Using a Neural Network to Design and Train a Model for an Intrusion Detection System (IDE)

Victoria Hektor – p2629898@my365.dmu.ac.uk
AI MSc Student, De Montfort University

Introduction

This project aims to create and assess a specialised neural network for identifying network intrusions. Its primary objective is to differentiate between malicious connections (intrusions or attacks) and legitimate connections (normal network behavior). With emphasise on dataset preparation, pre-processing, and training. Focusing on the neural network design, and parameter selection for accurate intrusion detection. [2]

AI is crucial in Cyber Security and Intrusion Detection due to its ability to enhance threat detection, real-time monitoring, and rapid response, making it a vital tool for defending against evolving cyber threats. Being aware of the threat landscape and the available datasets is imperative to success in this project. [1]

Aims

- Utilise the pre-processed "KDD Cup 99" dataset, leveraging 10% of the available data. [6]
- Extract and normalise the data for use in neural network training. [2]
- Develop a binary learning script for neural network implementation, aiming for good generalisation. [7]
- Explore the possibility of creating a multiclass script for neural networks, contingent upon available time. [3]
- Implement a robust evaluation process to interpret matrices and output results using the 'perform' function, ensuring precise assessment of neural network functionality for the intended purpose.

Pre-Processing

- Extracted data from a TXT file and transferred it to an Excel format for streamlined pre-processing.
- Verified and corrected column names for accuracy.
- Transformed categorical columns into numerical representations.
- Applied data normalisation within the [0, 1] range to facilitate binary classification.

Challenges

Classification Dilemma: Choosing between binary and multiclass classification posed challenges, impacting model complexity and accuracy, particularly when based on practicality. [8]

Algorithm and Topology Selection: Selecting suitable algorithms and network topologies from numerous options was a complex decision that greatly influenced model performance.

Time Constraints: The project encountered various time constraints that affected all phases of design and implementation.

Variable Naming Clarity: Ensuring clear and intuitive variable names was essential for code readability and comprehension.

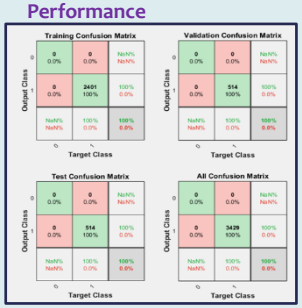
Code Structure: Implementing a separation of concerns within the code structure enhanced readability, debugging, and coding best practices.

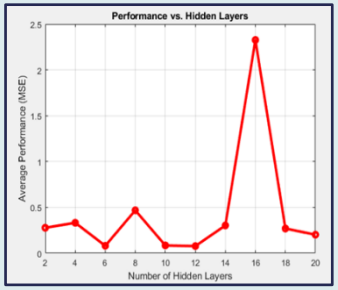
Data Preprocessing Challenges: Data preprocessing difficulties led to issues with neural network training scripts. These challenges were eventually overcome.

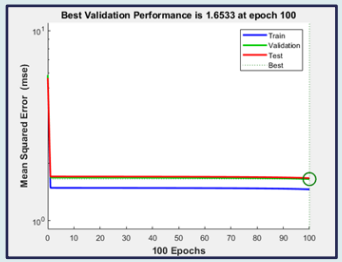
Binary Classification Output

The 100% output was achieved using the Gradient Descent training algorithm, coupled with the weight decay method.[4]

(L, R & top to bottom) Fig.1 – Overall Performance of Iterations, fig2 – Confusion Matrix, fig3 – Validation Performance







Model Development Process

Starting Simple: Initially, I began with a basic neural network model to establish a fundamental understanding.

Exploration and Complexity: My journey led me to explore various capabilities of Matlab, resulting in multiple versions of the model.

Reset and Refocus: I recognized the need to reset and start with a simpler foundation to build a better understanding.

Version Summary: Several iterations of neural network models were created, exploring aspects such as training algorithms, cross-validation, overfitting prevention through dropout and weight decay, ultimately culminating in a model achieving a 100% accuracy rate in the confusion matrix.

Achieving the 'Perfect Model': Ultimately, I achieved a model that produced a 100% accuracy rate in the confusion matrix. Concerned about overfitting, I introduced the weight decay method to evaluate its impact. Surprisingly, the model maintained a 100% accuracy rating. [4]

Utilising For Loops: To streamline the development process, I employed for loops to iterate through multiple configurations, calculate errors, and provide error and success messages throughout each actionable event in the script, ensuring the script's functionality.

Your experiences reflect a journey of exploration, learning, and continuous improvement in developing your neural network models for intrusion detection.

Architecture

The ideal architecture was deemed to be 10 hidden layers with a total of 20 neurons.

Fig.4 – 10 HU, 20 Neurons:

Conclusion

AI plays a vital role in Cyber Security and Intrusion Detection, enhancing threat detection and response. The journey through model development led to an impressive achievement of a 100% accuracy neural network model. Further research will focus on completing a multiclassification model to expand the scope of this work. [5]

References

[1]A. Divekar, M. Parekh, V. Savla, R. Mishra, and M. Shirole, “Benchmarking datasets for Anomaly-based Network Intrusion Detection: KDD CUP 99 alternatives,” *IEEE Xplore*, Oct. 01, 2018. https://ieeexplore.ieee.org/abstract/document/8586840 (accessed Feb. 02, 2022).

[2]C. Han, Y. Lv, D. Yang, and Y. Hao, “An intrusion detection system based on neural network,” *ieeexplore.ieee.org*, Aug. 19, 2011. https://ieeexplore.ieee.org/abstract/document/6025886 (accessed Oct. 18, 2023).

[3]D. Elizondo, “The linear separability problem: some testing methods,” *ieeexplore.ieee.org*, Mar. 06, 2006. https://ieeexplore.ieee.org/abstract/document/1603620 (accessed Oct. 18, 2023).

[4]G. Zhang, C. Wang, B. Xu, and R. Grosse, “Three Mechanisms of Weight Decay Regularization,” *openreview.net*, Sep. 27, 2018. https://openreview.net/forum?id=B1lz-3Rct7 (accessed Oct. 18, 2023).

[5]I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,” *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, 2018, doi: https://doi.org/10.5220/0006639801080116.

[6]M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Jul. 2009, doi: https://doi.org/10.1109/cisda.2009.5356528.

[7]S. El-Sappagh, A. S. Mohammed, and T. A. AlSheshtawy, “Classification Procedures for Intrusion Detection based on KDD CUP 99 Data Set,” *Social Science Research Network*, Jun. 10, 2019. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3401645 (accessed Jul. 20, 2023).

[8]Z. Liu, Y. Cui, and A. Chan, “Improve Generalization and Robustness of Neural Networks via Weight Scale Shifting Invariant Regularizations,” Aug. 2020. Accessed: Oct. 18, 2023. [Online]. Available: https://arxiv.org/pdf/2008.02965