**Network Intrusion Detection Analysis using Deep Learning Models**

* **Vijay Anirudh Aithi – 0036728672**

**Abstract: Information is the key in this modern world, using it in various ways to generate money and develop technologies that satisfy today's needs. As necessary, it should be protected. Many cyberattacks' sole motive is to gain information; to do that, they must sneak into the network. NIDS was introduced to detect these malicious intrusions. NIDS detects intrusion by signature-based by analyzing the packets that flow from one network to another. A packet that is not regular or accepted will be blocked or alerted to the SIEM based on the NIDS capability. After considering the weakness of signature-based NIDS, Machine and Deep Learning extract the best results. Deep learning proved to provide quality results because of feature selections, so it is now widely used to train models to detect anomalies in the network. The proposed work will evaluate the various DL Models and their performance in detection. The NSL-KDD dataset, taken from the University of New Brunswick website, is used to investigate. The dataset contains 42 features and 23 types of attacks.**

**Keywords:** NIDS – Network Intrusion detection system, SIEM – Security information and event management, Deep Learning, Machine Learning, NSL-KDD, Dataset, packets.

**Overview and Motivation:** In the evolving cybersecurity space, the need for intrusion detection is critical. Incidents will resonate with the need and serve as reminders of the aftermath of an attack. Evaluating notable attacks such as the Equifax data breach, the WannaCry ransomware attack, the SolarWinds supply chain compromise, the Target data breach, and the NotPetya cyberattack divulges the profound impact of lack of timely detection.

Target Data breach(2013): Adversaries gained access to the network through a third-party vendor, eventually stealing the information of payment details and their types of 40 million customers.

NotPetya(2017): A destructive malware disguised as ransomware was intended to target Ukrainian organizations but spread rapidly all over the world.

Thus, Prevention is better than cure. The idea of inventing IDS originated from a research paper by James P.. " How to detect intrusions in audit files." Between 1984 and 1986, Dorothy Denning and Peter Neumann developed the first model of IDS, a prototype named Intrusion Detection Expert System (IDES). There are two types of IDS, which are host-based and network-based.

HIDS: A host-based intrusion detection system sits at any described position in \the network to analyze all the packets passing through it. Also, it can alert the SIEM and block all suspicious packets.

NIDS: A network intrusion detection system usually sits at one end of the network, and all the data visiting the network should pass through; NIDS can alert the system that it has found suspicious packets.

Manual inspection of packets is tedious to detect anomalies, and automation is required. Famous automated tools like SNORT and NIDES are rule-based Intrusion detection systems that were a commercial breakthrough at the time they were invented however as time passed networks tended to upgrade to meet public demand and technology requirements, and as a consequence, these tools failed to adapt to those ascents and another major problem was to bless the with knowledge frequently to avoid zero-day attacks, While signature-based detection faces knowledge-based issues, using Machine and Deep learning models is critical. However, using them is also a complex task without adequate pre-processing of the data, but we can extract better results with the use of processing.

**Literature Review and Contributions: The paper proposes to develop an IDS using DNN(Deep et al.), as it generates information on the number of input and output neurons utilized for the KDD Cup 1999 dataset and the five-layer DNN model. This model is trained with the help of the Backpropagation technique. The paper also examines the potential results of various machine learning and deep learning model applications for intrusion detection. A worthy point in this paper is the comparison of DNN models with traditional AI classifiers, where DNN outperforms others. The fact that the trials have yet to be carried out on more recent datasets is a drawback. The research presents a strong argument for utilizing DNNs in constructing intrusion detection systems[1].**

**This paper also uses the KDD Cup 1999 dataset to compare the performance of CNN(Convolutional Neural Networks) and DNN(Deep Neural Networks.). The CNN consists of convolutional and pooling layers, while the DNN consists of five hidden layers; comparing results, the F1 score, accuracy, precision, and recall score were the evaluation metrics. This study uses the KDD Cup 1999 dataset to compare the efficacy of convolutional neural networks (CNNs) and deep neural networks (DNNs) for intrusion detection. Using the methodology, a CNN model with convolutional and pooling layers and a DNN model with five hidden layers are developed. Evaluation criteria such as F1-score, recall, accuracy, and precision are employed. The accuracy of the CNN is higher than the DNN's, at 99.12% as opposed to 79.26%. This paper's pertinent literature review also covers the use of deep learning techniques for network intrusion detection systems. The usage of just one older dataset is a limitation. It offers valuable information about CNNs and DNNs for intrusion detection[2].**

**This paper presented a compulsive structure for HIDS and NIDS using DNN, which employs three layers, primarily the input, five hidden, and output layers. These diverse layers provided smooth recognition of attacks and improved accuracy.**

**Finally, this paper further emphasizes the layer-to-layer explanation. i.e., the input layer consists of 41 neurons for KDD Cup99 and another 41 for NSL-KDD, 43 for UNSW-NB15, 17 for WSN-DS, and 77 for CICIDS 2017. Also, output layers are 1 for binary classification, 5 for multi, and 5,8,10 neurons for respected datasets mentioned above in the input layers.[9]**

**Improvement of NIDS through deep learning by implementing the RNN for the hidden attacks and synthetic signatures to improve the accuracy. This paper's authors were S.M.Sohi et al.[10]**

**Based on the knowledge from papers, CNN was implemented to experiment on the NSL-KDD dataset, an upgrade of the KDD Cup 1999. The dataset was obtained from University of New Brunswick, made in a laboratory environment by using VMs, attacking them using other VMs, and finally capturing the network in the form of packets. The features of NSL-KDD include**

**A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated**

**A screenshot of a computer program

Description automatically generated A group of black and white text

Description automatically generated**

**The data processing would be different from the papers used for review because extraction features from the dataset are bound to one's approach to solving the problem. This proposed work used a binarized label, a label encoder, and one hot encoder to preprocess and normalize the dataset and feed it to the various DL models to obtain the results.**

**Technology Details: This report implemented CNN, RNN, GRU, CNN-LSTM to evaluate the performance and compare which one is best, further machine learning algorithms are also described to improve background on this subject.**

**Support Vector Machine(SVM): The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.**

**The main goal is to find a hyperplane that has a maximum margin. These boundaries are constructed in the dimensional space or in the kernel space to obtain the best result in the case of non-linear data. Margin can classify the new data point using correctness[3][4][5]**

A black text on a white background

Description automatically generated

**Above picture is the mathematical representation.**

A diagram of a variety of lines

Description automatically generated with medium confidence

**Above picture represents how data is visualized in the multi-dimensional space.**

**K Means Clustering: K-means clustering is a vector quantization method, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. The main goal is calculating the distance between a centroid assigned to the cluster and each data point. Every cluster containing a higher weight will get assigned to the cluster with a more significant internal variance.[6]**

A diagram of a graph

Description automatically generated

A group of dots and crosses

Description automatically generatedA diagram of different colored dots and crosses

Description automatically generated

**Above Pictures represent how K means clustering happen visually.**

**Data normalization holds utmost importance before training the model; the results will be more efficient based on how efficiently we preprocess the data. This report used a Label encoder, Label binarizer, and one hot encoder to perform Normalization operations.**

**LabelEncoder: Label encoding is a technique that converts categorical data into numerical data to fit into the models and only takes numerical as input.**

**LabelEncoder is a utility class that will help normalize labels so that the resulting labels only contain values between 0 and n classes - 1.**

**A screenshot of a computer

Description automatically generated**

**Above represents before Label Encoding**

**A screenshot of a computer screen

Description automatically generated**

**This is after Label Encoding.**

**OneHot Encoding: It also does the same task as Label Encoder by changing the categorical data int numerical data while it is same operation Onehot encoder works in a different way, it also helps to avoid the problem of ordinality[7]**

A screenshot of a computer screen

Description automatically generated

**This picture represents after OneHot Encoding.**

**CNN: CNNs are a particular kind of neural network that is excellent at processing input with a grid-like layout, like photographs. Because of this, they are excellent for computer vision applications.**

**Convolutional, pooling and fully linked layers comprise a CNN's architecture. Convolutional layers use filters to extract significant features from the input data and learn them. As a result, manual feature engineering is no longer necessary.**

**Pooling layers combine the outputs of multiple neuronal clusters into one output, thereby downsampling the data. This lessens the need for overfitting and processing demands. Max, average, and sum pooling are examples of everyday pooling operations. In order to produce class scores or predictions based on the learned features, fully linked layers link every neuron from the previous layer to the subsequent layer.**

A diagram of a diagram of a complex layer

Description automatically generated with medium confidence**[8]**

**Above represents sample presentation of CNN.**

**These are the metrics used in this model:**

* Loss Function : Categorical Cross Entropy
* Optimizer: Adam
* Metrics: Accuracy
* Epochs: 100

**RNN: RNNs are a type of neural network designed to handle data sequentially with the help of loop architecture; this recurring or feedback loop will allow information to persist, enabling the neural network to capture temporal dependencies and process sequences of various lengths.**

**The architecture behind the RNN is that every single neuron receives input from the entire network, including output from the previous inputs. This allows the RNN to form a memory from these recurrent connections and maintain a hidden state that encodes information about the sequence seen so far.**

**The hidden state of RNN can be computed as follows[11]:**

**A screenshot of a math problem

Description automatically generated**

**Our model summary follows:**

**A screenshot of a computer

Description automatically generated**

**Metrics used in our RNN:**

* Loss Function: Categorical Cross Entropy
* Optimizer: Adam
* Metrics: Accuracy
* Epochs: 100

**GRU: GRU is a Gated Recurrent Unit derived from RNN, while the development cause for GRU is limitations in RNN, which are vanishing gradient problems; this fault in RNN makes it forget the memory learned from its previous inputs. GRU is known for its efficiency in capturing long-range dependencies in sequential data.**

**The Mathematical representation, model summary of GRU follows [12] :**

**A screenshot of a computer program

Description automatically generatedA screenshot of a math problem

Description automatically generated**

**Our Metrics of GRU follows:**

* Loss Function: Binary Cross Entropy
* Optimizer: Adam
* Metrics: Accuracy
* Epochs: 100

**CNN-LSTM: CNN-LSTM is a hybrid architecture that blends the strengths of CNN and LSTM to obtain spatial and temporal dependencies at the same time in sequential data; the architecture behind CNN-LSTM comprises gathering spatial feature extraction from data with the help of Convolutional layers and a sequential way of learning with the help of LSTM.**

**Mathematics behind CNN-LSTM follows[13]:**

**A math equations and formulas

Description automatically generated**

**A white text with blue text

Description automatically generated with medium confidence**

**Model Summary:**

**A screenshot of a computer

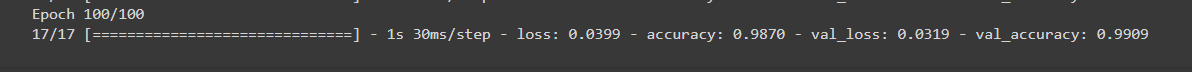
Description automatically generated**

**Metrics Followed:**

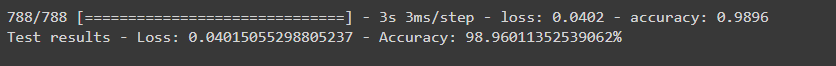
* Loss Function: Categorical Cross Entropy
* Optimizer: Adam
* Metrics: Accuracy
* Epochs: 100

**Results:**

**Train Results (CNN):**

****

**Test Results (CNN):**

****

Below represents **Loss vs Epoch and vice-versa (CNN):**

A graph of a graph

Description automatically generated

A graph of loss and loss of a train

Description automatically generated

**Train Results of RNN:**

**A black text with numbers

Description automatically generated with medium confidence**

**Test Results of RNN:**

**A close up of numbers

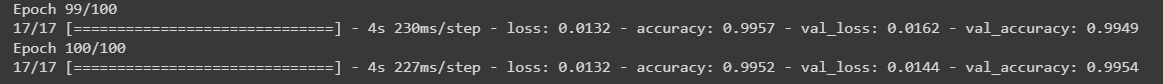
Description automatically generated**

**A graph with blue and orange lines

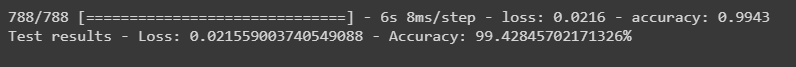
Description automatically generatedA graph of a graph

Description automatically generated** Below represents **Loss vs Epoch and Accuracy (RNN):**

**Train Results of GRU:**

****

**Test Results of GRU:**

****

Below represents **Loss vs Epoch and Accuracy (GRU):**

**A graph with a line

Description automatically generated**

A graph of loss and loss of a train

Description automatically generated with medium confidence

**Train Results (CNN - LSTM):**

**A black text with numbers

Description automatically generated with medium confidence**

**Test Results (CNN - LSTM):**

**A number and line on a white background

Description automatically generated**

**A graph of loss and loss of a train

Description automatically generatedA graph of a graph

Description automatically generated**Below represents **Loss vs Epoch and Accuracy (CNN-LSTM):**

**Conclusions and Future Scope: Using the NSL-KDD dataset, 42 features were extracted, labelled attack types, and categorized into their respective types using normalizers. After that, the normalized data was used and fed to CNN, RNN, and GRU, finally CNN-LSTM as the Input, and the train, test results were obtained.**

|  |  |  |
| --- | --- | --- |
| **Model:** | **Train Accuracy** | **Test Accuracy** |
| **CNN** | **98.91%** | **98.82%** |
| **RNN** | **99.57%** | **99.43%** |
| **GRU** | **99.54%** | **99.42%** |
| **CNN-LSTM** | **99.53%** | **99.49%** |

**After comparing the results, we can see that CNN-LSTM outperformed others; as we know, the limitations of CNN, as it is basic but combining with LSTM made it to achieve greater result.**

For the future scope, Multi-class will be implemented and employing of GAN with combination of NSL\_KDD and other modern Datasets.

**References:**

[1] General overview of CNN architecture: Krizhevsky, A., Sutskever, I., & Hinton, G.E. (2012). ImageNet classification with deep convolutional neural networks. Advances in neural information processing systems, 25.

[2] Usefulness for computer vision tasks: Rawat, W., & Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. Neural computation, 29(9), 2352-2449.

[3] [online] Available: <http://www.statsoft.com/textbook/support-vector-machines/>.

[4] [online] Available: <https://www.datacamp.com/community/tutorials/support-vevtor-machines-r/>.

[5] P. Flach, "Machine Learning – The Art and Science of Algorithms that Make Sense of Data" in , Cambridge:Cambridge University Press, pp. 212, 2012.

[6] Sharma, N. (2023) *K-Means Clustering explained*. https://neptune.ai/blog/k-means-clustering.

[7] 6.9. Transforming the prediction target (y). (n.d.). Scikit-learn. <https://scikit-learn.org/stable/modules/preprocessing_targets.html#preprocessing-targets>

[8] Chatterjee, C. C. (2021, December 11). Basics of the classic CNN - towards data science. Medium. <https://towardsdatascience.com/basics-of-the-classic-cnn-a3dce1225add>

[9] R.Vinayakumar, Mamoun Alazab, K.P Soman, Prabaharan Poornachandran,Ameer AlNemrat, Sitalakshmi Venkatraman, “Deep Learning Approach for Intelligent Intrusion Detection System” ,2019

[10] S. M. Sohi, J. Seifert, and F. Ganji, “RNNIDS : Enhancing Network Intrusion Detection Systems through Deep Learning,” Comput. Secur. pp. 102-151, 2020.

[11] Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep Learning (Vol. 1). MIT press Cambridge.

[12] Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.

[13] Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., & Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 2625-2634).