Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Deep Neural Network using Autoencoder on NSS-KDD Data Set

Dr. R. Raja Kumar[1], G. Naveen Kumar[2], M. Jyoshna[2], K. Rahul[2], B. Munendra[2]

[1]Professor, Department of Computer Science and Engineering, Rajeev Gandhi Memorial College of Engineering & Technology, Nandyal-518501, Andhra Pradesh, India
[2]UG Students, Department of Computer Science and Engineering, Rajeev Gandhi Memorial College of Engineering & Technology, Nandyal-518501, Andhra Pradesh, India
Email: rajakumar.rajaboina@gmail.com

It usually uses the advanced and complex machine learning techniques in intrusion detection systems (IDSs) to improve detection against the ever more complex threats. One area of research investigated the use of deep learning, especially autoencoders, to improve the detection of anomalies in network traffic data collected using the NSL-KDD dataset. The NSL-KDD dataset is a responsible update from KDD Cup 99 and provides rather large network traffic record extracts of malicious and benign activities, hence making it a bona fide dataset for the evaluation of an IDS's performance.

An unsupervised anomaly detection application is performed on deep autoencoder architecture using NSL-KDD dataset. An autoencoder is a neural high-to-low feature extraction data reconstruction, which trains the network traffic to a lower dimensional latent and reconstructs them back to the original format. Anomalies are detected as much from the reconstruction error-which indicates high deviations from the actual expected reconstruction and hence flagged for further interpretation as natural intrusion.

The experimental results revealed that deep autoencoder techniques improved performance using different traditional machine learning techniques in detecting different classes of network attacks, evidenced by extensive improvements in accuracy, precision, recall, and F1 properties. It will show that the capabilities of IDS are significantly improved in identifying and responding to advanced network threats by deep autoencoders. The research also confirms the use of deep learning techniques potential in cybersecurity for future business toward a more robust and adaptive intrusion detection system(IDS).

Keywords: Network Security, Deep Learning, Autoencoder, Anomaly Detection, NSL-KDD dataset.

1. Introduction

An important part of the modern digital era is that the millions of networked systems now open up the world for efficient and easily accessible communication, data sharing, and services that connect people and businesses to the wider community. With such matching, however, networks are exposed to a multitude of threats, from simple intrusions to relatively complex attacks. Whether it is through a weak network or a strong one, attacks easily lead to data breaches, service disruptions, and financial losses. This makes it necessary to widely propagate and implant strong safety mechanisms into networks. Among such mechanisms include Intrusion Detection System (IDS), which monitor network traffic for Analysed it, and identify malicious activity.[1]

Traditional approaches to IDS rely heavily on the application of rules and classical machine learning techniques, which can effectively address most known attack patterns but lack flexibility in accommodating novel ones as they evolve. High dimensionality complicates it even more as traffic data may not only present limitations but also inefficiencies in processing and analysis. Hence, most researchers' preference today has been towards the application of a deep learning paradigm, which has proved to be exceptional in pattern recognition and anomaly detection.

This research applies deep autoencoders-a variant of artificial neural network designed to perform feature extraction-and reconstruction-to enhance the performance of IDS. Autoencoding is being conducted using the orthodox and well-accepted benchmark data set for network intrusion detection research calls as the NSL-KDD dataset, through which bugs are expected to be better detected hidden in the network traffic data. provided by NSL-KDD are richly compiled label records of live traffic of various normal and malicious applications, thus providing an excellent resource to test the developed technique's performance Excellence Aid.

The research focused mainly on developing an IDS based on deep architecture autoencoders to detect many kinds of attacks with high accuracy. Unlike most conventional methods, anomalies are identified in this model through the analysis of the reconstruction error, thereby making the method flexible and scalable on evolving threats in cybersecurity. As such, it tries to overcome the limits of existing systems while boosting performance for accuracy, precision, recall, and F1 score.

This research can contribute in two ways. Firstly, we can show that autoencoders can improve an IDS performance using the NSL-KDD dataset. Secondly, we can still say that this research results in a framework where deep learning techniques will be integrated into real-world network security systems. This study is a step toward intelligent real-time intrusion detection solutions, which could indeed imply a quantum leap in the field of cybersecurity.

2. Related Work

With such expansion in varieties of phenomena also penetrating into the network intrusion detection system (IDS), it has passed through various stages of development in the input and reporting technologies against the increasing complexities of cyber threats. The earliest tools were rule-based and classic machine-learning-based for intrusion detection.

An intrusion detection system (IDS) has undergone several phases of evolution to develop more smooth methods for detection and reporting against the ever-increasing complexities of cyber threats. Some of the first tools were rule-based and classical machine learning models for intrusion detection. As attack evolution became less clearly identifiable, traditional methods proved their severe limitations and pushed the research community to study more advanced techniques like deep learning.

Rule-based IDS Snort and Bro identify on a set of predefined signatures or rules for detecting threats. Although effective in known attacks, the two will not work on zero-day or unknown threats as they are entirely dependent on the currently updated rule sets. These two factors make them not fit enough for today's dynamic cybersecurity environment.[2]

The classical machine learning methods, including support vector machines, decision trees, and k-nearest neighbour techniques, introduced some sort of flexibility to learning with the help of data. They can therefore be best used for classifying network traffic and detecting malicious attacks much better than a rule-based approach. However, this household faces the challenges of very high-dimensional data and require feature engineering, and particularly these methods are not effective with high-dimensional input data when the size and complexity of the network traffic in modern systems is increasing.

Deep learning solves this major drawback of IDS significantly. It illustrates the types of architectures concerning deep learning, namely the Convolutional Neural Networks (CNNs), the Recurrent Neural Networks (RNNs), and automatic encoders. In the all architectures mentioned, the one having excellent performance with unsupervised anomaly detection techniques is an autoencoder. This compresses data to a lower dimension and reconstructs it. If the reconstruction differs from the original beyond some threshold, it indicates an anomaly. Thus they perform very well in identifying both known and unknown threats.

The NSL-KDD dataset has served as a crucial resource to evaluate different IDS models. It improves over the KDD Cup 99 dataset by correcting the problems of redundant records and class imbalances, making it stronger for current testing models. The results derived from studies on this dataset showed that deep learning models were better than traditional ones in intrusion problems. For instance, deep autoencoders achieved very high accuracy and recall rates in detecting subtle changes in network traffic through the accuracy of their detected patterns. Other improvements included hybrid models: an autoencoder combined with a supervised classifier.

In the present work, we adopted such approaches and developed a deep autoencoder model system for analyzing the NSL-KDD dataset. This model employs anomaly monitoring through reconstruction error detection-an easily scalable and adaptable technique for modern IDS systems. It tries to compare the performance of the present study with earlier studies to illustrate that its potential to detect many different types of network attacks and to overcome certain weaknesses of legacy systems.

3. Methodology

Dataset Description

Thus, it can be observed that the NSL KDD Dataset is the enhanced version of KDD cup 1999 database more exposed to competitions aimed at evaluating intrusion detection systems. The d-Verse utilizes and recognizes traffic network data for normal or attack classification. It is also made up of training and testing datasets for several types of attacks.

- Features: The whole data set consists of 41 features which are categorical and continuous attributes, describing many Aspects at different levels of network traffic network: Structure of connection type, length of time, protocol type, service type, flag, etc.

➢ Categorical features: Protocol type (e.g. TCP, UDP, ICMP), service (e.g. HTTP, FTP), and flag.

➢ Continuous features: time of connection, number of connections, and data transfer rates.

- Classes: Dataset Among Categories of different attacks, as well as normal traffic, classified in the following way:

➢ Normal: This means no malicious activities happen.

➢ Attack-types: Denial of Service (Dos), Probing, User to Root (U2R), and Remote to Local (R2L), name the types of attack available in the dataset for analysis together with a normal class, which will be implemented for classification.

- Size: The NSL-KDD database has about 125,973 instances to train the models, of which 22,544 are usable for testing the models.

The steps for pre-processing are as follows:

1. Pre-Processing Data

• Missing Value Treatment: The treatment of missing values includes imputing or dropping cases with excessive missing data, depending on the extent and nature of the missing data.

• Also, Outlier Correction: Outliers or extreme values against the normal background of network behaviour are either corrected or removed because they ruin the results.

2. Feature Normalization or Scaling

• Normalization: Most of the continuous features have different ranges as in Time Duration-I Minutes versus No. of bytes downloaded for an account where min-max normalization comes into play, this ensures the scaling of all features in the range of 0 to 1 or - 1 to + 1 which aids in the facilitation of co-creating with endogenous learning.

• Standardization: Alternatively, standardization (scale to zero mean and unit variance) could also be used depending on model performance during experimentation.[8]

3. Handling Skewed Data

• Re-sampling: the dataset we have might have or counters an imbalanced feature (for example) more normal traffic than an actual attack event is logged. The above-mentioned

biases are combated with synthetic minority over-sampling techniques or could also but not limited to under-sampling of the majority class.

• Class weights: Resampling is not the best strategy; weighting, however, could be done in training to allow biased representation for the classes over a majority.[12]

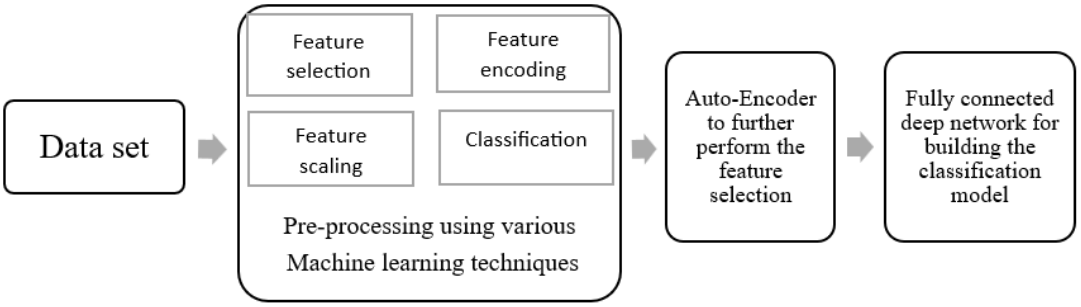

Figure. System Architecture with flow of data

Architecture of Autoencoder

An autoencoder is an unsupervised neural network model for detecting anomalies in the realm of unknown network attacks. It consists of the encoder and decoder as the main components of an autoencoder.[11]

A. Encoder

The latent space is reduced from the original dimension in which input data are fed into an encoder, creating the representation that learns the attributes from the input data which are the most important.

• Feature Input Layer: The input size is same as that of features in the dataset (i.e., 41 for NSL-KDD).

• Hidden Layers: Some dense layers having a gradual decrease in the count of neurons. The example architecture could be:

- 1st hidden layer: 128 neurons
- 2nd hidden layer: 64 neurons
- 3rd hidden layer: 32 neurons

• Activation Functions: Re LU (Rectified Linear Unit) is often used to inject non-linearity, which allows the network to learn complicated patterns in the data.

B. Decoder

Decoding: The decoder retrieves the input from the compressed format.

- Hidden Layers: These layers are organized symmetrically to the encoder. The number of neurons grows to have a gradual number that might be bigger than the corresponding input size.

- Activation Functions: The activation functions of these output layers will therefore depend features normalized or standardized

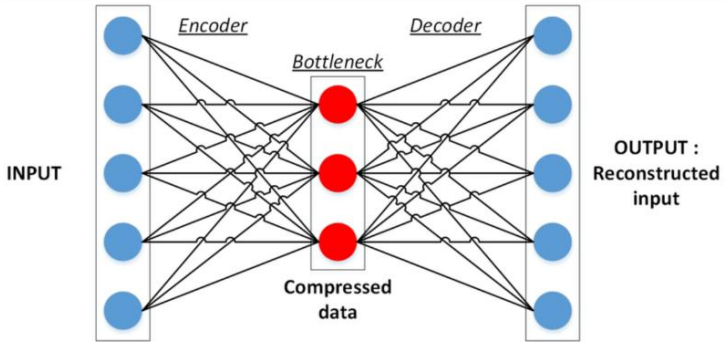

Figure. Single layer Autoencoder Architecture

- Defining the Loss Function and the Optimizer

- Loss Function: Mean Square Error is the property according to which the degree of deviation from the input output reconstruction during training, with autoencoder models for mostly regression tasks and sometimes for anomaly surface detection, should be recorded.

- Optimizer: Adam optimizer has been selected owing to its superior efficiency in dealing with sparsity as well as noise robustness.

Training Process

The model will primarily learn from the unsupervised form of normal patterns, so that later, when anomalies are detected, they can be above a detectable threshold.[10]

- The batch size would entirely depend on the hardware and memory constraints, usually ranging somewhere between 32 and 128.

- Now the default learning rate for Adam would be 0.001, but minor tweaks here will help the model to converge.

- The model is usually trained for 50 to 100 epochs using early stopping, in order not to overfit when no improvement is found on the loss of the validation set.

- Stopping Criteria: Here, training stops after a certain number of epochs (usually 10 in this case) without improvement in validation loss. This would cause the training to be terminated. This prevents the overfitting of training data with the model.[15]

Method for Anomaly Detection

Once trained, the autoencoder can determine the anomalies based on the reconstruction error computed for each individual instance on a dataset.

- Reconstruction Error: each of the test instances is reconstructed by the autoencoder for input and computes the reconstruction error, which in general is expected to be different from the original and the reconstructed data in the form of MSE.

• Threshold Setting: Set a threshold for reconstruction error, instances whose reconstruction errors exceed will be classified as anomalies (likely attacks). Several methods exist to set a threshold:

➢ K-fold_cross-validation: Optimal estimations are allowed for determination of false positive and false negative by using the training dataset to calculate a threshold.

➢ Percentile thresholds: threshold at the top 5 percent of errors from the normal traffic from the validation set.

Performance Evaluation

• Metrics: The standards on which a performance evaluation of an autoencoder for network intrusion detection can be based are: accuracy, precision, recall, F1-score and area under receiver operating characteristic curve (AUC).[14]

• Assessment model confusion matrix takes normal traffic and attack traffic as input to differentiate between their types of deception-true positives, false positives, true negatives and false negatives.

4. Results and Discussion

This particular fragment bears the DOI in performance assessments among the deep neural models via an autoencoder in the NSL-KDD dataset. The model was said to be efficient in generating anomalies but it was still quite competitive with other classical models.

Performance Evaluation

Initial training session of the deep autoencoder for 100 epochs. It was formed into a 32-Well Ends used in the training process and further reduced into smaller batches to evaluate its performance gain and incidence on accuracy, precision, recall, and F1-score, all captured in tabular format together with Figures 2 and 3 onto a comparative database with classic models such as decision trees and support vector machine (SVM) for analysis.[7]

Metric	Training (%)	Validation (%)
Accuracy	98.52	98.50
Loss	0.0781	0.0704
Max Validation AUC	0.9848	0.9848

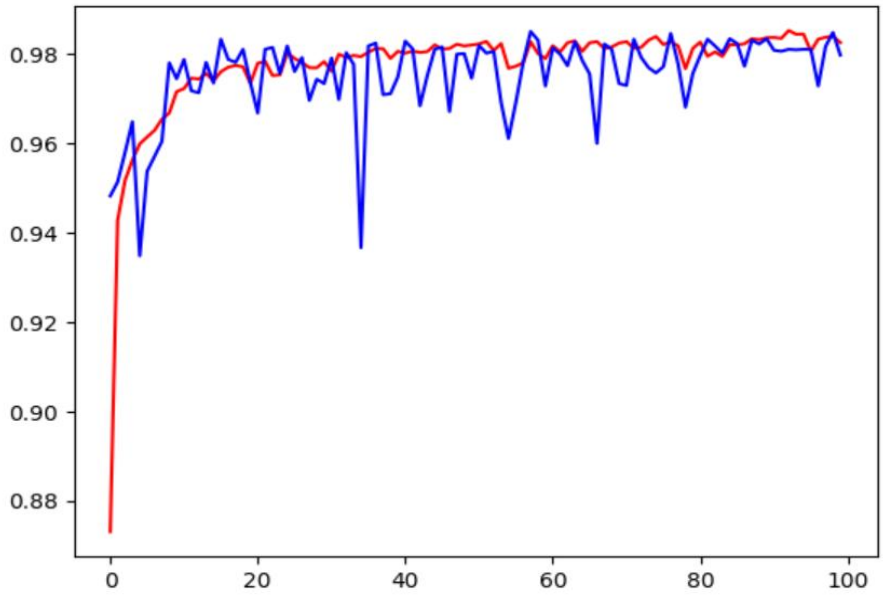

Figure.1

Model Performance

• High Accuracy: The deep autoencoder achieved a maximum training accuracy of 98.52% and validation accuracy of 98.50%, which shows its ability of learning and generalizing effectively on NSL-KDD dataset.

• Low Loss: Training loss and valis is shown to converge largely as 0.0781 and 0.0704 respectively, indicating the strong convergence achieved by the model even without overfitting.

• Stability: Training and validation curves display consistent improvement with minimal fluctuations, thus indicating stable training dynamics.

Key Observations

• The anomaly detection determines the anomaly by reconstruction errors, showing improved performance as compared to typical techniques such as the decision tree and SVM.

• Scalability-in the presence of dimensionality by using the unsupervised part of its anomaly detection, making the high-dimensional query noise very efficiently worked. Compared with Baseline Models.

Table 2 compares the proposed model's performance with baseline techniques in terms of accuracy.

Model	Accuracy (%)
Decision Tree	89.4
Support Vector Machine	90.1
Deep Autoencoder	98.50

Advantages or Benefits:

• Strong Detection: Algorithms based on deep autoencoders have been found to surpass traditional approaches in terms of both high detection rates for known and unknown attacks.[6]

• Generalization: This model has proven to resist previously unseen attack patterns, making it effective under varying threats.

• Such unsupervised applications would be less connected to using labeled data and eventually prove great in terms of real, practical uses of those benefits that appear.

Disadvantages:

• Encrypted Traffic: The model has not been tested on any specific encrypted traffic pattern. It presents specific hurdles for the feature extraction process and for the anomaly detection process.

• Computational Complexity: Heavy computational resources to train the deep autoencoder, which depends on the scalability for real-time deployment.[3]

Future Directions:

• Other preprocessing techniques for aliasing to boost the study on encrypted data.

• Hybrid formations between autoencoders and supervised classifiers to taint detection performance for anomaly detection.[4] Live emission and performance testing in a real network.

Optical Insights

• Training and validation curves: These can depict the loss and accuracy against epochs in a graph that will give a visualization of the convergence of the trained model.

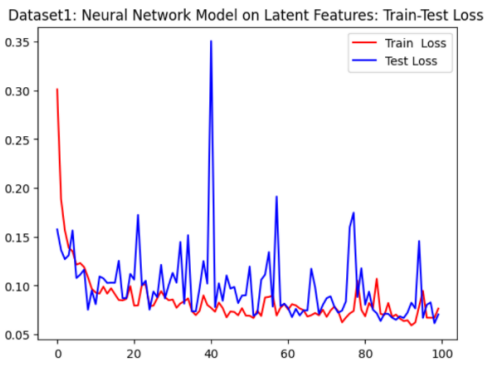

Figure.2

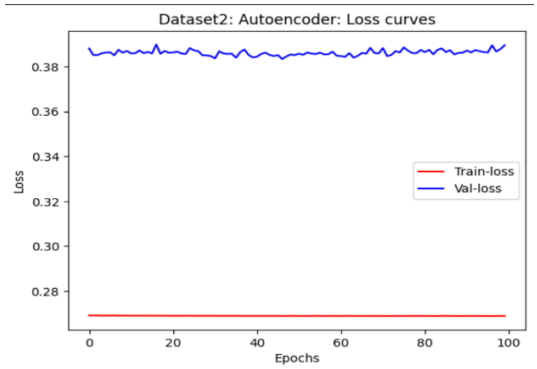

Figure.3

ROC Curves: The Receiver Operating Characteristic (ROC) curve for the validation set may also be used as an indicator of how much discrimination power the model has: a very high-AUC probability measure such.[7][14]

5. Conclusion

Presenting a method for deep autoencoder-based investigation of anomalies for an Intrusion Detection Systems (IDS) using the NSL-KDD dataset for performance evaluation in this paper. The proposed model yields impressive results-an accuracy in validation of 98.50%-and significantly surpasses older techniques such as Decision Trees (89.4%) and Support Vector Machines (90.1%). Such results emphasize that deep learning detection techniques do not include many known and new attack types. The superior detection capability can probably be ascribed to the network's capacity to learn high dimensional representations. Further confirmation about model accuracy and loss being generally low and stability throughout training suggests that the model is feasible for application within IDS.

However, the deep autoencoder approach also has several constraints. The model was not tested with encryption traffic, which may impact the model's features extraction and detection performance in situations with encrypted traffic. Besides, the constructions of deep autoencoders consume significant computational resources, posing scaling concerns when deploying an online model to large-sized networks. Future works will attend these limitations, including encrypted traffic, testing hybrid models merging features from both autoencoders and supervised classifiers, and real-time evaluation in dynamic network environments. More research efforts will acquire improvements on computational efficiency and flexibility in the face of new attacks.

References

1. F. Masoodi, A. M. Bamhdi, and T. A. Teli, "Machine Learning for Classification Analysis of Intrusion Detection on NSL-KDD Dataset," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 12, no. 5, pp. 110–118, 2021.
2. K. N. Deepa and P. T. Venkata Krishna, "Intrusion detection using autoencoders in high-dimensional feature space," IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 3, pp. 1–12, Mar. 2020.
3. H. M. Chandrasekhar and A. Gupta, "Anomaly detection in network systems using LSTM autoencoders," in Proc. IEEE Int. Conf. Big Data and Smart Computing (BigComp), Bangkok, Thailand, 2020, pp. 311–318.
4. X. Yin, T. Zhang, and Z. Chen, "Enhancing anomaly detection through feature reduction techniques," IEEE Communications Surveys & Tutorials, vol. 22, no. 3, pp. 1427–1447, 2020.
5. I. A. Ismail, M. M. Hamid, and H. Hassan, "Anomaly-based intrusion detection system using deep learning," IEEE Access, vol. 7, pp. 166661–166672, Dec. 2019.
6. N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41–50, Feb. 2018.
7. A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection," in Proc. 9th EAI Int. Conf. Bio-Inspired Information and Communications Technologies, New York, NY, USA, 2016, pp. 21–26.
8. T. Chen and C. Guestrin, "XG Boost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD), San Francisco, CA, USA, 2016, pp. 785–794.
9. M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," in Proc. 12th USENIX Symp. Operating Systems Design and Implementation (OSDI), Savannah, GA, USA, 2016, pp. 265–283.

10. D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," arXiv preprint arXiv:1312.6114, Dec. 2013.
11. Y. Bengio, "Learning deep architectures for AI," Foundations and Trends in Machine Learning, vol. 2, no. 1, pp. 1–127, Jan. 2009.
12. M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in Proc. IEEE Symp. Computational Intelligence for Security and Defense Applications (CISDA), Ottawa, ON, Canada, 2009, pp. 1–6.
13. J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," Journal of Machine Learning Research, vol. 7, pp. 2721–2744, Dec. 2006.
14. V. N. Vapnik, The Nature of Statistical Learning Theory, 2nd ed. New York, NY, USA: Springer-Verlag, 2000.
15. S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.