more neurons in the input layer when compared to the input layer. To prevent the network from simply copying the inputs into its hidden layer, a sparsity constraint in the form of activity regularizer is added. This is usually L2 regularization.

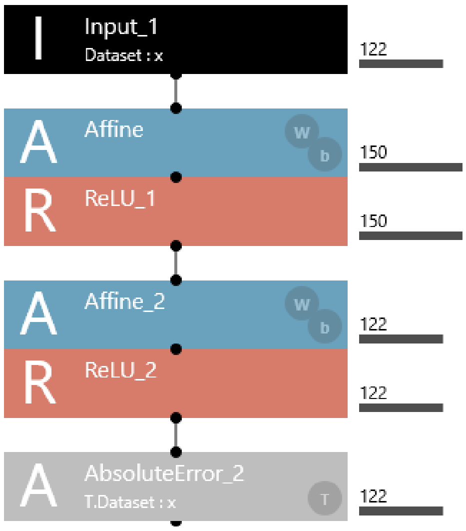

***Figure 5**: Sparse Autoencoder Architecture.*

Training happens in the same manner.

D. Sparse Denoising Deep Autoencoder

This is a combination of both Denoising Autoencoder, Sparse Autoencoder and a deep Autoencoder. A Dropout layer is present after the input layer which corrupts the input, has more neurons in the hidden layer of the encoding part with L2 Activity Regularization.

***Figure 6**: Sparse Denoising Deep Autoencoder Architecture.*

The first hidden layer has 144 neurons and the 2^{nd} has 150 neurons. L2 activity regularization of 10e-4 is present in both these layers. Training is done in a similar fashion.

V. Training Results

All the models were trained on the Google Colab platform with the help of its free TPU instances. Tensorflow and Keras were used. The models were compiled with Mean Squared Error as the Loss function, and Nesterov Implemented Adam was used as the optimizer. Dropout layer had P=0.3. The L2 activity regularizers had a value of 10e-4. All the models were trained for 25 epochs, during which they converged.

The models converged in 30 epochs. The best model was chosen for evaluation.

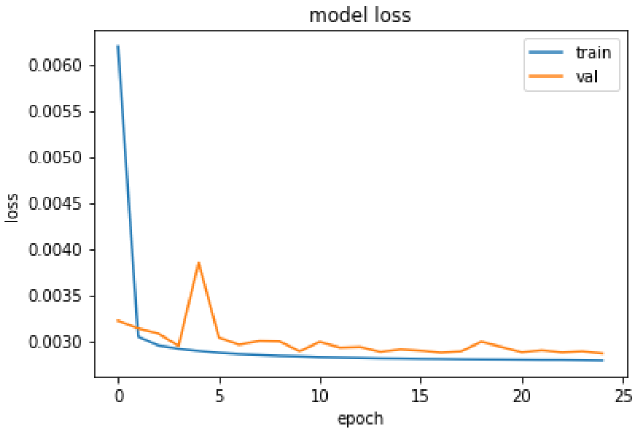

***Figure 7**: Loss per epoch graph for Deep Autoencoder.*

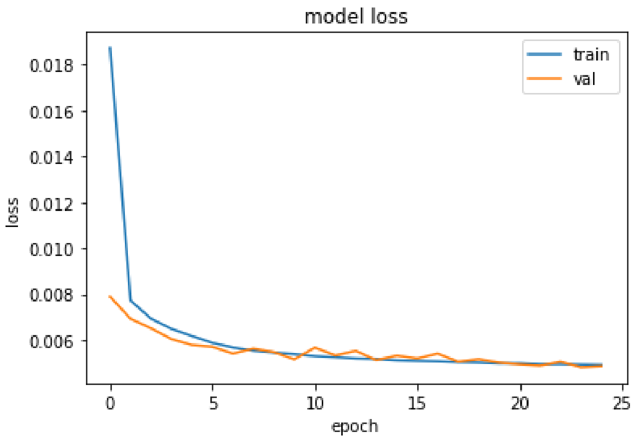

Figure 8*: Loss per epoch graph for Denoising Autoencoder.*

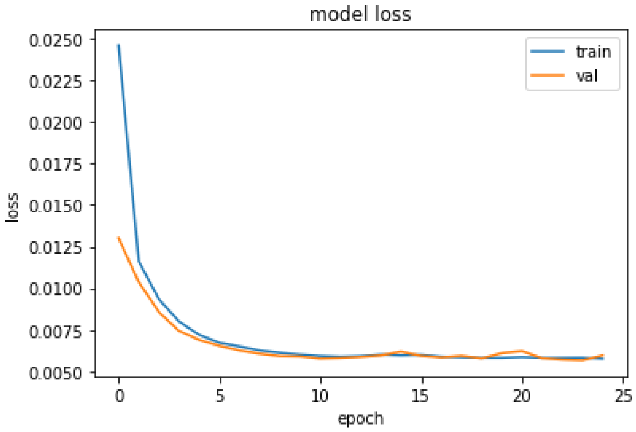

Figure 9*: Loss per epoch graph for Sparse Autoencoder.*

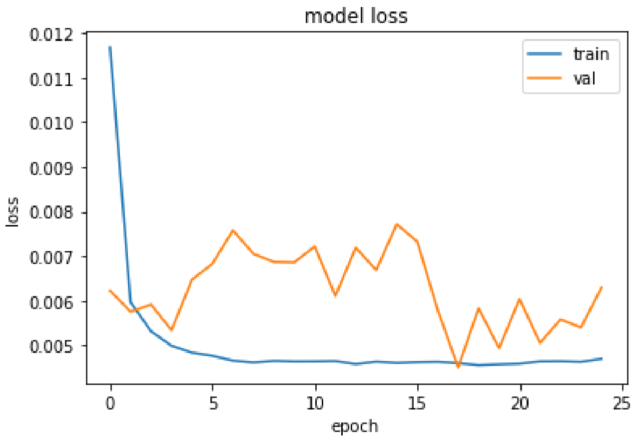

Figure 10*: Loss per epoch graph for Deep Sparse Autoencoder.*

Table 1: Threshold Values of Autoencoders

Model	Threshold
Deep Autoencoder	0.0028
Denoising Autoencoder	0.0049
Sparse Autoencoder	0.0058
Denoising Sparse Autoencoder	0.0045

These were the last epoch's loss values. They were used to set the threshold.
The trained models were then evaluated on the testing dataset.

VI. Analysis Of Results

The thresholds were set as the loss value of the 25th epoch of the respective models. If the reconstruction error of a particular data point is lesser than the threshold, then we classify it as a regular data point. If not, we proceed to classify it as an abnormal datapoint. The results along with their respective violin plots of reconstruction error are plotted below.

Table 2: Confusion Matrix of Deep Autoencoder

Denoise Model	Normal Predicted	Attack Predicted
Normal Actual	6818	2892
Attack Actual	457	12376

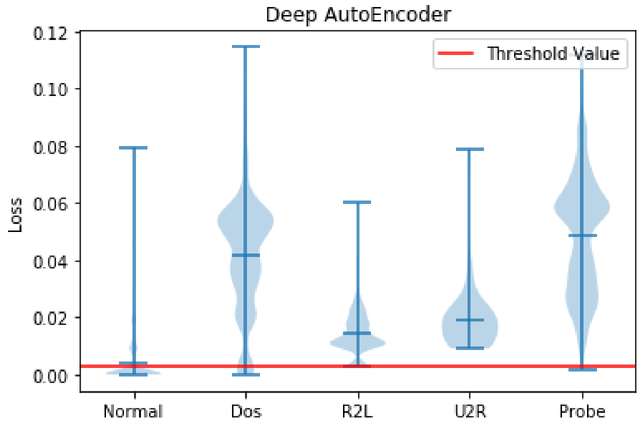

Figure 11*: Distribution of Reconstruction error for Deep Autoencoder.*

Table 3: Confusion Matrix of Denoising Autoencoder

Deep Model	Normal Predicted	Attack Predicted
Normal Actual	6857	2853
Attack Actual	547	12286

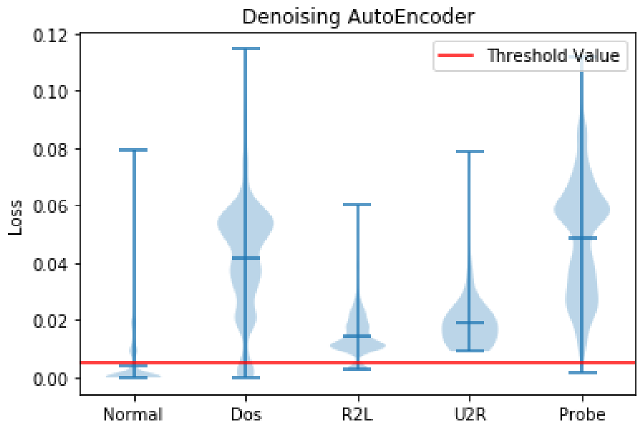

Figure 12*: Distribution of Reconstruction error for Denoising Autoencoder.*

Table 4: Confusion Matrix of Sparse Autoencoder

Sparse Model	Normal Predicted	Attack Predicted
Normal Actual	7025	2685
Attack Actual	636	12197

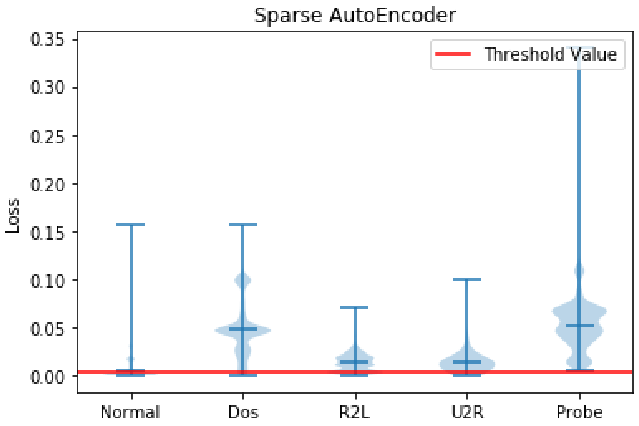

***Figure 13**: Distribution of Reconstruction error for Sparse Autoencoder.*

Table 5: Confusion Matrix of Sparse Denoising Deep Autoencoder

DSD Model	Normal Predicted	Attack Predicted
Normal Actual	79942	1768
Attack Actual	633	12200

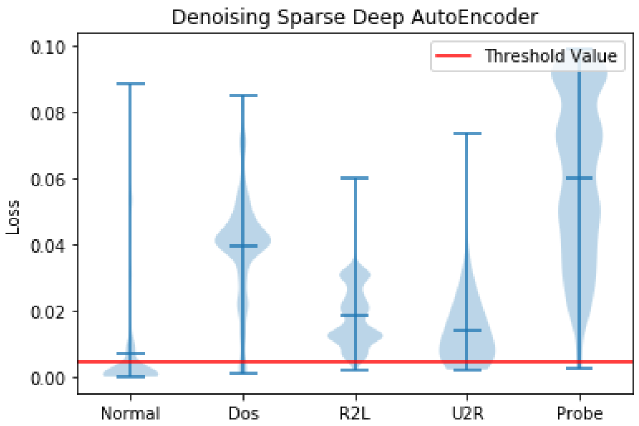

***Figure 14**: Distribution of Reconstruction error for Denoising Sparse Deep Autoencoder.*

Table 6: Accuracy comparison of Models

Model	Total	Normal	DOS	R2L	U2R	Probe
Deep	85.78	74.35	92.10	87.96	100	99.95
Denoise	84.91	70.62	93.54	98.04	100	99.95
Sparse	85.27	72.35	96.12	87.85	83.58	100
DSD	89.34	81.8	92.58	98.70	94.02	98.84

We can see that the Denoising Sparse Deep Autoencoder outperforms the other types of Autoencoders. This is just from the reconstruction error. The (virtual) absence of U2R packets in the training data did not hamper the model's performance in classifying them as an attack packet. This is one of the main advantages of Autoencoder based anomaly detection.

VII. Conclusion and Future Enhancement

The NSL-KDD benchmark dataset was used to train 4 different types of autoencoders. The autoencoders were trained on the normal packets to distinguish them from the attack packets. The combination of 3 models gave us the best results. The Denoising Sparse Deep Autoencoder was able to achieve an accuracy of 89.34% on the unseen test data. Class imbalance due to the presence of extremely low records of U2R attacks in the training set was mitigated by the use of autoencoders. 2 of the models were able to achieve 100% accuracy in classifying U2R type packets as anomalies.
For future research, these models can be saved and the encoding parts can be used as dimensionality reducers. The output of these encoding parts can be used as inputs for other discriminatory ML and DL models such as SVM, Logistic Regression, MLP etc.

VIII. References

[1] Javaid, Ahmad, et al. "A deep learning approach for network intrusion detection system." Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2016.

[2] Chimphlee, Witcha, et al. "Anomaly-based intrusion detection using fuzzy rough clustering." 2006 International Conference on Hybrid Information Technology. Vol. 1. IEEE, 2006.

[3] Revathi, S., and A. Malathi. "A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection." International Journal of Engineering Research & Technology (IJERT) 2.12 (2013): 1848-1853.

[4] An, Jinwon, and Sungzoon Cho. "Variational autoencoder based anomaly detection using reconstruction probability." Special Lecture on IE 2.1 (2015).

[5] Wiewel, Felix, and Bin Yang. "Continual Learning for Anomaly Detection with Variational Autoencoder." ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019

[6] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009, July). A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (pp. 1-6). IEEE.