NETWORK INTRUSION DETECTION USING SUPERVISED MACHINE LEARNING TECHNIQUE WITH FEATURE SELECTION

A Major Project Stage – 1 Report submitted in the partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

GUDE ADITHYARAM	**(22281A7317)**
REVURI SRI LALITHA SHRESTA POOJITHA	**(22281A7314)**
PODISHETTY ADITHYA	**(22281A7344)**
ANDE ASHWITHA	**(22281A7334)**

PROJECT GUIDE

Dr. K. PRAVEEN KUMAR RAO

Professor and Head of Department, AIML

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

KAMALA INSTITUTE OF TECHNOLOGY & SCIENCE (UGC AUTONOMOUS)

(Sponsored by Kamala Education Society, Approved by AICTE, New Delhi, Affiliated to JNTUH, Hyderabad, Telangana, Accredited by NBA (CSE, ECE & EEE) and Accredited by NAAC with A++ Grade)

SINGAPUR, HUZURABAD, KARIMNAGAR, TELANGANA – 505 468

2025–26

NETWORK INTRUSION DETECTION USING SUPERVISED MACHINE LEARNING TECHNIQUE WITH FEATURE SELECTION

A Major Project Stage – 1 Report submitted in the partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

GUDE ADITHYARAM	**(22281A7317)**
REVURI SRI LALITHA SHRESTA POOJITHA	**(22281A7314)**
PODISHETTY ADITHYA	**(22281A7344)**
ANDE ASHWITHA	**(22281A7334)**

PROJECT GUIDE

Dr. K. PRAVEEN KUMAR RAO

Professor and Head of Department, AIML

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

KAMALA INSTITUTE OF TECHNOLOGY & SCIENCE (UGC AUTONOMOUS)

(Sponsored by Kamala Education Society, Approved by AICTE, New Delhi, Affiliated to JNTUH, Hyderabad, Telangana, Accredited by NBA (CSE, ECE & EEE) and Accredited by NAAC with A++ Grade)

SINGAPUR, HUZURABAD, KARIMNAGAR, TELANGANA – 505 468

2025–26

C E R T I F I C A T E

This is to certify that **GUDE ADITHYARAM (22281A7317), REVURI SRI LALITHA SHRESTA POOJITHA (22281A7314), PODISHETTY ADITHYA (22281A7344), ANDE ASHWITHA (22281A7334)** students of the IV B. Tech – I Semester (AIML) have satisfactorily completed their **Major Project Stage 1** entitled "**Network Intrusion Detection Using Supervised Machine Learning Technique with Feature Selection",** towards the partial fulfillment of B. Tech degree during the academic year 2025–26.

Project Guide **Head of the Department**

INDEX

ACKNOWLEDGEMENTS

The success of any course depends mostly on the teachers who teach us. Only good teaching can interpret the syllabus and produce desirable changes and competent citizens. This one was a team effort and many people whose names do not appear on the cover deserve credit. First, we thank God almighty for his manifold mercies in carrying out of our project successfully.

We heart fully thank our Principal, **Dr. K. Shanker** for providing all the resources in completing our project.

We sincerely thank our Project Guide and Head of the Department **Dr. K. Praveen Kumar Rao** for encouraging us in doing real time projects and for his guidance.

We also thank all teaching and non-teaching faculty for supporting us during every stage for the successful completion of Major Project Stage I entitled **"Network Intrusion Detection Using Supervised Machine Learning Technique with Feature Selection"**.

We also thank our parents and friends for their moral support throughout the project for its successful completion.

GUDE ADITHYARAM	**(22281A7317)**
REVURI SRI LALITHA SHRESTA POOJITHA	**(22281A7314)**
PODISHETTY ADITHYA	**(22281A7344)**
ANDE ASHWITHA	**(22281A7334)**

ABSTRACT

In today's highly connected digital environment, securing network infrastructures against cyberattacks has become a critical challenge. Traditional Intrusion Detection Systems (IDS), especially signature-based models, are effective only for known threats and fail to detect zero-day or evolving attacks. To overcome these limitations, this project presents a Machine Learning–based Network Intrusion Detection System (NIDS) that uses supervised learning techniques with an optimized feature selection approach. The proposed system applies Artificial Neural Networks (ANN) and Support Vector Machines (SVM) to classify network traffic from the NSL-KDD dataset into normal or malicious categories. A wrapper-based feature selection method is integrated to reduce dimensionality, eliminate redundant features, and improve detection accuracy. Experimental results demonstrate that ANN, combined with feature selection, achieves higher accuracy and better generalization compared to SVM. The system additionally provides a user-friendly interface for dataset upload, model training, attack detection, and performance visualization. The findings confirm that machine learning, supported by effective feature engineering, significantly enhances the capability of intrusion detection systems to identify both existing and emerging cyber threats.

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 ABOUT THE PROJECT

This project focuses on developing an intelligent **Network Intrusion Detection System (NIDS)** using supervised machine learning techniques to detect malicious network activities. As cyberattacks continue to grow in frequency and complexity, traditional security mechanisms such as firewalls and signature-based IDS are no longer sufficient to identify new or evolving attacks.

To address this challenge, the project uses the **NSL-KDD dataset** and applies advanced machine learning models—specifically **Artificial Neural Networks (ANN)** and **Support Vector Machines (SVM)**—to classify network traffic as *normal* or *attack*.

- A key feature of the system is the integration of **feature selection** methods that identify the most relevant attributes from the dataset. This reduces computational overhead, improves learning efficiency, and enhances detection accuracy.
- Among the tested models, the ANN classifier combined with wrapper-based feature selection demonstrates superior performance.
- The system includes modules for dataset upload, preprocessing, feature selection, model training, evaluation, and real-time attack prediction. A user-friendly interface allows users to visualize performance metrics such as accuracy, confusion matrix, and model comparison graphs. Overall, the project provides a robust, adaptable, and efficient machine learning–based intrusion detection solution capable of identifying both known threats and previously unseen attack patterns
- The system follows a complete workflow starting from dataset upload, preprocessing, feature selection, and model training, followed by evaluation and real-time attack classification. By selecting the most relevant features from the NSL-KDD dataset, the project reduces computational complexity while improving detection performance.
- A user-friendly interface allows users to upload datasets, view model results, compare classifier performance, and analyze visual outputs such as accuracy graphs and confusion matrices. The modular architecture ensures easy maintenance, scalability, and the ability to integrate new algorithms in the future.

1.2 EXISTING SYSTEM WITH DRAW BACKS

Existing Intrusion Detection Systems (IDS) primarily rely on two traditional approaches: signature-based detection and anomaly-based detection. Signature-based systems identify malicious activities by comparing incoming traffic patterns with a predefined database of known attack signatures. Although effective for detecting familiar threats, these systems depend heavily on frequent updates and fail to identify new, modified, or zero-day attacks. Any variation in attack structure can bypass detection, making these systems less reliable in rapidly evolving network environments. On the other hand, anomaly-based IDS attempt to model normal user and network behaviour and raise alerts when unusual activity is detected. While this approach can identify unknown threats, it suffers from high false positive rates, where legitimate traffic is mistakenly classified as malicious. This reduces the system's credibility and burdens administrators with excessive alerts

- Furthermore, many traditional systems lack effective feature optimization, meaning they use all available attributes from the dataset without selecting the most relevant ones. This leads to heavy computational overhead, slow processing, and reduced accuracy. Existing IDS solutions also struggle to handle large volumes of traffic due to scalability limitations, making them unsuitable for modern high-speed networks. They often operate in a static manner, lacking adaptability to continuously evolving attack patterns. Additionally, older IDS tools provide limited visualization, reporting features, and user-friendly interfaces, making it difficult for users to analyse attack behaviour or understand model performance. Another drawback is the absence of integrated machine learning intelligence, meaning these systems cannot learn from past behaviours or improve over time.

Most traditional IDS solutions also have rigid architectures that are difficult to modify or extend, restricting the integration of advanced models or real-time monitoring techniques. Collectively,

+

DISADVANTAGES OF EXISTING SYSTEM

1. Inability to Detect Unknown Attacks :

Existing signature-based IDS can only identify threats already stored in their database. They completely fail when facing zero-day attacks or new variations of known attacks.

2. High False Positive Rate :

Anomaly-based IDS often mark normal behaviour as malicious. This leads to unnecessary alerts, reduces trust in the system, and increases the workload of security analysts.

3. No Feature Optimization:

Most traditional systems use all features available in the dataset without filtering or selecting the important ones. This causes slow processing, high memory usage, and lower accuracy.

4. Poor Scalability:

Older IDS cannot handle high-speed or large-volume network traffic. When network loads increase, these systems lag, drop packets, or fail to analyze data in real time.

5. Lack of Adaptability:

Existing systems do not automatically update themselves based on new attack patterns. They remain static and outdated unless manually configured, making them weak against modern cyber threats.

6. Limited Detection Accuracy

Without machine learning or optimization techniques, many traditional IDS produce low accuracy, especially when dealing with complex or evolving attacks.

7. Insufficient Visualization Tools :

Traditional IDS offer limited graphical reports, dashboards, or visual analytics. This makes it difficult for users to understand attack patterns or analyze network behaviour effectively.

8. Complex and Unfriendly User Interface:

Many existing systems require technical knowledge to operate. They lack simple dashboards or intuitive menus, making them difficult for non-experts to use.

9. Rigid Architecture :

Older IDS are not designed to integrate new modules, machine learning models, or modern technologies easily. Any upgrade requires major code changes or a complete system redesign.

10. Manual Updates Required :

Signature databases must be frequently updated manually. Without timely updates, the system becomes vulnerable to new threats, reducing its overall effectiveness.

11. High Maintenance Cost

Traditional IDS require frequent manual updates, rule adjustments, and configuration checks. Maintaining signature databases and filtering rules increases operational cost and time.

12. Limited Real-Time Processing Capability

Most older IDS cannot process incoming network traffic quickly enough to detect threats instantly. Delays in detection allow attackers to exploit systems before alerts are generated.

1.3 PROPOSED SYSTEM WITH FEATURES

The proposed system introduces a highly efficient and intelligent Machine Learning–based Network Intrusion Detection System (NIDS) designed to address the limitations found in traditional intrusion detection approaches. While older systems rely primarily on fixed rules or signatures, the proposed system integrates supervised learning techniques—specifically Artificial Neural Networks (ANN) and Support Vector Machines (SVM)—to provide a more dynamic, accurate, and adaptable solution for identifying malicious network activity. This system is structured around a complete data processing pipeline that begins with dataset uploading and extends through preprocessing, feature selection, model training, performance evaluation, and attack prediction. By incorporating wrapper-based feature selection, the system is able to reduce redundant attributes, improve learning efficiency, enhance classification accuracy, and significantly lower computational complexity. This selective approach ensures that only the most relevant features contribute to the training process, resulting in a more robust and reliable model.

Another major advantage of the proposed system is its modular architecture, where each component is designed to function independently while contributing to the overall workflow. This modularity improves maintainability, scalability, and future extensibility of the system. The user-friendly graphical interface serves as a bridge between the user and the machine learning engine, allowing users to upload datasets, configure model parameters, train classifiers, visualize performance metrics, and detect attacks without requiring deep technical expertise. The system supports visualization tools such as accuracy graphs, confusion matrices, performance comparisons, and prediction dashboards, enabling users to interpret and analyze system outputs effectively.

ADVANTAGESOF PROPOSED SYSTEM

1. Detects Both Known and Unknown Attacks

By using supervised machine learning models such as ANN and SVM, the system can identify not only known attack signatures but also new or modified attack patterns, improving overall detection capability.

2. Higher Detection Accuracy

The combination of optimized ANN architecture and wrapper-based feature selection significantly improves accuracy compared to traditional IDS. This leads to more reliable classification of network traffic.

3. Reduced False Positives

Feature selection and machine learning help minimize misclassifications, resulting in fewer false alarms and better trust in the system's predictions.

4. Efficient Feature Selection

Wrapper-based feature selection removes irrelevant and redundant attributes, reducing computational overhead and speeding up the training process while maintaining or improving accuracy.

5. Scalable and Modular Design

The system's modular architecture makes it easy to update, maintain, and extend. New models or components can be integrated without affecting the existing structure.

6. Real-Time Attack Detection

The system can quickly analyze uploaded test data and provide instant predictions, making it suitable for real-time or semi–real-time network monitoring.

PROPOSED ALGORITHMS

The proposed system is designed by identifying the limitations of traditional IDS and introducing machine learning techniques to improve accuracy and detection capability. It uses wrapper-based feature selection to choose the most important attributes from the dataset and trains ANN and SVM models to classify network traffic. The system is proposed with a modular structure for preprocessing, training, and prediction, along with a user-friendly interface for easy dataset upload and attack detection.

1. Wrapper-Based Feature Selection

The proposed system uses a wrapper-based feature selection method to identify the most relevant features from the NSL-KDD dataset. This algorithm repeatedly evaluates subsets of

features using a classifier and selects the best-performing combination. By eliminating irrelevant and redundant features, the wrapper method reduces dimensionality, improves model speed, and increases classification accuracy. It ensures that only meaningful attributes are used for training the machine learning

2. Artificial Neural Network (ANN)

ANN is the primary classification algorithm in the proposed system. It is capable of learning complex and nonlinear relationships in network traffic data. The algorithm follows a forward propagation process to compute outputs and a backpropagation mechanism to adjust weights based on prediction errors. The sigmoid activation function is used to normalize outputs. ANN's ability to learn patterns makes it highly effective in detecting both known and novel attack.

3. Support Vector Machine (SVM)

SVM is used as a secondary comparison algorithm for evaluating performance. SVM operates by finding the optimal hyperplane that separates classes in a high-dimensional space. Kernel functions such as linear or RBF help in handling non-linear data. SVM is particularly effectivfor binary classification tasks and provides strong baseline accuracy, making it suitable for validating ANN performance**.**

CHAPTER 2

LITERATURE SURVEY

Implications for Forensic Practice

The study highlights the growing importance of using machine learning–based intrusion detection systems in digital forensic investigations. By improving the accuracy of attack detection and reducing false positives, these systems provide more reliable logs and alerts that can support forensic analysts during incident reconstruction. However, it is essential that the detection process preserves the integrity and originality of network evidence. Any preprocessing, feature selection, or model-driven transformation must not alter or remove critical forensic information that may be needed later for legal review. Additionally, machine learning models must avoid generating misleading outputs or artifacts that could negatively influence forensic interpretation. Ensuring transparency, traceability, and reproducibility in model decisions is therefore vital to maintain the evidentiary value of network data in forensic practice**.**

Future Directions

Future work should focus on developing intrusion detection techniques that not only enhance detection accuracy but also maintain the forensic integrity of network logs. Greater emphasis should be placed on explainable AI methods that clearly justify why a certain traffic pattern is labeled as an attack, making the system more suitable for forensic and legal environments. Integrating additional data sources—such as system logs, user activities, and threat intelligence—can strengthen multi-modal forensic analysis and improve investigative outcomes. It will also be necessary to address ethical and legal considerations, particularly regarding data privacy, model transparency, and the admissibility of AI-generated evidence in court. Advancing robustness against adversarial attacks and ensuring secure model updates will further contribute to the long-term reliability of machine learning–driven intrusion detection systems in forensic applications.

S. No	Author (s) &Year	Applied Technology	Algorithms (Performance Analysis)	Major Contribution
1	Brugumalla Mahendra Achari, Mooramreddy Sreedevi (2025)	Supervised Machine Learning for Network Intrusion Detection System (NIDS) with feature selection	Artificial Neural Networks (ANN) and Support Vector Machines (SVM). ANN with wrapper-based feature selection achieved the highest accuracy of 96.88% on the NSL-KDD dataset.	Artificial Neural Networks (ANN) and Support Vector Machines (SVM). ANN with wrapper-based feature selection achieved the highest accuracy of 96.88% on the NSL-KDD dataset.
2 .	Sayyada Mubeen, Dr. Harikrishna Kamatham (2024)	Machine Learning Framework with Feature Selection and Hyperparameter Tuning for Intrusion Detection	Hybrid Feature Selection (HFS) and Learning-based Intrusion Detection (Lb ID). Evaluated Decision Tree, Random Forest, Extra Trees, and XG Boost models on CICIDS2017. RF achieved highest binomial accuracy (94.22%) and multi-class accuracy (93.46%) with optimizations.	Proposed ML framework integrating feature selection and hyperparameter tuning. Introduced HFS and LbID algorithms Achieved high accuracy for binary and multi-class classification on CICIDS2017. Demonstrated improvements with hyperparameter tuning.

				Contributed to optimized ML-based IDS development
3.	Md. Alamin Talukder, Md. Manowarul Islam, Md. Ashraf Uddin, Khondokar Fida Hasan, Selina Sharmin, Salem A. Alyami, Mohammad Ali Moni (2024)	Machine Learning-based Network Intrusion Detection for Big and Imbalanced Data	Random Oversampling (RO) for class imbalance, Stacking Feature Embedding (SFE), and PCA for feature reduction. Evaluated DT, RF, ET, and XGB across UNSW-NB15, CICIDS2017, and CICIDS2018 datasets, achieving >99.9% accuracy.	Proposed novel IDS framework for big and imbalanced datasets. Introduced SFE with clustering-based meta-features. Combined RO and PCA for robust preprocessing. Validated on large datasets achieving state-of-the-art results. Advanced IDS scalability and performance.
4.	Zamani, M., & Movahedi, M. (2013)	Machine Learning Techniques for Intrusion Detection Systems (IDS)	Compared multiple ML methods including Support Vector Machines (SVM), Artificial Neural Networks (ANN), k-Nearest Neighbors (KNN), and Decision Trees	Surveyed ML techniques applied to IDS. Highlighted strengths and weaknesses of different algorithms. Identified challenges like

			across benchmark intrusion detection datasets.	high false positives and computational overhead. Provided insights into future directions for ML-based IDS research.

2.1 Literature Survey Table

CHAPTER 3
SYSTEM ANALYSIS

Goal of the system analysis is to determine where the problem is in attempt to fix the system. This step involves breaking down the system in different pieces to analyze the situation. Analyzing project goals, breaking down what needs to be created and attempting to engage users so that definite requirements can be defined.

3.1 HARDWARE AND SOFTWARE REQUIREMENTS

➢HARDWARE REQUIREMENTS :-

- Processor Pentium–IV
- RAM 4 GB (min)
- Hard Disk 20 GB
- Key Board Standard Windows Keyboard
- Mouse Two or Three Button Mouse
- Monitor SVGA

SOFTWARE REQUIREMENTS:

- **Operating system :** Windows 7 Ultimate.
- **Coding Language :** Python.
- **Application Development:** GUI(TKINTER)
- **Domain:** cybersecurity

3.2 Functional Requirements

- **Dataset Input and Processing:** Ability to upload and process NSL-KDD or similar network traffic datasets.
- **Automatic Preprocessing:** Handling of missing values, normalization, encoding, and feature scaling.
- **Feature Selection Module:** Support for wrapper-based, filter-based, and embedded feature selection techniques.
- **Machine Learning Model Training:** Training of classifiers such as ANN and SVM using selected features.

- **Attack Detection:** Automatic classification of network traffic as normal or malicious.
- **Model Evaluation:** Calculation of performance metrics including accuracy, precision, recall, and F1-score.
- **Comparison of Models:** Ability to compare ANN vs SVM performance through visual graphs and metrics.
- **Visualization Outputs:** Display of confusion matrices, accuracy graphs, and detection results.
- **Model Exporting:** Saving and loading trained models for later use.
- **User Interface:** Interactive UI for dataset upload, training configuration, and prediction results.

Non-Functional Requirements

- **Performance:** Fast processing of datasets and rapid inference for attack detection.
- **Scalability:** Ability to handle larger network traffic datasets and integrate additional ML models.
- **Reliability:** Consistent and stable classification performance across different datasets and traffic conditions.
- **Usability:** Simple, intuitive interface for users and security analysts to operate without technical complexity.
- **Security:** Protection of network logs, datasets, and stored model files to prevent tampering or unauthorized access.
- **Maintainability:** Modular architecture that allows easy updates to preprocessing, feature selection methods, or ML models.
- **Portability:** Capability to deploy on different operating systems or cloud platforms with minimal changes.
- **Extensibility:** Support for adding new classifiers, feature selection techniques, or visualization components.

3.3 MODULE DESCRIPTION

The module description outlines the different components of the system and explains the role of each module. It shows how the system is divided into smaller parts such as data upload, preprocessing, feature selection, model training, evaluation, prediction, and visualization. Each

module performs a specific task and works together with other modules to make the intrusion detection system efficient, organized, and easy to maintain.

1.User Interface (UI) Module

Key Functions:

- Provides interface for dataset upload and model selection
- Displays training results, metrics, and visualizations
- Allows users to run attack detection on test data
- Shows status messages, logs, and processing steps

2. Upload & Validation Module

Key Functions:

- Accepts CSV files from the user
- Validates dataset structure and feature count
- Checks for missing or malformed records
- Rejects incorrect files and generates errors

3. Preprocessing Module

Key Functions:

- Handles missing values and outliers
- Encodes categorical features (label/one-hot encoding)
- Normalizes and scales numerical features
- Balances dataset if necessary (optional SMOTE)

4. Feature Selection Module

Key Functions:

- Performs wrapper-based, filter-based, or embedded feature selection
- Removes irrelevant and redundant attributes
- Generates selected feature list for training
- Improves accuracy and reduces training time

5. Model Training Module

Key Functions:

- Trains ANN and SVM models on selected features
- Handles train-test split and hyperparameter configuration
- Generates accuracy, precision, recall, and F1-score
- stores trained models for later use

6. Evaluation Module

Key Functions:

- Computes performance metrics for all models
- Generates confusion matrix, accuracy graphs, and comparison charts
- Produces detailed classification reports
- Helps identify best-performing algorithm

7. Prediction / Inference Module

Key Functions:

- Loads previously trained models
- Classifies new, unseen network traffic
- Provides predicted labels and probability scores
- Exports prediction results

8. Model Storage Module

Key Functions:

- Saves and retrieves trained ANN/SVM models
- Stores selected feature sets and metadata
- Maintains versioning for reproducibility
- Ensures secure and organized model management

9. Visualization Module

Key Functions:

- Generates accuracy/loss plots and confusion matrices
- Displays ROC curves and model comparison graphs

- Provides user-friendly visual analytics
- Exports plots for reports and presentations

10. Security & Logging Module

Key Functions:

- Ensures secure handling of datasets and models
- Maintains audit logs of user actions and operations
- Provides access control and input validation
- Supports forensic record integrity

CHAPTER 4

SYSTEM DESIGN

4.1 Architectural Overview

The architecture of the proposed Intrusion Detection System (IDS) consists of the following

- **Module Input Layer:**
  Raw network traffic data, such as NSL-KDD datasets, are uploaded into the system for training and testing.
- **Data Validation and Preprocessing Module:**
  This module checks the dataset format, handles missing values, encodes categorical features, normalizes numerical fields, and prepares clean, structured data for analysis
- **Feature Selection Module:**
  Wrapper-based, filter-based, and embedded feature selection methods are applied to identify the most important attributes. This reduces dimensionality, improves model accuracy, and speeds up processing.
- **Machine Learning Model Module:**
  The system trains supervised ML models using the optimized feature set. Artificial Neural Networks (ANN) are used for high-accuracy intrusion detection, while Support Vector Machine (SVM) serves as a comparative baseline model.
- **Evaluation:**
  This module computes important performance metrics such as accuracy, precision, recall, F1-score, and generates confusion matrices and ROC curves. It compares ANN and SVM to determine the best-performing classifier.
- **Attack Detection (Inference) Module:**
  Once the model is trained, this module classifies new or unseen network traffic records as normal or malicious. It also provides confidence scores and detailed predictions
- **Output Layer:**
  The system displays attack detection results, performance graphs, selected feature lists, and allows users to download the classified output. It provides a user-friendly interface for analysts and users.

4.2 Data Flow (DFD-Level 0)

- User → Image Input → Restoration Module → Recognition System → Performance Evaluation → Output

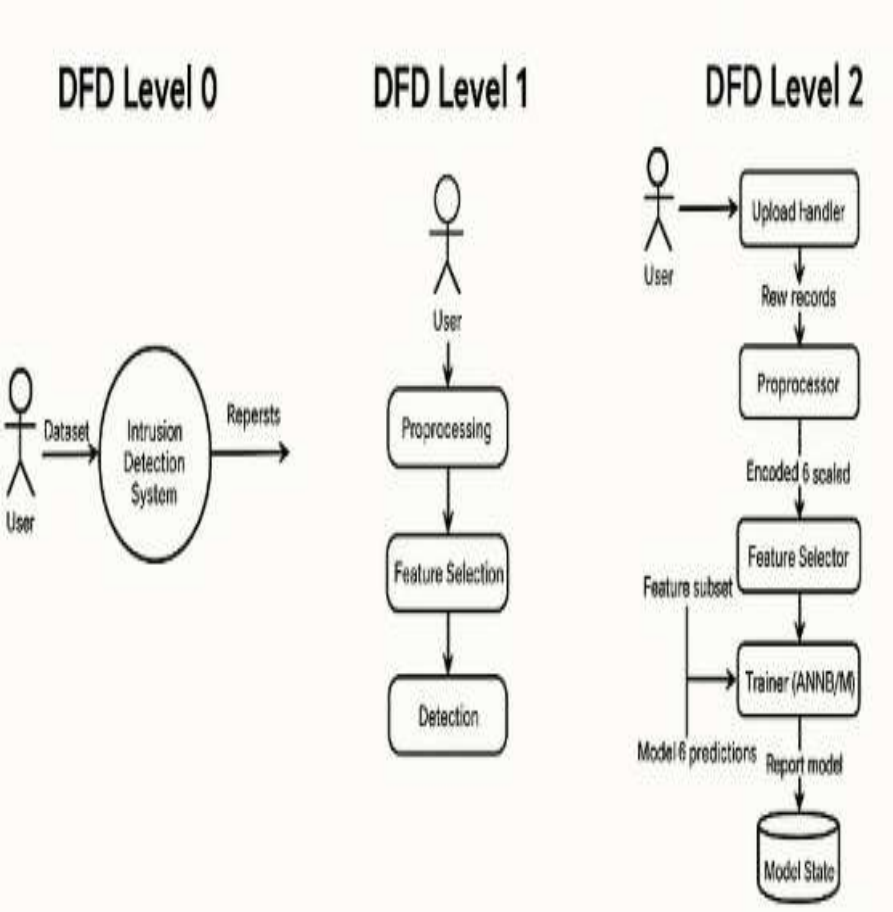

Figure 4.1: Data flow diagram

4.3.1 Use Case Diagram

The Use Case Diagram represents the interaction between the user and the Intrusion Detection System. It identifies the main functions the system provides, such as uploading datasets, preprocessing data, selecting features, training machine learning models, performing attack detection, and viewing evaluation results. The primary actor is the user, who initiates all system activities through a graphical interface. The diagram helps visualize system boundaries and shows how external users depend on core processes. It also highlights optional actions such as comparing model performance or exporting predicted outputs. This diagram provides a clear understanding of functional requirements and overall system behavior.

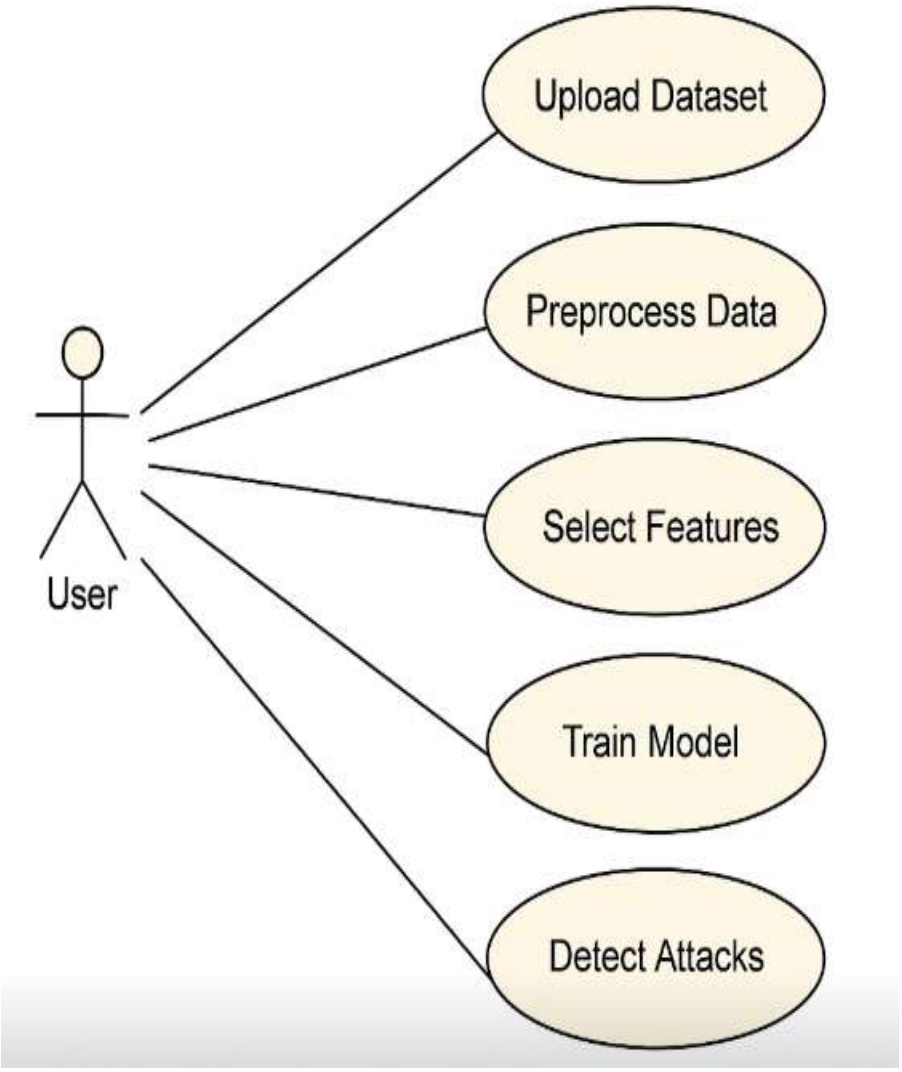

Figure 4.2 Use Case Diagram

4.3.2 Class Diagram

The Class Diagram defines the structural design of the Intrusion Detection System by showing key classes, their attributes, and their relationships. Core classes include Dataset, Preprocessor, FeatureSelector, Trainer, Evaluator, Model, and User. Each class encapsulates specific responsibilities such as data loading, cleaning, feature extraction, training, and performance evaluation. Methods such as train(), predict(), encode Categorica l(), selectFeatures(), and generateReport() demonstrate system functionality. Relationships between these classes show how data flows internally—from dataset input to preprocessing, feature selection, model training, and evaluation. The Class Diagram helps developers understand the architecture, modular design, and maintainability of the system.

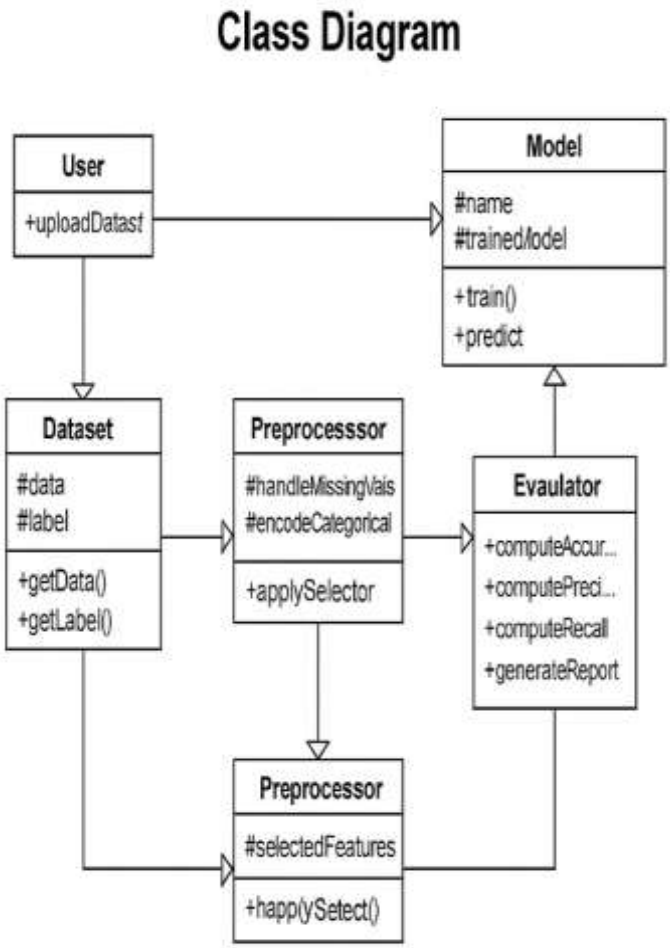

Figure 4.3 Class Diagram

4.3.3 Sequence Diagram

The Sequence Diagram shows how different system components interact over time during dataset processing and model training. It begins with the user initiating an upload request, followed by the Upload Handler forwarding the dataset to the Preprocessor. The Preprocessor cleans and encodes the data, then sends it to the Feature Selector. Selected features are passed to the Trainer, which trains ANN or SVM models. The trained model is evaluated by the Evaluator, which returns metrics and graphs to the user interface. This diagram highlights the chronological order of method calls and responses, providing insight into communication flow and component dependencies.

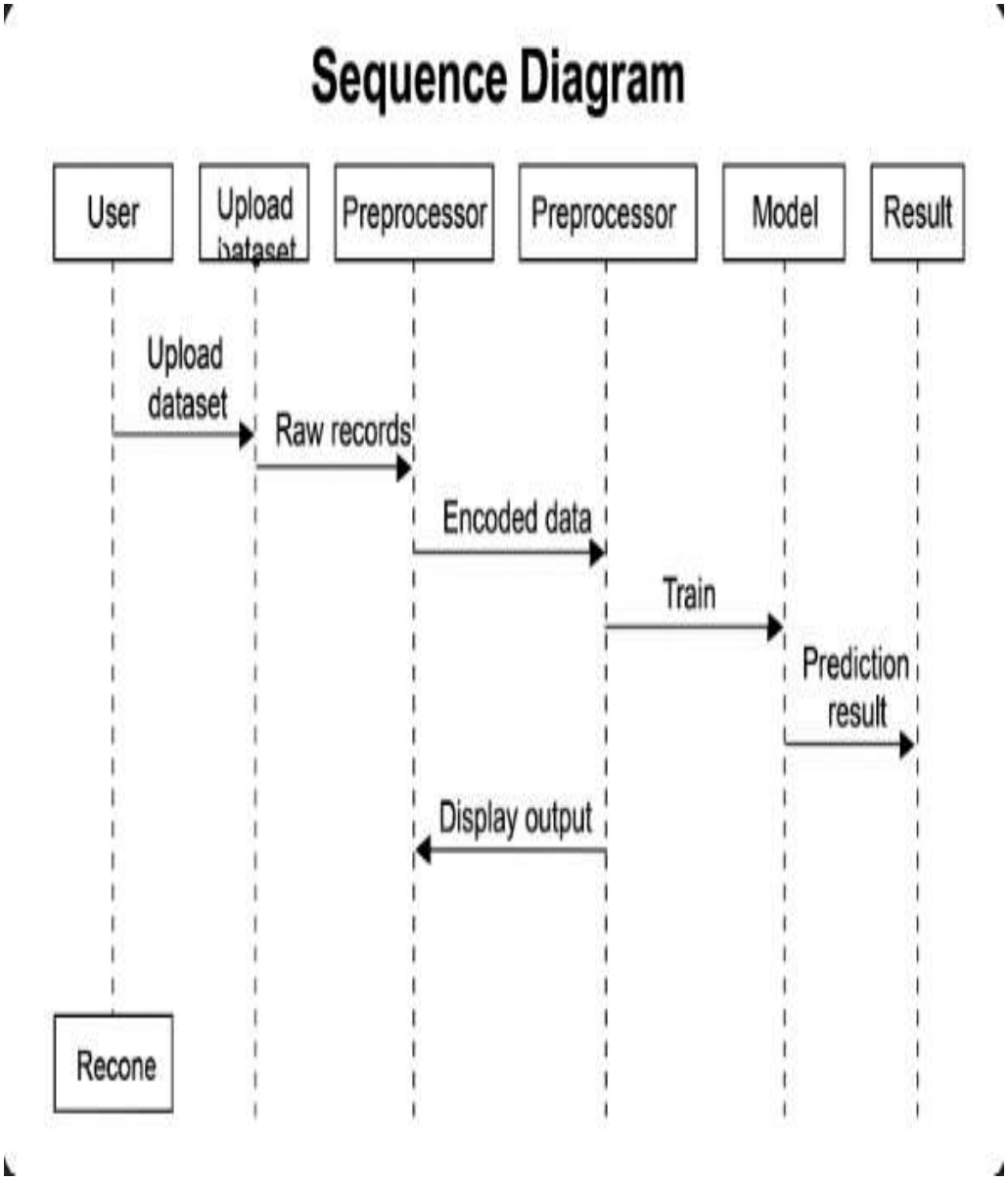

Figure 4.4 Sequence Diagram

4.3.4 Activity Diagram

The Activity Diagram illustrates the step-by-step workflow followed by the Intrusion Detection System. It begins with the user uploading a dataset and proceeds through preprocessing, feature selection, model training, and evaluation. Decision points help determine whether the model is ready for prediction or requires retraining. Once evaluation criteria are satisfied, the system performs attack detection and generates output results. The diagram captures parallel processes, data transformation steps, and user interactions in a clear, sequential flow. This helps analysts visualize the entire lifecycle of system operations, ensuring smooth transitions between stages and identifying potential areas for optimization or improvement.

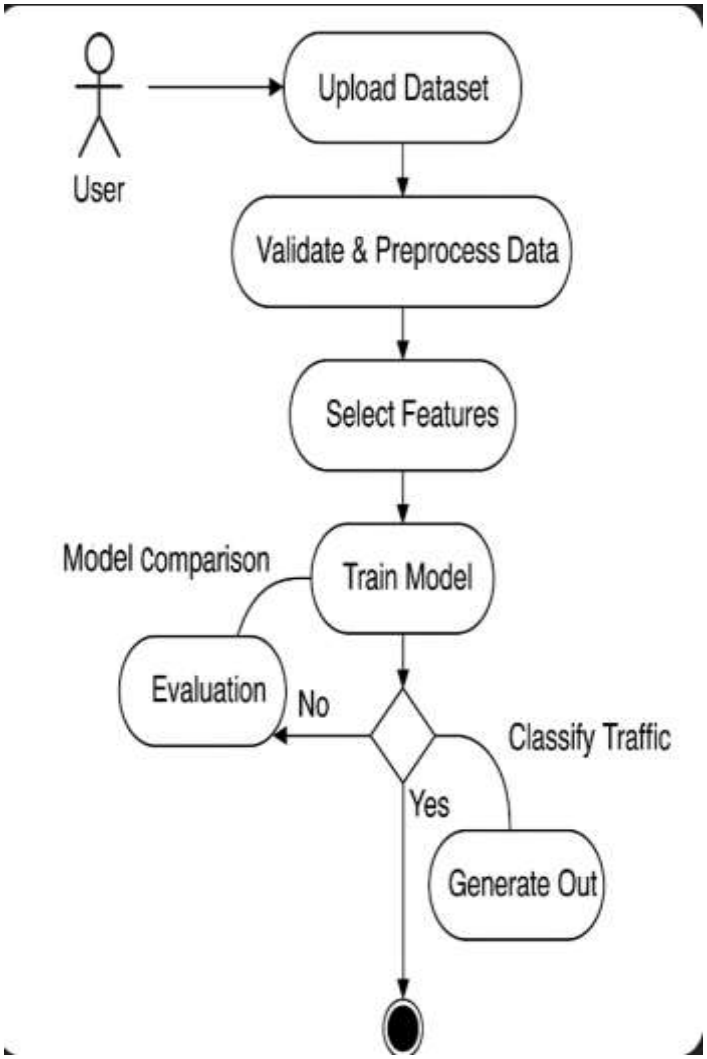

Figure 4.5 Activity Diagram

4.3.5 COLLABORATION DIAGRAM

The Collaboration Diagram focuses on how system components communicate with each other to achieve intrusion detection. It shows relationships and message exchanges among modules such as Upload Handler, Preprocessor, FeatureSelector, Trainer, Evaluator, and the User interface. Each message reflects an action, such as sending raw records, returning encoded data, selecting features, or producing model predictions. This diagram emphasizes interconnectedness rather than timing, helping developers understand how modules cooperate to complete tasks. It highlights responsibility assignment, object interaction, and the structural organization of processes. This view improves system modularity, communication efficiency, and understanding of collaborative behavior.

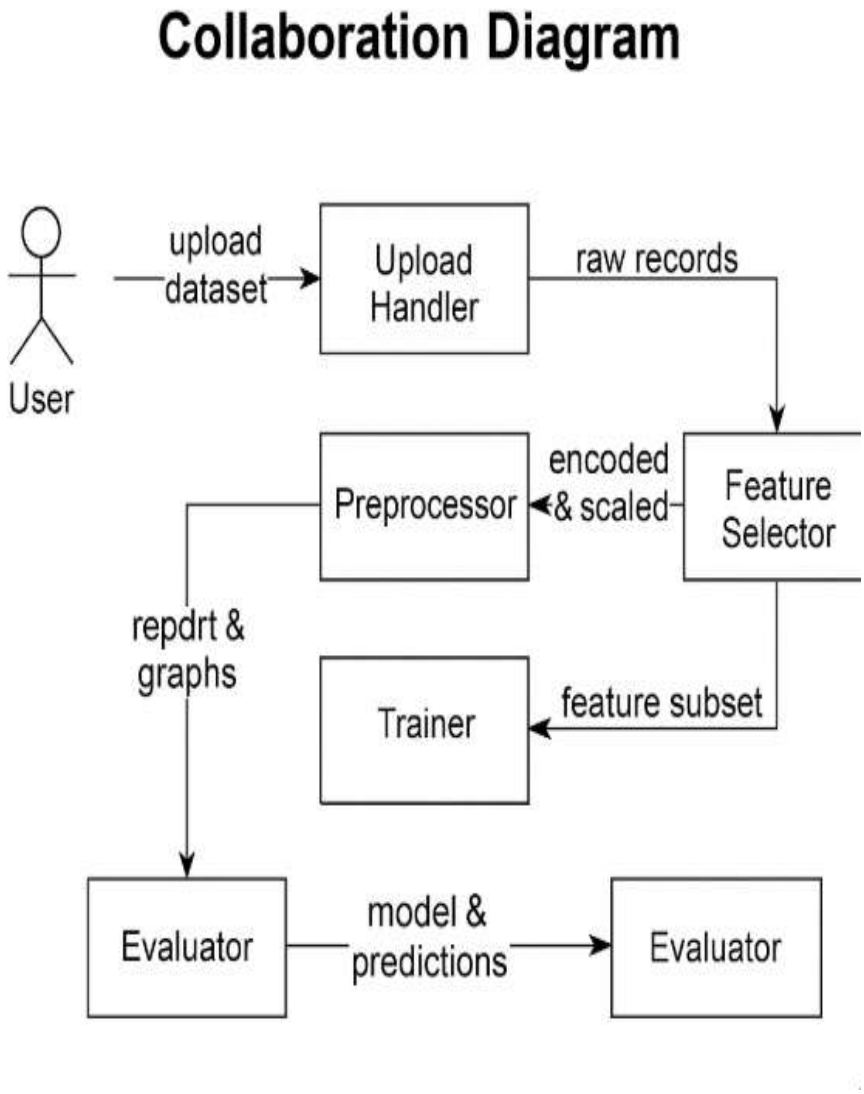

Figure 4.6 Collaboration Diagram

4.3.6 ENTITY RELATIONSHIP DIAGRAM

The ER Diagram represents the data model of the Intrusion Detection System by defining entities, attributes, and relationships. Key entities include Dataset, Model, Prediction, User, and Evaluation Result. Attributes such as dataset_id, model_id, accuracy, label, and timestamp describe essential metadata stored in the system. Relationships show how datasets generate models, how models produce predictions, and how evaluations are linked to specific experiments. The diagram ensures a clear logical organization of stored information, supporting consistency, integrity, and efficient querying. It is crucial for database design and helps maintain structured records needed for analysis, model tracking, and auditability.

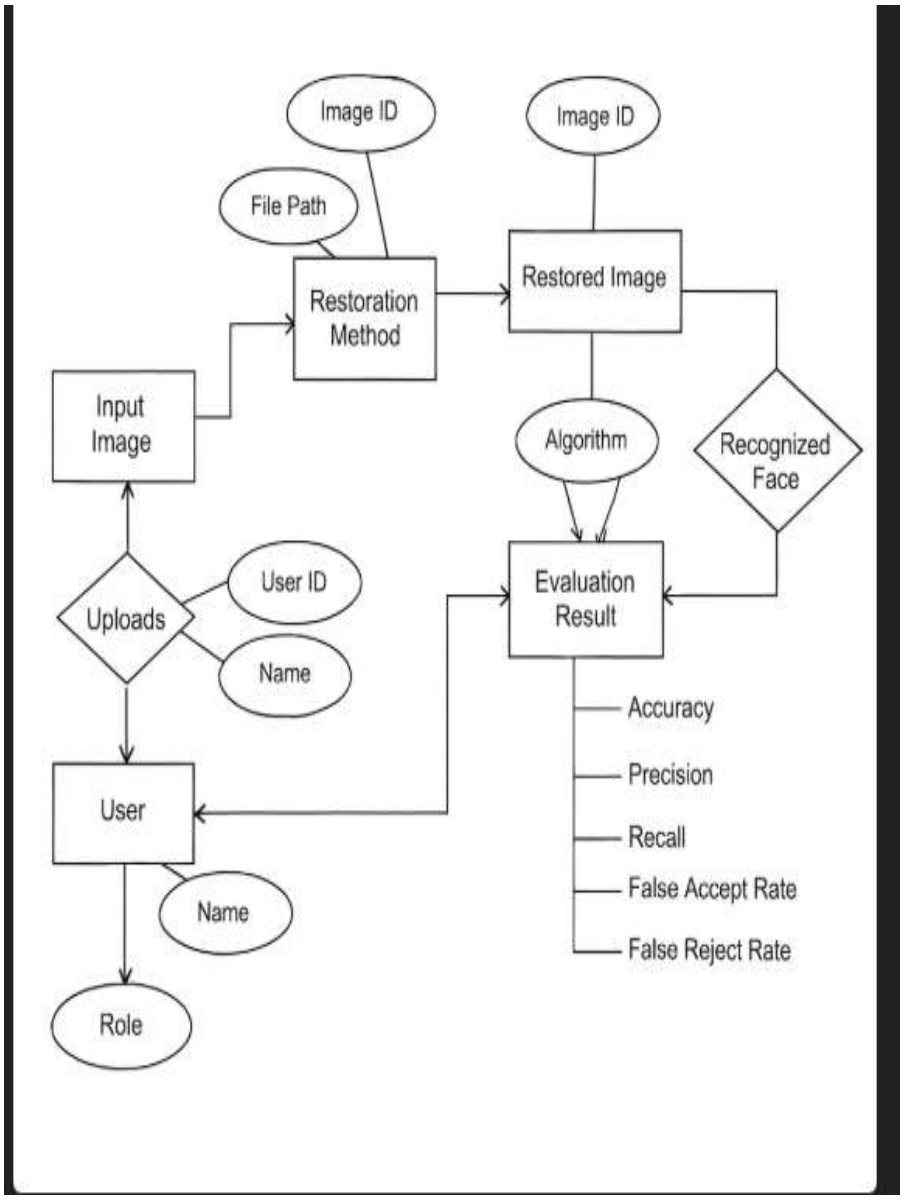

Figure 4.7 ENTITY RELATIONSHIP Diagram

4.3.7 BLOCK DIAGRAM

The Block Diagram presents the high-level architecture of the Intrusion Detection System, illustrating the major functional components and flow of data between them. It typically includes modules such as Input Layer, Preprocessing, Feature Selection, Machine Learning Model, Evaluation, Attack Detection, and Output Layer. Each block performs a specific function and communicates sequentially to process and classify network traffic. The diagram provides a simplified overview of the system's workflow, enabling easy understanding of core

operations. It highlights modularity, scalability, and the logical arrangement of processes, making it useful for presentations, documentation, and architectural planning.

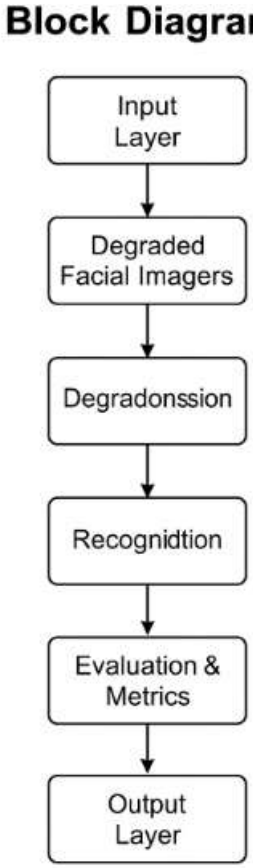

Figure 4.8 BLOCK Diagram

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING
KAMALA INSTITUTE OF TECHNOLOGY & SCIENCE, SINGAPUR
HUZURABAD, KARIMNAGAR, TELANGANA, INDIA – 505 46