# Department of Electrical and Information Engineering

## University of Ruhuna

## Mini Project Proposal

### Artificial Intelligence

### EC6301

### AI-Powered Network Intrusion Detection System


**Sanjitha W.M.K**

**Sandeepa M.K.P**

**Rathnayaka H.M.M.U**

**Medakanda M.W.M.W.E.C**


**Submission date**


Department of Electrical and Information Engineering
University of Ruhuna

# Contents

## 1. Problem Statement

(Maximum word limit: 150 words) Clearly define the problem or motivation behind your project. Explain and justify what issue, limitation, or real-world challenge you aim to address through image processing or computer vision with existing work.

How to use references using BibTeX files [1]. How to use references using BibTeX files [2]. How to use references using BibTeX files [3], [2].

Back to Table of Contents

## 2. Objectives

List the main goals of your mini project. Highlight what you intend to achieve or demonstrate at the end of the project. Give points-wise.

Back to Table of Contents

## 3. Literature Review

Recent developments in cyber-attacks and the growing intricacy of network traffic have driven researchers to investigate Artificial Intelligence-based approaches for Network Intrusion Detection Systems (NIDS). Conventional signature-based and rule-based systems struggle to identify new and evolving threats. Consequently, deep learning methods like Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM) networks have been extensively researched for intrusion detection. This section examines significant studies related to ANN- and LSTM-based NIDS, focusing on their methodologies, advantages, drawbacks, and highlighting the gaps that the proposed project seeks to address.

The ability of ANN-based systems to simulate non-linear correlations between attack classes and network properties is their main advantage. For well-known attack types including Denial of Service (DoS) and Probe attacks, ANN classifiers outperformed conventional rule-based systems in terms of detection accuracy and false-positive rates. When given enough labelled data to train, these systems showed good generalisation.

ANN-based intrusion detection systems do have some significant drawbacks, though. ANNs are unable to recognise the sequential patterns and temporal connections found in network traffic because they process each network record separately. Instead of happening as discrete incidents, many cyberattacks take place across a series of packets or sessions. As a result, ANN-only methods are not very good at identifying complex and slow-moving assaults, especially zero-day attacks that don't follow established statistical trends.]Study 1: ANN-Based Intrusion Detection Using Network Traffic Features [1] By approaching the issue as a multi-class classification challenge, a number of early studies examined the application of artificial neural networks for intrusion detection. In these methods, benchmark datasets like KDD Cup 99 and NSL-KDD were used to train feedforward ANN models. Protocol type, connection duration, packet counts, and byte statistics were among the network traffic parameters that were extracted and fed into the neural network.

The ability of ANN-based systems to simulate non-linear correlations between attack classes and network properties is their main advantage. For well-known attack types including Denial of Service (DoS) and Probe attacks, ANN classifiers outperformed conventional rule-based systems in terms of detection accuracy and false-positive rates. When given enough labelled data to train, these systems showed good generalisation.

ANN-based intrusion detection systems do have some significant drawbacks, though. ANNs are unable to recognise the sequential patterns and temporal connections found in network traffic because they process each network record separately. Instead of happening as discrete incidents, many cyberattacks take place across a series of packets or sessions. As a result, ANN-only methods are not very good at identifying complex and slow-moving assaults, especially zero-day attacks that don't follow established statistical trends.Study 1: ANN-Based Intrusion Detection Using Network Traffic Features [1] By approaching the issue as a multi-class classification challenge, a number of early studies examined the application of artificial neural networks for intrusion detection. In these methods,

benchmark datasets like KDD Cup 99 and NSL-KDD were used to train feedforward ANN models. Protocol type, connection duration, packet counts, and byte statistics were among the network traffic parameters that were extracted and fed into the neural network.

The ability of ANN-based systems to simulate non-linear correlations between attack classes and network properties is their main advantage. For well-known attack types including Denial of Service (DoS) and Probe attacks, ANN classifiers outperformed conventional rule-based systems in terms of detection accuracy and false-positive rates. When given enough labelled data to train, these systems showed good generalisation.

ANN-based intrusion detection systems do have some significant drawbacks, though. ANNs are unable to recognise the sequential patterns and temporal connections found in network traffic because they process each network record separately. Instead of happening as discrete incidents, many cyberattacks take place across a series of packets or sessions. As a result, ANN-only methods are not very good at identifying complex and slow-moving assaults, especially zero-day attacks that don't follow established statistical trends.

Study 2: LSTM-Based Network Intrusion Detection Systems[2] Researchers used Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) networks, in intrusion detection to overcome the drawbacks of static classifiers. Network traffic datasets as NSL-KDD, UNSW-NB15, and CICIDS2017, where traffic records were handled as time-series sequences, were subjected to LSTM models.

Learning long-term temporal dependencies in network behaviour is the main benefit of LSTM-based intrusion detection systems. LSTM models can identify intricate attack patterns that develop over time, such as multi-stage and covert attacks, by examining sequences of network events. When compared to conventional machine learning and ANN-based techniques, the experimental results from these investigations demonstrated higher detection rates and greater effectiveness in recognising unknown or changing threats.

LSTM-only systems have drawbacks in spite of these benefits. Scalability in real-time settings may be constrained by their usual requirements for sizable training datasets and more processing power. Additionally, LSTM models could underuse significant statistical correlations among network elements while putting a lot of emphasis on temporal patterns. Increased false-positive rates may arise from this, especially in dynamic networks where typical behaviour fluctuates regularly.

Study 3: Hybrid Deep Learning Models for Intrusion Detection[3] In order to enhance intrusion detection performance, more recent studies have investigated hybrid deep learning architectures that include several AI techniques. To take use of the temporal and non-temporal aspects of network traffic, LSTM networks were paired with feedforward neural networks or other classifiers in a number of experiments.

In these hybrid systems, the ANN component classifies data using features that have been retrieved, while the LSTM component learns sequential patterns from traffic flows. When compared to single-model techniques, the experimental findings showed better detection accuracy and robustness. Reducing false positives and enhancing classification performance across several assault types were two areas where hybrid models excelled.

However, a lot of hybrid techniques ignore real-time deployment constraints in favour of concentrating exclusively on improving accuracy. Excessive complexity introduced by certain models results in longer training times and less interpretability. Furthermore, a few of studies primarily assess their models on datasets of known attacks, offering scant insight into how well they defend against zero-day or previously undiscovered attacks.

Study 4: Anomaly-Based Intrusion Detection Using Deep Learning Rather than depending on pre-established attack characteristics, anomaly-based intrusion detection systems seek to detect departures from typical network behavior. In this area, deep learning models like LSTM-based anomaly detectors and autoencoders have been frequently deployed.

These systems discover anomalous patterns or reconstruction errors to identify intrusions after being trained on typical traffic data. Since anomaly-based techniques don't rely on labeled attack data, their primary advantage is their capacity to identify unknown and zero-day attacks. Several research that used deep learning-based anomaly detection claimed high detection rates for novel assaults.

However, because not all abnormalities indicate malicious behavior, anomaly-based systems have large false-positive rates. Furthermore, it can be challenging to identify particular attack types because many anomaly-based models lack categorization skills. This restricts their applicability in real-world network security scenarios where insights that can be put into practice are needed.

Research Gap and Proposed Improvement According to the reviewed research, LSTM-based models are excellent at capturing temporal dependencies in network traffic, whereas ANN-based systems are good at learning statistical correlations among network variables. Although hybrid approaches that combine these methods show better performance, they frequently have more complexity, less real-time applicability, and inadequate evaluation of zero-day assaults. The following are the main gaps in the current research:

- Limited ability to balance the modeling of statistical and temporal features
- High rates of false-positive results in anomaly-based and LSTM-only systems
- Not enough attention is paid to real-time and scalable intrusion detection
- Limited assessment of novel and changing assault patterns

These deficiencies are filled by the proposed AI-Powered Network Intrusion Detection System using ANN and LSTM, which combines ANN-based classification with LSTM-based temporal modelling. While the ANN component uses statistical traffic data for precise classification, the LSTM component records sequential attack behaviour. The goal of this hybrid strategy is to increase the system's capacity to identify known and unknown assaults in contemporary network environments, decrease false positives, and improve detection accuracy.

Back to Table of Contents

# 4. Methodology

## 4.1. System overview

The proposed system simulates network attacks and provides a means for detection using AI techniques, notably LSTM and ANN. This would involve data generation, preprocessing, feature extraction, model development, training, testing, and visualization. The solution design is such that the entire system would be implemented as a web application, with an interface showing up at two ends: one for an attacker and one for a user

## 4.2. Data Simulation

### 4.2.1 Attack Simulation

Python scripts generate synthetic traffic for selected attacks. Each attack type is parameterized with variables such as:

- Number of packets
- Flow duration
- Packet size distribution
- Attack frequency

The attacker interface allows the user to choose these parameters. The generated traffic is formatted as network feature vectors, which serve as input to the AI model..

### 4.2.2 Normal Traffic Simulation

Normal network behavior is simulated for comparison. Features such as packet rate, flow duration, and protocol usage are generated based on realistic traffic patterns. This ensures that the model learns to distinguish between normal and malicious patterns effectively.

## 4.3. Data Preprocessing

Preprocessing is required before feeding traffic to the AI model.

- Missing values should be handled
- Continuous features are scaled to bring them into a uniform range
- For LSTM, data is structured into time-sequential windows to capture temporal dependencies
- Encoding

### 4.4. Feature Extraction

Key features are extracted to represent each network flow effectively

- Flow Duration
- Total Forward and Backward Packets
- Average Packet Size
- Flow Bytes per Second
- SYN Flag Count
- Number of Failed Connections

These features are chosen based on their ability to highlight attack behaviors.Feature selection reduces dimensionality, improves training speed, and enhances model accuracy.

### 4.5. Model Development

#### 4.5.1 LSTM for Anomaly Detection

- Input: Sequential network traffic windows
- Purpose: Detect deviations from normal behavior
- Architecture: Multiple LSTM layers capture long-term dependencies in network traffic
- Output: Binary prediction (Normal / Abnormal) with probability score

#### 4.5.2 ANN for Attack Classification

- Input: Detected anomalous flows from LSTM
- Purpose: Classify the type of attack (e.g., DDoS, SYN flood, Port Scan)
- Architecture: Fully connected layers with ReLU activation, Softmax output
- Output: Predicted attack class with confidence percentage

#### 4.5.3 Training

- Loss Functions: Binary cross-entropy for LSTM, categorical cross-entropy for ANN
- Optimizer: Adam optimizer with learning rate tuning
- Metrics: Accuracy, Precision, Recall, F1-score

### 4.6. Model Evaluation

The trained model is evaluated using dataset traffic and simulated attack traffics.

- Confusion matrices for multi-class classification
- ROC curves use for anomaly detection
- Real-time testing with attacker interface

### 4.7. Web Application Implementation

Web application has 2 interfaces,

- **Attacker's interface** - Allow attackers to select attack type and other parameters
- **User's interface** - Displays the attack type with confidence score

The web application communicates with the Python AI backend via REST APIs, sending feature vectors and receiving predictions.

- ### 4.8. Tools and Technologies

- Python: Simulation, preprocessing, feature extraction
- TensorFlow / PyTorch: LSTM and ANN model development
- Pandas / NumPy / Scikit-learn: Data handling and preprocessing
- Flask / FastAPI: Web application backend
- HTML / CSS / JavaScript: Frontend interfaces
- Matplotlib / Seaborn / Plotly: Visualization of traffic patterns and model outputs

Back to Table of Contents

## 5. Dataset / Data Collection

The project utilizes the publicly available CyberTec IIoT Malware Dataset (CIMD-2024) from Kaggle, which contains approximately 1.5 million network traffic records in CSV format. It includes normal traffic and various attack types, such as DoS, Probe, and infiltration attacks, capturing both temporal and statistical features. Data preprocessing ensures quality by handling missing values, normalizing features, and encoding categorical data. Using this benchmark dataset guarantees representative, diverse, and ethically sourced data, avoiding personal or sensitive information. A subset may be sampled for efficient AI model training while maintaining class diversity, providing a reliable foundation for the proposed AI-powered intrusion detection system.

Back to Table of Contents

## 6. Expected Outcomes

- Accurate detection and classification of network attacks such as DDoS and SYN flooding using LSTM and ANN.
- Real-time prediction with confidence percentages displayed in the user interface.
- Interactive web application allowing attacker simulation and user monitoring.
- Reduced false-positive rates compared to traditional rule-based systems.
- Demonstration of AI-driven adaptive intrusion detection applicable to modern network environments.
- Visualization of traffic patterns and attack behaviors for better understanding of network anomalies.

Back to Table of Contents

## 7. Percentage of AI Involvement

The system achieves 95% AI content. LSTM captures temporal patterns in network traffic for anomaly detection, while ANN classifies attacks like DDoS and SYN flooding. Preprossessing and feature extraction handle scaling, encoding, and key traffic metrics. Python with TensorFlow/PyTorch, Pandas, and Scikit-learn is used for simulation, model training, and prediction.

Back to Table of Contents

## References

[1] R. C. Gonzalez and R. E. Woods, "Digital image processing," *Prentice Hall*, 2008.

[2] S. Sanei and J. A. Chambers, *EEG signal processing*. John Wiley & Sons, 2013.

[3] L. Hu and Z. Zhang, *EEG signal processing and feature extraction*. Springer, 2019.

A minimum of ten (15) references must be included in the References section.