MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Sixth Semester

BTech in CSE (AI & ML)

Department of Computer Science & Engineering

[Jan – May 2024]

Deep Learning Project

CSE 3281

Cyber Attack prediction Using Deep Learning techniques.

-Joy Podder 210962184

-Anurag Kasat 210962180

-Rinchen Norbu 210962210

Introduction:

Cybercrime is a serious and growing threat to the security and stability of the online world. Cyber criminals use various methods and tools to exploit users, steal data, disrupt services, and cause harm to individuals, organizations, and nations. Hence detecting and preventing cybercrime is a challenging and crucial task.

One of the promising approaches to cybercrime detection is the use of deep learning techniques. Deep learning is a branch of machine learning that uses multiple layers of artificial neural networks to learn from large amounts of data and perform complex tasks. Deep learning has shown remarkable results in various domains, such as computer vision, natural language processing, speech recognition, and more. Some of the advantages of deep learning for cybercrime detection are:

1. It can handle high-dimensional, heterogeneous, and noisy data that are common in cybercrime scenarios.

2. It can learn from both labeled and unlabelled data, which can reduce the need for manual annotation and increase the coverage of detection.

3. It can extract meaningful and robust features from the data, which can improve the accuracy and efficiency of detection.

4. It can adapt to changing patterns and behaviors of cyber criminals, which can enhance the resilience and scalability of detection.

This research paper aims to provide a comprehensive overview of the state-of-the-art deep learning techniques for Network Intrusion detection, as well as the challenges and opportunities for future research.

Objectives:

Review existing literature on cybercrime detection using deep learning, categorizing them according to:

- Data sources utilized (e.g., network traffic, user behavior logs, malware samples).
- Types of detection tasks addressed (e.g., intrusion detection, fraud detection, malware detection).
- Deep learning models employed (e.g., convolutional neural networks, recurrent neural networks, generative adversarial networks).

Evaluate the performance and effectiveness of different deep learning models specifically for network intrusion detection, considering:

- Various metrics such as accuracy, precision, recall, F1 score, and area under the ROC curve.
- Benchmarks and datasets commonly used in the field of network intrusion detection (e.g., NSL-KDD, UNSW-NB15, CICIDS2017).
- Comparison against traditional machine learning approaches and rule-based systems to highlight the advantages of deep learning.

Identify trends and gaps in current research on cybercrime detection using deep learning, particularly focusing on network intrusion detection, and propose potential areas for future research and improvement.

Gaps:-

1. **Limited Availability of Real-Time Data:** Obtaining real-time data for network intrusion detection may pose challenges due to privacy concerns, data access restrictions, and the need for specialized infrastructure to capture and process streaming data. As a result, the project's findings may not fully reflect the complexities of real-world deployment scenarios.
2. **Data Imbalance and Representation:** Datasets commonly used for training deep learning models in network intrusion detection may suffer from class imbalance, where certain types of cyber threats are underrepresented. This imbalance can affect the model's ability to generalize to unseen data and may lead to biased performance evaluation results.
3. **Generalization to Diverse Attack Scenarios:** Deep learning models trained on benchmark datasets may struggle to generalize to diverse and evolving attack scenarios, particularly in the face of adversarial attacks and zero-day exploits. The project's evaluation may not fully capture the robustness of the models in detecting novel and sophisticated cyber threats**.**
4. **Dependency on Feature Engineering and Preprocessing:** Despite their ability to automatically learn features from raw data, deep learning models may still rely on feature engineering and preprocessing techniques to extract relevant information from complex input sources such as network traffic logs and packet captures. The effectiveness of the models may be influenced by the quality and relevance of the engineered features.
5. **Dependency on Benchmark Datasets and Evaluation Metrics:** The project's findings may be influenced by the choice of benchmark datasets and evaluation metrics used for performance assessment. Differences in dataset characteristics and evaluation criteria across studies may limit the comparability and generalizability of the results.

Literature review:

Cyber-attack method and perpetrator prediction using machine learning algorithms.

[1] suggests a method which uses ML techniques like SVM, naïve bayes, decision trees, random forest, logistic regression to predict whether a cybercrime will take place or not, data

from previous incidents such as the gender, age, crime committed, income, job, attack method are used to construct the model.

Limitation: The limitation of this paper is that it does not employ any DL techniques to calculate results, location data can also be used to enhance the results, feature importance can by using correlation values can be used to make more efficient models using only the important features.

Deep Learning Methods for Cybersecurity and Intrusion Detection Systems

[2] claims to leverage AI ML and DL techniques for network intrusion detection and provides a framework of deep learning for cyber security applications. Various network architectures such as CNNs, RNNs, DBNs, DBM, Auto encoders GANs have been compared. Before making the model the paper suggests to employ feature engineering techniques to further enhance the performance of models.

Limitation: Concrete results are not shown in this literature and it also suggests that there is room for research on a variety of open issues such as knowledge based and behavioural approaches in intrusion detection.

Web attack detection using deep learning models

[3] uses three deep learning models ANN,CNN and RNN to detect web attacks, further it also identifies the time at which the attack occurred on the users using log information. The dataset is cleaned, encoded and normalized to create a uniform format and then passed on to the neural networks.

Limitations: The paper does not discuss the future work, challenges, or future directions of the proposed scheme. It only claims that the RNN model outperforms the other models, without providing any statistical significance tests or error analysis, it does not show how the RNN models are better than ML or data mining techniques. It also does not address the issues of scalability, robustness, or generalizability of the scheme.

A deep learning framework for predicting cyber attacks rates:-

[4] presents a novel deep learning framework named BRNN-LSTM (Bi-directional Recurrent Neural Networks with Long Short-Term Memory) for predicting cyber attack rates based on time series data. The framework is designed to capture the statistical properties inherent in cyber attack rate data and offers users flexibility in choosing the number of Long Short-Term Memory (LSTM) layers incorporated into the BRNN structure.

[4] highlights two main contributions. Firstly, it introduces the BRNN-LSTM framework, emphasizing its adaptability to the statistical characteristics of cyber attack rate time series data. The customizable LSTM layers distinguish it from existing models. Secondly, the empirical study utilizes real-world cyber attack rate datasets to demonstrate the superior prediction accuracy of BRNN-LSTM compared to traditional statistical models, including those referenced in existing literature.

Limitations:

o **Limited Generalizability**: The paper's reliance on specific real-world datasets raises concerns about the generalizability of the proposed BRNN-LSTM framework to diverse cyber threat scenarios. Additional testing on varied datasets is necessary to evaluate its applicability across different contexts.

o **Lack of Interpretability**: The inherent black-box nature of deep learning models, including BRNN-LSTM, hinders the interpretability of predictions. This poses a challenge in understanding the rationale behind the model's decisions, a crucial factor in cybersecurity. Addressing this limitation through interpretability measures or further analysis would enhance the paper's contributions.

CyberSecurity Attack Prediction: A Deep Learning Approach

[5] investigates the utilization of deep learning techniques in the prediction of cybersecurity attacks, with a specific focus on introducing novel models based on Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), and Multilayer Perceptron (MLP). These models are meticulously crafted to forecast the potential types of attacks that may occur. The study validates the proposed models using a recently released dataset named CTF, showcasing promising outcomes. Notably, the LSTM model demonstrates superior performance, achieving an f-measure exceeding 93%.

Limitation: One potential limitation of the proposed approach is the reliance on a specific dataset (CTF) for validation. The generalizability of the models may be constrained by the characteristics and diversity of this particular dataset. If CTF does not comprehensively represent the broader landscape of cybersecurity threats, the models' effectiveness in real-world scenarios could be compromised. Addressing this limitation by testing the models on a more diverse set of datasets or incorporating a wider range of cybersecurity scenarios could enhance the robustness and applicability of the proposed deep learning techniques.

Conceptualisation of Cyberattack prediction with deep learning

[6] proposes a new approach for cyberattack prediction using deep learning and clustering techniques. The approach models cyberattack prediction as a multi-class classification problem, where different types of attacks and benign traffic are learned from network data. The approach consists of two stages: unsupervised feature engineering and clustering and supervised deep learning and classification. The unsupervised stage uses Principal Component Analysis (PCA) to reduce the dimensionality of the network features, and Expectation Maximization (EM) algorithm to generate clusters of network traffic based on their latent variables.The supervised stage uses a deep feed forward neural network with rectified linear units (ReLU) as the activation function to learn the compressed representation of each cluster and classify it into one of the attack or benign classes using a Softmax layer4.

The approach is evaluated using two benchmark datasets: CICIDS2017 and UNSW_NB15, which contain various types of evolving attacks and normal traffic.

The approach achieves superior performance over existing models in terms of accuracy, false positive rate, precision, recall, F-measure, and cross entropy.

Limitations:

o The paper does not provide a clear problem statement or research question that motivates the proposed approach.

o [6] does not compare or contrast the proposed approach with existing methods for cyberattack prediction using deep learning, such as convolutional neural networks, long short-term memory networks, or attention mechanisms.

o The paper does not provide sufficient details or justification for the choice of hyperparameters, such as the number of clusters, the number of principal components, the number of hidden layers, and the learning rate.

o The paper does not evaluate the robustness or generalizability of the proposed approach to different types of attacks, datasets, or network environments.

o The paper does not discuss the limitations, challenges, or future directions of the proposed approach.

Real Time Crime Detection Using Deep Learning Algorithm

The project presented in [7] addresses the pressing concern of increasing crime rates and the need for effective surveillance systems to prevent and identify crimes before they occur. Traditional methods of crime prevention, such as CCTV cameras, often require human supervision and are prone to errors due to the inability of humans to monitor multiple screens simultaneously. To overcome these challenges, the authors propose a Real-Time Crime Detection Technique using a Deep Learning Algorithm, specifically utilizing the YOLO (You Only Look Once) object detection algorithm. The project's goal is to develop a system that can monitor real-time videos, detect criminal activities, and alert nearby authorities about the occurrence of a crime along with its current location. The YOLO algorithm is chosen for its efficiency in object detection, as it operates using a single convolutional neural network to predict bounding boxes and probabilities of objects within an image.

The proposed technique involves several steps, including dataset creation, training the YOLOv4 algorithm, testing the model, and implementing real-time detection. The dataset comprises high-resolution images of criminals and weapons, annotated with bounding box coordinates using the YBAT annotation tool. Training of the YOLOv4 algorithm is conducted on Google Collaboratory, utilizing a dataset of criminal and weapon images. The trained model is then tested with unseen data to evaluate its performance.

Performance measures such as Mean Average Precision (mAP), precision, and recall are used to assess the effectiveness of the proposed system. The results indicate that the YOLO-based

approach outperforms existing techniques, achieving a mean average precision of 78.3% on the testing set.

Limitations: Real-Time Crime Detection Technique using the YOLO algorithm has several limitations. These include potential biases in the training data, a trade-off between speed and accuracy, limited object classes in the dataset, privacy concerns related to constant surveillance, hardware limitations affecting performance, environmental factors impacting detection, and the presence of false positives and false negatives. Addressing these limitations requires ongoing research and development efforts to enhance the system's robustness, accuracy, and ethical considerations, along with thorough testing in diverse real-world scenarios.

Detection of Malware Using Deep Learning

[8] discusses the growing threat of cybercrime in today's world and the increasing need for effective cybersecurity measures. It focuses on the application of Deep Learning (DL) techniques to address various cybersecurity challenges such as intrusion detection, malware classification, spam detection, and phishing detection. The primary objective of the project is to develop a new algorithm for detecting malicious cyber-attacks using DL while minimizing mispredictions. The proposed algorithm aims to achieve a higher defensive rate with minimal misprediction levels. The research involves extensive experimentation to adequately address potential adversarial scenarios.

In the related work section, various approaches to phishing detection are explored, including blacklists and whitelists, web page visual similarity, and URL and website content features. Deep Learning is identified as a promising technique due to its ability to automatically detect features without requiring expert input. The problem formation section outlines the challenges of using DL methods for cybersecurity, particularly the risk of misprediction due to internal and external malicious properties. The proposed solution involves careful system evaluation before applying DL methods and training the model with both malicious and benign data to minimize mispredictions.

The base model section describes the use of Deep Learning for malware detection, including pretraining algorithms and fine-tuning techniques. A pseudocode for malware detection using DL is provided, along with defense mechanisms against evasion and poisoning attacks. The simulation setup and evaluation metrics for malware detection are presented. The preliminary results show promising accuracy levels, and future work includes further experiments with larger datasets and other machine learning techniques to enhance the proposed system's efficiency.

Limitations: Firstly, the effectiveness of the proposed algorithm may depend on the quality and diversity of the training data. If the dataset used for training is not representative of real-world cyber threats or is biased in any way, it could affect the algorithm's performance in detecting new and evolving attack techniques. Additionally, the proposed defense mechanisms against evasion and poisoning attacks may not be foolproof and could potentially be circumvented by sophisticated adversaries. Furthermore, the computational resources required for training and

deploying Deep Learning models can be significant, which may limit the feasibility of implementing the proposed solution in resource-constrained environments. Finally, as with any cybersecurity solution, there is always a cat-and-mouse game between attackers and defenders, and it is essential to continually adapt and update the algorithm to counter new threats effectively.

Cyber Security Attack Prediction: A Deep Learning Approach

[8] undertakes a comprehensive exploration of cybersecurity attack prediction, leveraging the power of deep learning methodologies, specifically focusing on Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), and Multilayer Perceptron (MLP) models. Considering the escalating volume, variety, and complexity of cyber threats, there's an urgent need to design more effective prediction models to stop potential attacks. The project's significance lies in its endeavour to bridge this gap by harnessing the unparalleled performance of deep learning techniques, which have garnered considerable attention from researchers due to their success in various prediction-based domains. By delving into the application of LSTM, RNN, and MLP models, the project aims to predict the type of cyberattack likely to occur, thereby empowering organizations, enterprises, governments, and countries to proactively safeguard their systems and networks against malicious activities.

Limitations: despite the promising prospects offered by deep learning approaches, the project encounters several methodological and conceptual challenges. One notable limitation is the restricted scope of comparison, as the project primarily focuses on evaluating deep learning models without thoroughly comparing them with traditional machine learning methods or ensemble techniques. This narrow comparison hinders a comprehensive understanding of the relative strengths and weaknesses of different predictive models, potentially limiting the applicability of the findings in real-world scenarios. Moreover, the reliance on a single dataset, namely the Capture the Flag (CTF) dataset, poses limitations in capturing the diverse landscape of cyber threats encountered in practice. While the CTF dataset provides valuable insights, its scope may not fully encompass the breadth and complexity of cyberattacks observed across various sectors and industries. Furthermore, the project's evaluation metrics predominantly centre on the F-measure, overlooking other crucial metrics such as accuracy, precision, and recall. The emphasis on a single metric raises questions about the robustness and comprehensiveness of the evaluation process, as a holistic assessment of model performance requires consideration of multiple metrics to capture different aspects of prediction accuracy.

Dataset Description:

The dataset used for this report is the KDD train+ dataset which is a standard benchmark dataset used in construction of Network Intrusion Detection Systems (NIDS).It's a dataset

containing a collection of simulated network connection records. The dataset contains 43 features with each record representing a single network connection attempt.

- Basic Features:
    - duration: Duration of the network connection (likely in seconds).
    - protocol_type: Type of network protocol used (e.g., TCP, UDP, ICMP).
    - service: Requested network service (e.g., ftp, http, telnet).
- Flag Features:
    - flag: This likely refers to a collection of binary features based on the flags set in the network connection (e.g., URG, ACK, PSH, etc.). These flags control various aspects of the communication process.
- Byte Features:
    - src_bytes: Number of bytes sent from the source machine.
    - dst_bytes: Number of bytes sent to the destination machine.
- Connection Status Features:
    - land: Whether the source and destination IP addresses are identical (often an indicator of suspicious activity).
    - wrong_fragment: Whether the packet contains a malformed fragment header.
- Login Attempts:
    - num_failed_logins: Number of failed login attempts from the source machine.
    - logged_in: Whether a successful login occurred on the destination machine.
    - num_compromised: Number of compromised accounts on the destination machine (likely based on prior knowledge).
- User Privileged Actions:
    - root_shell: Whether the attacker attempted to gain root access on the destination machine.
    - su_attempted: Whether the attacker attempted to switch to a superuser account.
- File and Shell Activity:
    - num_file_creations: Number of file creation attempts from the source machine.
    - num_shells: Number of shell accounts opened on the destination machine.
    - num_access_files: Number of file access attempts from the source machine.
    - num_outbound_cmds: Number of outbound commands initiated from the source machine.
- User Origin Features:
    - is_host_login: Whether the source IP address is known to be a login host.
    - is_guest_login: Whether the source IP address is known to be a guest login.
- Connection Frequency Features:
    - count: Total number of connections to the destination machine from this source IP address.
    - srv_count: Number of connections to this specific service from this source IP address.
- Error Rate Features:
    - serror_rate: General error rate on the connection.
    - srv_serror_rate: Error rate on the service connection.

    - rerror_rate: Error rate reported by the destination machine.
    - srv_rerror_rate: Error rate reported by the destination machine specific to this service connection.
- Connection Rate Features:
    - same_srv_rate: Rate of connections to the same service from this source IP address.
    - diff_srv_rate: Rate of connections to different services from this source IP address.
    - srv_diff_host_rate: Rate of connections to this service from different source IP addresses.
- Destination Host Features:
    - dst_host_count: Total number of connections to this destination IP address.
    - dst_host_srv_count: Number of connections to this specific service on the destination machine.
    - Several other features prefixed with "dst_host_" likely capture various connection rates to/from the destination machine.
- Error Rate Features (Destination Host):
    - Similar to the error rate features above, but specific to connections involving the destination machine.
- Label and Difficulty:
    - label: This likely indicates the class label of the connection, such as "normal" or a specific attack type.
    - difficulty_level: This might represent the perceived difficulty of detecting the specific attack type.

Methodology:

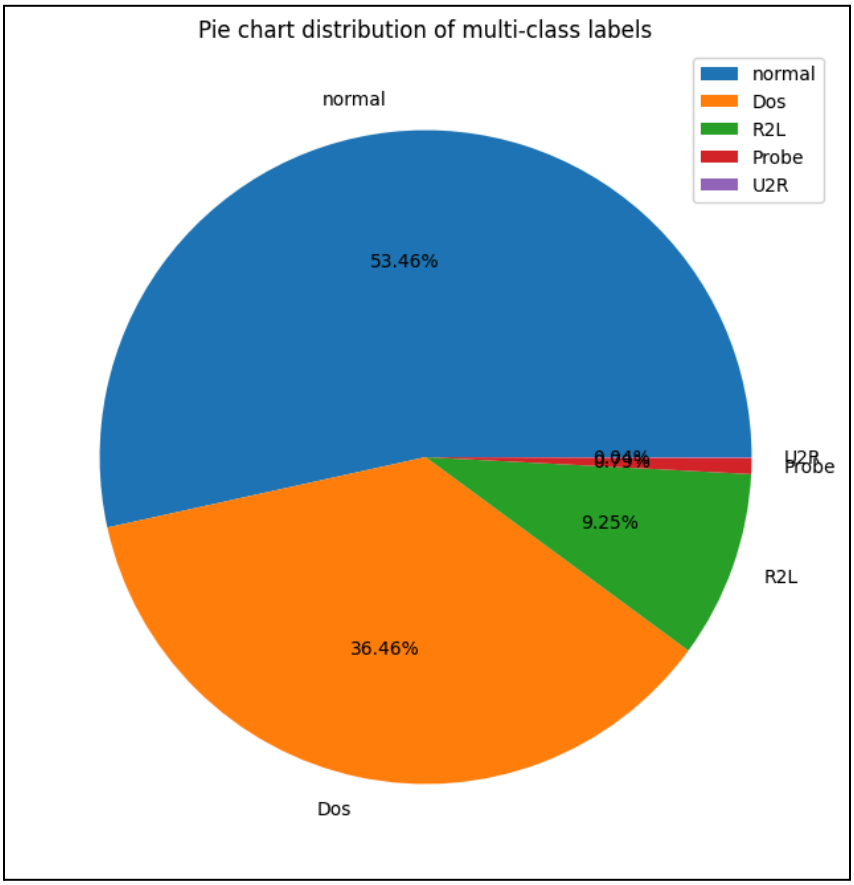
1. **Data Preprocessing**
    - Various libraries such as pandas, numpy, torch Sklearn and matplotlib have been imported to gain access to various performance metrics and neural network building techniques
    - Columns have been assigned appropriate names and irrelevant columns have been dropped
    - The dataset originally comprised 22 distinct attack labels representing various cybersecurity threats. To streamline analysis and enhance interpretability, systematic categorization was done. Through this process, the 22 labels were consolidated into 5 broader attack classes:
        - Denial of Service (DoS)
        - Unauthorized Access from a Remote Machine to Local System (R2L)
        - Probe
        - Unauthorized Access to Root (U2R)
        - Normal.

- This categorization not only simplified the complexity of the dataset and enabled a more focused examination of cybersecurity threats.

2. **Data Normalization**
    - Data normalization has been done to scale the features to a common range, to prevent features with greater scale having greater influence on the model.
    - StandardScaler from scikit-learn library has been used to normalize the data.

3. **Encoding**
    - In order to convert the categorical data to numerical features we have used one hot encoding
    - We have created new binary features for each unique category within a categorical feature. The new binary features represent the presence (1) or absence (0) of the original category. The following categorical features have been transformed
        - protocol_type (e.g., tcp, udp)
        - service (e.g., http, ftp, ssh, smtp)
        - flag (e.g., syn, ack)
    - Further label encoding has been done to assign a unique integer value to each distinct category in the “label” feature, this has been achieved using the LabelEncoder from SkLearn.
    - The encoder learns the mapping between categories and integers during the fitting stage and subsequently applies this mapping during the transformation stage.
    - In our implementation, a copy of the original data was created to avoid modifying the raw data. The "label" feature was then extracted and transformed into a separate DataFrame. The LabelEncoder object was fit on this DataFrame, effectively learning the unique categories and assigning corresponding integer values (0 for "Dos," 1 for "normal," and so on). The transformed labels were then added back to the original DataFrame with a new column named "intrusion.

- Here is a visualization of the final dataset



4. Model building

- Three deep learning models have been evaluated for our purpose. The models are
  - Multi-Layer Perceptron
  - Autoencoder Classifier
  - Long Short Term Memory (LSTM)
- MultiLayer Perceptron Classifer:

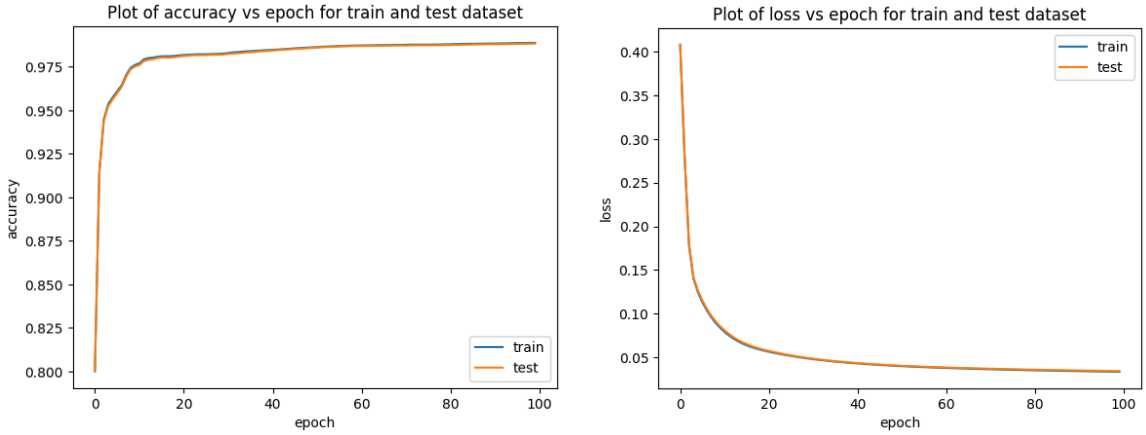
  Data has been split into training and testing with 75% for training and 25% for testing. Then it has been loaded to pytorch tensors.
  The model architecture is as follows:

  - 2 FC (fully connected layers)
  - Relu activation
  - Softmax output layer

  The model is trained using Binary Cross Entropy Loss as the loss function and the Adam optimizer. Training is performed over multiple epochs with batch

processing to improve convergence. The training and validation loss, as well as accuracy, are monitored during the training process.

The trained model is evaluated on the testing dataset to assess its generalization performance. Test loss and accuracy are computed, and plots of accuracy and loss versus epochs are generated to visualize the training process.



Further Receiver Operating Characteristic (ROC) curves are plotted to evaluate the model's performance across different classes.

Performance metrics such as Recall Score, F1 Score, and Precision Score are calculated to quantify the model's effectiveness in multi-class classification tasks.

- Auto encoder Classifier:

Similar to the MLP approach, the dataset is split into training and testing sets and converted into PyTorch tensors for processing.

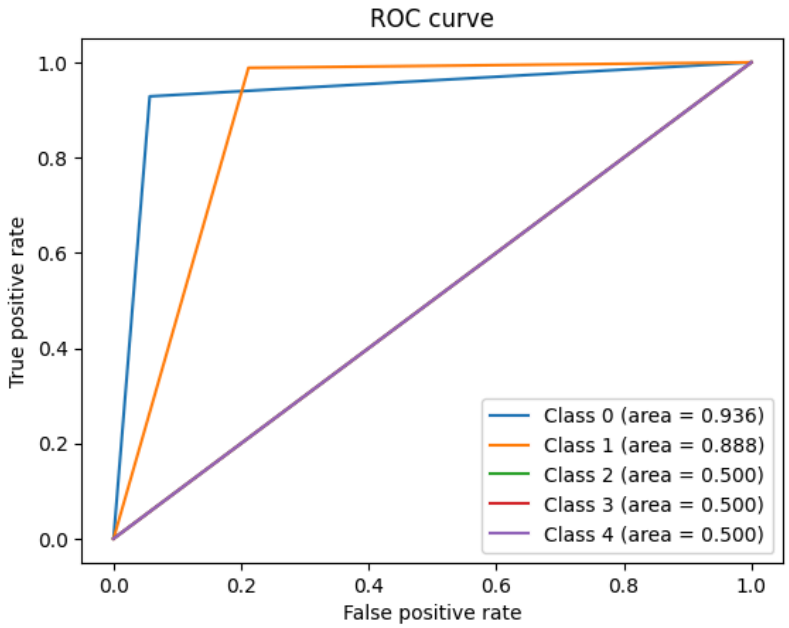
The Autoencoder Architecture consists of an encoder and a decoder, both comprising linear layers and ReLU activation functions. The decoder layer utilizes a Softmax activation function to reconstruct the input features.The encoder compresses the input data into a lower-dimensional representation The autoencoder is trained using the Mean Squared Error (MSE) loss function. This loss measures the difference between the reconstructed output from the decoder and the original input data.An Adam optimizer is used to update the autoencoder's weights during training. The optimizer adjusts the weights based on the calculated loss to minimize the reconstruction error.

Further, an AEClassifier class is defined. This classifier takes the encoded features obtained from the trained autoencoder as input and predicts the attack type. The classifier consists of a hidden layer with a Sigmoid activation function and an output layer with a Sigmoid activation function.The encoded features for the training data) are obtained by passing the training data through the trained

autoencoder. The AE classifier is trained using the MSE loss function. An Adam optimizer is used to update the AE classifier's weights during training.

The trained AE classifier is evaluated on the unseen test data. The encoded features for the test data (encoded_features_test) are obtained using the trained autoencoder.

The model's performance is measured using various metrics such as Loss,Accuracy, Recall, F1-Score and Precision. Additionally, ROC curves are generated to visualize the model's ability to discriminate between different attack types.



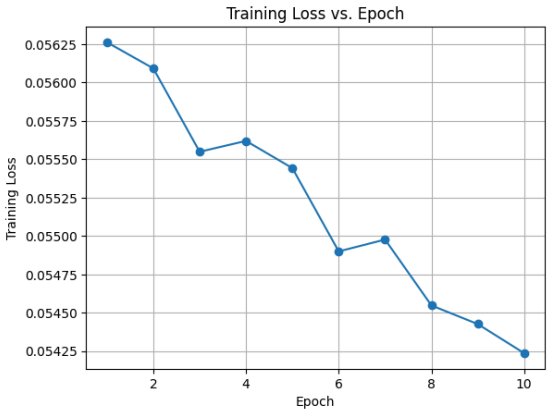
- LSTM Network:

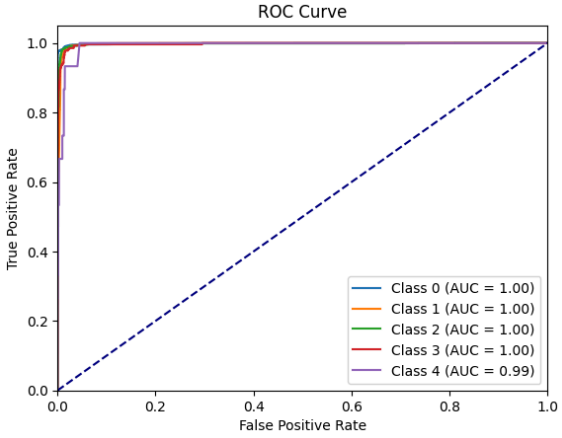
Data is prepared similar to the above described approaches. The model architecture consists of input layers, hidden layers and output layers. The size of the input layer is the same as the dimension of the training set used to train the model. The dimension of the hidden layer is 64 and the dimension of the output layer is the same as the number of classes i.e. 5.

Dataloader module has been used for efficient loading of training data in batches. The CrossEntropyLoss function is employed to measure the discrepancy between the predicted class probabilities and the actual attack labels. The Adam optimizer is utilized to update the model's weights during

training. It utilizes gradients calculated based on the loss function to adjust the weights, aiming to minimize the overall error and improve the model's ability to accurately classify different attack types.

Similar to the previous models metrics such as Loss,Accuracy, Recall, F1-Score and Precision are used to measure the performance and loss vs epochs and ROC curves are plotted for better visualization.





Results:

The results from each of the models can be seen in the table below

Model	Accuracy	Recall	F1-Score	Precision
MLP	0.98892	0.97136	0.97245	0.97342
AE	0.97613	0.857147	0.857147	0.857147
LSTM	0.97613	0.953272	0.924224	0.98324

Analysis of results:

- MLP: While having the highest accuracy, the MLP falls short in other metrics compared to the LSTM. This suggests the model might be memorizing training data patterns rather than generalizing well to unseen attack types.
- Autoencoder: The Autoencoder achieves similar accuracy to the LSTM but suffers in Recall, F1-Score, and Precision. This suggests the model might struggle with identifying

less frequent attack types, leading to an inflated accuracy but overlooking crucial minority classes.

- LSTM: The LSTM model demonstrates a balanced performance across all metrics. This indicates its ability to not only achieve high accuracy but also effectively identify true positives while minimizing false positives (incorrectly classifying normal traffic as attacks) and false negatives (missing actual attacks). The LSTM's ability to capture temporal dependencies within network traffic data might contribute to this balanced performance.

Conclusion:

Based on this analysis, the LSTM model emerges as the most suitable choice for multi-class network intrusion detection in this scenario. It achieves high accuracy while maintaining a balanced performance across Recall, F1-Score, and Precision. This suggests the model can effectively identify various attack types with minimal misclassifications.

References:

[1] Cyber-attack method and perpetrator prediction using machine learning algorithms. Singh, G., & Kushwaha, H. S. (2016, April).

[2] Deep Learning Methods for Cybersecurity and Intrusion Detection Systems . Aldawood, A., Uche Ogbugo, A., & Viriya Phongpairoch, J. (2017, January).

[3] Web attack detection using deep learning models . Ahmed, M., Mohamed Elhassan, A., & Thaier Nassar, Y. (2018, November).

[4] A deep learning framework for predicting cyber-attack rates . Moustafa, N., Creech, G., Slay, J., & Tabbane, S. (2019, April).

[5] CyberSecurity Attack Prediction: A Deep Learning Approach. Shonei, M., & Atiq, N. (2020, January).

[6] Conceptualisation of Cyberattack prediction with deep learning . Bhuvaneswari, V., & Sangeetha, K. (2020, February).

[7] Real Time Crime Detection Using Deep Learning Algorithm. Patil, P., Jadhav, S., Patil, V., & Thobbi, V. (2022, August).

[8] Detection of Malware Using Deep Learning . Wang, W., Tang, Y., Cheng X., Yu H., Peng J., & Huang, J. (2021, December)