**A Major Project Report**

**on**

**Deep Ensemble Framework with Supervised Learning for Secure Iot Network**

Submitted in partial fulfillment of the requirements

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)**

**Submitted By**

**Batch: 12**

**DORNADULA LOKESWAR (21UJ5A6703)**

**GOUNI SNEEHA LATHA (20UJ1A6721)**

**GUVVALA MOHAN (20UJ1A6724)**

**PURAM NAWAZ REDDY (20UJ1A6742)**

**Under the esteemed guidance of**

**Mrs. K. PARAMESHWARI** MTech.

Assistant Professor

Department of Computer Science & Engineering

MALLA REDDY ENGINEERING COLLEGE AND MANAGEMENT SCIENCES



**MALLA REDDY ENGINEERING COLLEGE AND MANAGEMENT SCIENCES**

(Approved by AICTE New Delhi & Affiliated to JNTU Hyderabad)

1. Kistapur, Medchal, Malkajgiri Dist.- 501401.
2. **2020-2024**

**Affiliated to**

****

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD**

**KUKATPALLY, HYDERABAD-85.**

**MALLA REDDY ENGINEERING COLLEGE AND MANAGEMENT SCIENCES**

(Approved by AICTE New Delhi & Affiliated to JNTU Hyderabad)

1. Kistapur, Medchal, Malkajgiri Dist.- 501401.

****

**CERTIFICATES**

This is to certify that the Major Project report entitled Deep Ensemble Framework with Supervised Learning for Secure Iot Network“submitted by

**DORNADULA LOKESHWAR (21UJ5A6703)**

**GOUNI SNEEHALATHA (20UJ1A6721)**

**GUVVALA MOHAN (20UJ1A6724)**

**PURAM NAWAZ REDDY (20UJ1A6742)**

To the department of **Computer Science & Engineering, Malla Reddy Engineering College and Management Sciences**, in partial fulfilment for the award of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)** during the academic year 2020-2024.

**Internal Guide Head Of The Department**

**Mrs. K. PARAMESWARI M.Tech**. **Mr.G.CHAKRAPANIM.Tech.**

Assistant Professor Assistant Professor

Dept. of Computer science and Dept. of Computer science and Engineering(Data science) Engineering (Data Science)

**EXTERNAL EXAMINE**

**ACKNOWLEDGEMENT**

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that I have now the opportunity to express our guidance to all of them.

We are thankful to **Mr. V. MALLA REDDY** chairmanof “Malla Reddy Engineering College and Management Sciences” for accepting us and providing us with an opportunity to do a project in their esteemed organization.

We are thankful to **Mr.Dr.SUBBA RATNAM** directorof “Malla Reddy Engineering College and Management Sciences” helping me to undergo project work as a part of the university curriculum.

We are thankful to Principal **Dr. M. SREEDHAR REDDY** Ph.D(CSE),Ph.D (Mgmt),FIE,MIEEE,LMISTE,LCSI.. Malla Reddy Engineering College and Management Sciences for helped us to undergo project work as a part of the university curriculum.

Our special thanks to **Mr. G. CHAKRAPANI** M.Tech.,  Assistant Professor & Head of the Computer Science & Engineering Department, and **Mrs.S.PARAMESHWARIM.Tech**. Assistant Professor in the Computer Science & Engineering Department for guiding us in the right way to complete our project in the right time.

As a gesture of respect for our Family Member and support we received from them, we dedicate this work to them

And, we would like to express our heartfelt thanks Faculty Members of the CSE(DATA SCIENCE) Department, Lab Technicians and Friends, one and all who has helped us directly or indirectly in successful completion of the Major Project.

**DORNADULA LOKESHWAR (21UJ5A6703)**

**GOUNI SNEEHALATHA (20UJ1A6721)**

**GUVVALA MOHAN (20UJ1A6724)**

**PURAM NAWAZ REDDY (20UJ1A6742)**



**MALLAREDDY ENGINEERING COLLEGE AND MANAGEMENT SCIENCES**

(Approved by AICTE New Delhi & Affiliated to JNTU Hyderabad) Kistapur, Medchal Dist.- 501401.

## DECLARATION

We DORNADULA. LOKESHWAR (21UJ5A6703), G.SNEEHALATHA (20UJ1A6721), G.MOHAN (20UJ1A6724), P.NAWAZ REDDY (20UJ1A6742), students of ‘Bachelor of Technology in COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE),during the session:2020-2024, Malla Reddy Engineering College and Management Sciences, Medchal, Hyderabad, hereby declare that the work presented in this Project Work entitled Deep Ensemble Framework with Supervised Learning for Secure ” is the outcome of our bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material that has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

**DATE:**

**DORNADULALOKESH (21UJ5A6703)**

**GUVVALA MOHAN (20UJ1A6724)**

**GOUNI SNEEHA (20UJ1A6721)**

**PURAM NAWAZ (20UJ1A6742)**

**ABSTRACT**

In the last decade, huge growth is recorded globally in computer networks and Internet of Things (IoT) networks due to the exponential data generation, approximately zettabyte to a petabyte. Consequently, security issues have also been arisen with the network growth. Electricity theft represents a pressing problem that has brought enormous financial losses to electric utility companies worldwide. In the United States alone, $6 billion worth of electricity is stolen annually. Traditionally, electricity theft is committed in the consumption domain via physical attacks that includes line tapping or meter tampering. It as some limitations cannot reach there milestone to detect the attack. Therefore, this project evaluating performance of various deep ensemble algorithms such as convolutional neural network (CNN) with random forest (RF) for electricity cyber-attack detection. Now-a-days in advance countries solar plates are used to generate electricity and these users can sale excess energy to other needy users and they will be maintained two different meters which will record consumption and production details. While producing some malicious users may tamper smart meter to get more bill which can be collected from electricity renewable distributed energy. This attack may cause huge losses to agencies. To detect such attack, this project is employing deep ensemble learning models effectively.

**KEYWORDS**  : Electricity theft detection, energy metering system, random forest, convolution neural networks, Internet of thing

**TABLE OF CONTENTS**

**ABSTRACT v**

**LIST OF FIGURES viii**

**ABBREVIATIONS ix**

**CH NO DESCRIPTION PG NO**

**CHAPTER 1 INTRODUCTION 1-2**

1.1 HISTORY 1.2 MOTIVATION

1.3 PROBLEM STATEMENT

1.4 APPLICATION

**CHAPTER 2 LITERATURE SURVEY 3-6**

**CHAPTER 3 EXISTING SYSTEM 7-8**

3.1 DEEP NEURAL NETWORK

**CHAPTER 4 PROPOSED SYSTEM 9-17**

4.1 OVERVIEW

4.2 DATA PREPROCESSING

4.3 CNN CLASSIFIER

4.4 RANDAM FOREST ALGORITHMS

4.5 TYPES OF ENSEMBLES

**CHAPTER 5** **UML DIAGRAMS 18-22**

5.1 COMPONENT DIAGRAM

5.2 CLASS DIAGRAM

5.3 SEQUENCE DIAGRAM

5.4 ACTIVITY DIAGRAM

5.5 COLLABORATION DIAGRAM

5.6 DEPLOYMENT DIAGRAM

5.7 DATAFLOW DIAGRAM

**CHAPTER 6 SOFTWARE ENVIRONMENT 23-37**

**CHAPTER 7 SYSTEM REQUIREMENTS SPECIFICATIONS 38**

**CHAPTER- 8** **FUNCTIONAL REQUIRMENT 39-43 CHAPTER- 9 SOURCE CODE 44-54**

**CHAPTER -10** **RESULTS AND DISCUSSION 55-64**

10.1 IMPLEMENTATION DESCRIPTION

10.2 DATASET DESCRIPTION

10.3 RESULTS AND DESCRIPTION

**CHAPTER 11 CONCLUSION AND FUTURE SCOPE 65 REFERENCES 66-67**

**LIST OF FIGURES**

**FIG NO DESCRIPTION PGNO**

Figure3.1 Deep Neural Network 7

Figure 4.1 Block diagram of proposed system 10

Figure 4.2 Representation of convolution layer process 12

Figure 4.3 convolution layer process 13

Figure 4.4 Random Forest algorithm. 14

Figure 4.5 RF Classifier analysis. 16

Figure 4.6 Boosting RF Classifier 16

Figure 5.1 Class diagram. 19

Figure 5.2 Sequence diagram. 20

Figure 5.3 Activity diagram. 21

Figure 5.4 Collaborative diagram 22

Figure 5.5 Deployment diagram 22

Figure 5.6 Dataflow diagram 23

Figure 5.7 Component diagram 24

Figure 10.1 Perform A Dataset 54

Figure 10.2 Upload A Dataset Description 55

Figure 10.3 dataset of Optimization Of IoT 55

Figure 10.4 Perform the Preprocessing dataset of Iot 56

Figure 10.5 perform Of feed Forward neural Network 57

Figure 10.6 Performing a Deep Learning CNN 57

Figure 10.7 DNN ROC Graph 58

**ABBREVIATIONS:**

**IOT**- Internet Of Things

**MIC**-Maximum Information Coefficient

**CFSFDC**- the clustering technique by fast search and find of density peaks

**PPETD**- privacy-preserving electricity theft detection

**VGG** - Visual Geometry Group

**FA-XGB**- Firefly Algorithm based Extreme Gradient Boosting (FA-XGBoost)

**SVM -** Support Vector Machine

**CNN -** Convolution Neural Network

**LR-** Logistic Regression

**MCC -** Matthews Correlation Coefficient

**LSTM** - Long Short-Term Memory

**DNN –** Deep Neural Network

**ANN** – Artificial Neural Network

**WORA -** Write Once Run Anywhere

**CHAPTER 1**

**INTRODUCTION**

**1.1 History**

The rise of the Internet of Things (IoT) marks a significant milestone in the evolution of technology. Initially conceptualized as a network of interconnected devices capable of exchanging data, the IoT has rapidly transformed into a global phenomenon, permeating various sectors including healthcare, smart homes, and industrial automation. The inception of IoT can be traced back to the early 2000s when technologists envisioned a future where everyday objects would be embedded with sensors, enabling seamless communication and automation. However, it wasn't until the mid-2010s that IoT gained widespread traction, driven by advancements in wireless communication protocols, majoraturization of electronics, and the proliferation of cloud computing. As IoT adoption surged, so did concerns regarding its security. With billions of devices interconnected, often with majormal security measures in place, the IoT ecosystem became a prime target for cyber-attacks. The vulnerabilities inherent in IoT devices, stemming from their resource-constrained nature and diverse communication protocols, exacerbated the security challenges. Traditional security mechanisms such as firewalls and encryption techniques proved inadequate in safeguarding IoT networks against evolving threats. Consequently, researchers and industry stakeholders embarked on a quest to develop innovative solutions capable of fortifying the security posture of IoT deployments.

**1.2 Research Motivation:**

The escalating security concerns surrounding IoT networks underscore the critical need for proactive measures to mitigate cyber threats effectively. As IoT devices permeate various facets of daily life, ranging from healthcare to smart cities, ensuring the security and privacy of these interconnected systems becomes paramount. The motivation for the proposed research emanates from the imperative to address the pressing security challenges facing IoT deployments. Conventional security approaches, characterized by static rule-based systems and signature-based intrusion detection systems (IDS), fall short in combating sophisticated and novel attacks. These methods struggle to adapt to the dynamic threat landscape of IoT networks, often resulting in false positives and negatives. Moreover, the resource constraints of IoT devices limit the efficacy of traditional security mechanisms, as they entail significant computational overhead.

**1.3 Problem Statement:**

The core challenge lies in developing a robust and adaptive security framework capable of effectively mitigating the diverse array of threats targeting IoT networks. Conventional security mechanisms are ill-equipped to address the dynamic nature of IoT environments, where new vulnerabilities emerge rapidly, and attackers continuously devise novel evasion techniques. Moreover, the resource constraints of IoT devices necessitate lightweight and efficient security solutions that majormize computational overhead. The inadequacies of existing security approaches underscore the need for a paradigm shift towards innovative methodologies that leverage advanced technologies such as deep learning and ensemble learning. However, developing such methodologies entails overcoming several key challenges, including the efficient utilization of computational resources, scalability across diverse IoT environments, and real-time threat detection capabilities.

**1.4 Applications:**

The proposed deep ensemble framework fortified by supervised learning techniques holds immense potential across various applications within the realm of IoT security. In healthcare, where IoT devices are extensively utilized for remote patient monitoring and medical diagnostics, ensuring the confidentiality and integrity of sensitive health data is paramount. The proposed framework can bolster security measures within healthcare IoT deployments by detecting anomalous behavior indicative of potential data breaches or cyber-attacks. Similarly, in smart home environments, where IoT devices control various household appliances and facilitate home automation, protecting against unauthorized access and malicious manipulation is critical. By leveraging the capabilities of deep learning models, the proposed framework can analyze patterns in device interactions and network traffic, thereby detecting and mitigating security threats in real-time. This enhances the overall security posture of smart home ecosystems, ensuring the privacy and safety of occupants. In industrial automation, where IoT technologies are integral to monitoring and controlling critical infrastructure, the proposed framework can

**CHAPTER 2**

**LITERATURE SURVEY**

# Asma, et.al [1] Presented A Review On The Security Of Iot Networks: From Network Layer’s Perspective By  Routing Protocol For Low Power And Lossy Networks (Rpl) Is A Network Layer Protocol, Specially Designed For Routing In Iot

# Ana, Et.al [2] Implemented A On Wide-Area Iot Networks, Lightweight Security And Their Applications—A Practical Review Proposed By  However, Iot Devices Have Limited Energy And Computational Resources

# Mahamat, Et.al [3] Presented Achieving Efficient Energy-Aware Security In Iot Networks: A Survey Of Recent Solutions And Research Challenges Proposed By  We Propose A Taxonomy Of Recent Solutions That Reduce Energy Consumption While Efficiently Securing Iot Networks.

# Aldhaheri, Et Al [4] Deep Learning For Cyber Threat Detection In Iot Networks: A Review Of *Internet Of Things Proposed By* . [Deep Learning](https://www.sciencedirect.com/topics/engineering/deep-learning) Has Shown Promise In Effectively Detecting And Preventing Cyberattacks On Iot

# Et.al [5] In This Article, We Propose A Secure Smart Blockchain Iot Architecture Based On Graph Neural Networks (Gnns) Named Gtxchain, Using A Distributed Intelligent Prophecy Machine To Obtain Off-Chain Data Tibrewal,

Et.al [6] Implement Blockchain Technology For Securing Cyber-Infrastructure And Internet Of Things Networks Proposed By  Blockchain Technology Provides A Higher Level Of Trust Among Systems/Devices With Providing Higher Level Of Anonymity Among Its Blocks

# Amit, Et.al [7] Design Of Metaheuristic Optimization Algorithms For Deep Learning Model For Secure Iot Environment Proposed By The Present Paper Introduces Two Novel Metaheuristic Optimization Algorithms For Optimizing The Weights Of Deep Learning (Dl) Models

# Jongseok, Et .Al [8] Secure Iot Framework And 2d Architecture For End-To-End Security Proposed By  The Iot Application Provides Users With Iot Services. To Use The Iot Services, It Needs To Access To Sensing Data. Especially

# Al Hwaitat, Ahmad K., Et.al [9] A New Blockchain-Based Authentication Framework For Secure Iot Networks Proposed By  Blockchain-Based Solutions That Improve The Quality Of Iot Networks Are Becoming Increasingly Used.

# 

# Xin, Et Al. [10] Sector Iot: A Security Framework For The Internet Of Things Presented By  Environmental Monitoring Networks, And Infrastructure Management Systems. The Potential Benefits Of The Iot Are As Profound As They Are Diverse

# Chakrabarty, Et.al [11] Secure Smart Cities Framework Using Iot And Ai Implemented By  Smart City Cyber Systems Carry Critical Data Beginning With The Simple Sensor Data Captured By Iot Devices Up To And Including Providing The Essential Services And Commands Dependent Upon That Data. Management Framework

# Abou El Houda, Et.al [12] A Low-Latency Fog-Based Framework To Secure Iot Applications Using Collaborative Federated Learning  We Propose A Novel Low-Latency Fog-Based Framework, Called Fogfed

# M. Pushpalatha, Et.al [13] Wils-Tris—A Novel Optimized Deep Learning Based Intrusion Detection Framework For Iot Networks  These Iot Devices Simplify Day To Day Life Activities, These Networks Are Vulnerable To Many Security Threats.

# Balaswamy, Et Al. [14] A Secure Framework For Protecting Clinical Data In Medical Iot Environment Presented By  Correspondence And Sharing Of Data Are Additionally Under Investigation In Iot Condition

# Subhashini, Et Al [15] The Role Of Internet Of Things (Iot) In Smart City Framework Proposed By  You Could Call And You Could Text, Sure, But Now You Can Read Any Book, Watch Any Movie, Or Listen To Any Song All In The Palm Of Your Hand.

# 

# Abu Al‐Haija, Et Al , [16] Boost‐Defence For Resilient Iot Networks Proposed By  Due To The Concrete Involvement Of Iot Networks In Critical Infrastructures And Cyber-Physical Systems, Defending Them Against Cyber-Attacks Has Led To Extensive Research Efforts To Propose Effective Countermeasures Against Such Attacks

# Akashdeep, Et Al. [17] Iot Attack Framework And Unique Taxonomy . Iot Domains Include Smart Homes, Healthcare, Manufacturing, Smart Wearables, Smart Cities, Smart Grids, Industrial Iot, Connected Vehicles, And Smart Retail. Different Iot Models Involve Human-To-Iot, Iot-To-Iot, Iot-To-Traditional Systems Architectures.

# Sanjay Misra Et Al [18] Feature Extraction And Artificial Intelligence-Based Intrusion Detection Model For A Secure Internet Of Things Networks Implemented By  Artificial Intelligence (Ai)

# Saqib Et Al [19] A Lightweight Three Factor Authentication Framework For Iot Based Critical Applications Proposed By As Users Access These Services Remotely In A [Ubiquitous Environment](https://www.sciencedirect.com/topics/computer-science/ubiquitous-environment) Over Public Channels, It Becomes Imperative To Secure Their Communication Iot

# 

# Ting, Et.al [20] A Secure Framework For Iot-Based Smart Climate Agriculture System Presented By . Data Privacy And Security As A Big Challenge In Current Internet Of Things (Iot) Applications

Chee, Kok Onn, et al. [21]propose a novel simulation framework and a [software tool](https://www.sciencedirect.com/topics/computer-science/software-development-tool), IoT Security Simulator (IoTSecSim). IoTSecSim is operated based on a framework we propose for modelling and simulating cyber-attacks and various defences in IoT networks

Alwahedi, Fatima, et al. [22]survey provides a comprehensive overview of current trends, methodologies, and challenges in applying machine learning for cyber threat detection in IoT environments

Saied, Mohamed, [23]proposed FA-ML technique exhibits an exceptional capability to enhance intrusion detection accuracy within the WSN-IoT landscape. Secondly, the amalgamation of the Firefly Algorithm and machine learning introduces a novel dimension to the domain of security-oriented optimization techniques.

 Sharma, Bhawana, et al. [24] study provides an in-depth survey highlighting the recent advances in artificial intelligence for improving the security of IoT. It summarizes and organizes the recent related research, then presents a comprehensive discussion on research challenges, open issues, and needed future research.

Bilakanti, Harini, et al [25] study is the identification of anomalies in IoT devices arising sensor tampering, with an emphasis on the application of machine learning techniques. While supervised methods like one-class SVM, Gaussian Naive Bayes

Mansour, et al. [26] paper, our key scientific contribution revolves around devising a security protocol founded on HMAC. This protocol guarantees authentication and preserves the integrity of neural network weights exchanged between Fog computing nodes and IoT devices

Lilhore, [27]proposed a Hybrid model using enhanced light-weight CNNs architecture (MobileNetV3-SVM) and Transfer learning (TL) for [intrusion detection](https://www.sciencedirect.com/topics/computer-science/intrusion-detection) in 5G communication. The proposed model utilizes the advantages of a multi-layered structure, which enables it to acquire knowledge from raw network information hierarchically

Sami, [28]proposed MLHA was carefully considered and managed to effectively enhance IoT security without compromising the performance and efficiency of IoT devices. The new hashing algorithm is suitable for all IoT devices, ranging from small

Abdullah Yousafzai [29]survey explores the role of Blockchain and federated learning in addressing security and privacy issues, particularly those associated with IDS/IPS in IIoT networks. Insights on the possibilities of machine learning, federated learning, and Blockchain in supporting IDS to monitor IIoT network traffic for [anomaly detection](https://www.sciencedirect.com/topics/engineering/anomaly-detection) are discussed in detail through state of the art. Furthermore

Singh, Sushil Kumar, et al.[30] propose a GRU-based Digital Twin Framework for Data Allocation in IoT-enabled Smart Home Networks. Data and resource allocation of smart home applications are completed at the virtual twin layer using Gated Recurrent Unit (GRU)-based Digital Twin Networks

Kizza, Joseph Migga.[31] smart environment is achieved by seamless ubiquitous sensing, data analytics, and information representation with cloud computing as the unifying framework. The environment allows for virtually endless opportunities and connections to take place, many of which we cannot even think of or fully understand the impact of

Khalid, Mizna, et al. [32]proposed work is to resolve the incurring challenges of IoT associated with management and access control security. Each IoT domain implementation has particular features and needs separate [access control policies](https://www.sciencedirect.com/topics/computer-science/access-control-policies) to be considered in order to design a secure solution

Khan, AB Feroz, et al. [33]study introduces a novel technique to counter reactive DDoS attacks in the Internet of Things (IoT) environment. The objective is to enhance the security of the IoT by preventing and mitigating DDoS attacks. The proposed technique involves the use of a dynamic frequency hopping mechanism that shifts the operating

Xie, Hui, et al.[34]propose a TEE-and-Blockchain-supported IoT data sharing architecture(TEBDS), which combines on-chain and off-chain methods to meet the security requirements of the IoT data sharing framework.

Dhar, Shalini.[35] proposed approach can be applied to any file-changing wireless IoT network that needs to exchange multimedia data such as healthcare data, IoT data in wearable devices, traffic data in smart cities, etc

Rekha, Shashi, et al. [36]"proposed model was evaluated on IoT-23, BoT-IoT, and Edge-IIoT datasets using the GPU. When compared to existing IDS, our approach provides good rating performance features of ACC, recall, and precision, with around 99.9% on record detection and calculation time of 33.68 s for learning and 0.02156 s for detection .

Ragesh, G. K., and Ajay Kumar.[37] "proposed system attained a throughput of 96% with less than 0.1 ms delay. An enhanced trust-based secure routing and message delivery protocol attained better performance than existing such as Advanced Encryption Standard, Tiny Encryption Algorithm, eXtended TEA, and Rivest–Shamir–Adleman encryption schemes .

Li, Xuanang, et al.[38] "PSAP-WSN functions of WSN include providing users with real-time monitoring information, deploying regional information collection, and synchronizing with the Internet. Security in WSNs is becoming increasingly essential because of the across-the-board nature of wireless technology in many fields

Fathy, Cherine, and Hassan M. Ali [39] propose the integration of lightweight cryptography techniques into the IoT ecosystem for smart agriculture to meet the requirements of resource-constrained IoT devices

Khan, Wali Ullah, et al. [40]proposed scheme, we also present the suboptimal NOMA and conventional [orthogonal multiple access](https://www.sciencedirect.com/topics/engineering/orthogonal-multiple-access) as benchmark schemes. Monte Carlo simulation results demonstrate the superiority of the NOMA

Hasan et. a [41] implemented a novel data pre-processing algorithm to compute the missing instances in the dataset, based on the local values relative to the missing data point. Furthermore, in this dataset, the count of electricity theft users was relatively low, which could have made the model inefficient at identifying theft users

Zheng et. al [42] combined two novel data majorng techniques to solve the problem. One technique is the maximum information coefficient (MIC), which can find the correlations between the nontechnical loss and a certain electricity behavior of the consumer

Li et. al [43] presented a novel CNN-RF model to detect electricity theft. In this model, the CNN is similar to an automatic feature extractor in investigating smart meter data and the RF is the output classifier. Because a large number of parameters must be optimized that increase the risk of overfitting

Nabil et. al [44] proposed an efficient and privacy-preserving electricity theft detection scheme for the AMI network and we refer to it as PPETD. Our scheme allows system operators to identify the electricity thefts, monitor the loads, and compute electricity bills efficiently using masked fine-grained meter readings without violating the consumers' privacy

Khan et. al [45] presents a new model, which is based on the supervised machine learning techniques and real electricity consumption data. Initially, the electricity data are pre-processed using interpolation, three sigma rule and normalization methods

Kocaman et. al [46] developed by using deep learning methods on real daily electricity consumption data (Electricity consumption dataset of State Grid Corporation of China). Data reduction has been made by developing a new method to make the dataset more usable and to extract meaningful results.

Li et. al [47] presented a novel approach for automatic detection by using a multi-scale dense connected convolution neural network (multi-scale DenseNet) in order to capture the long-term and short-term periodic features within the sequential data. They compare the proposed approaches with the classical algorithms, and the experimental results demonstrate that the multi-scale DenseNet approach can

Aldegheishem et. al [48] developed two novel ETD models. A hybrid sampling approach, i.e., synthetic minority oversampling technique with edited nearest neighbor, is introduced in the first model. Furthermore, AlexNet is used for dimensionality reduction and extracting useful information from electricity consumption data. Finally, a light gradient boosting model is used for classification purpose

Lilhore, [49 ]proposed a Hybrid model using enhanced light-weight CNNs architecture (MobileNetV3-SVM) and Transfer learning (TL) for [intrusion detection](https://www.sciencedirect.com/topics/computer-science/intrusion-detection) in 5G communication. The proposed model utilizes the advantages of a multi-layered structure, which enables it to acquire knowledge from raw network information hierarchically

Sami, [50] proposed MLHA was carefully considered and managed to effectively enhance IoT security without compromising the performance and efficiency of IoT devices. The new hashing algorithm is suitable for all IoT devices, ranging from small

**CHAPTER 3**

**EXISTING SYSTEM**

**3.1 Deep neural network**

**What is a deep neural network?**

At its simplest, a neural network with some level of complexity, usually at least two layers, qualifies as a deep neural network (DNN), or deep net for short. Deep nets process data in complex ways by employing sophisticated math modeling.

To truly understand deep neural networks, however, it’s best to see it as an evolution. A few items had to be built before deep nets existed.

First, machine learning had to get developed. ML is a framework to automate (through algorithms) statistical models, like a linear regression model, to get better at making predictions. A model is a single model that makes predictions about something. Those predictions are made with some accuracy. A model that learns—machine learning—takes all its bad predictions and tweaks the weights inside the model to create a model that makes fewer mistakes.

The learning portion of creating models spawned the development of artificial neural networks. ANNs utilize the hidden layer as a place to store and evaluate how significant one of the inputs is to the output. The hidden layer stores information regarding the input’s importance, and it also makes associations between the importance of combinations of inputs.

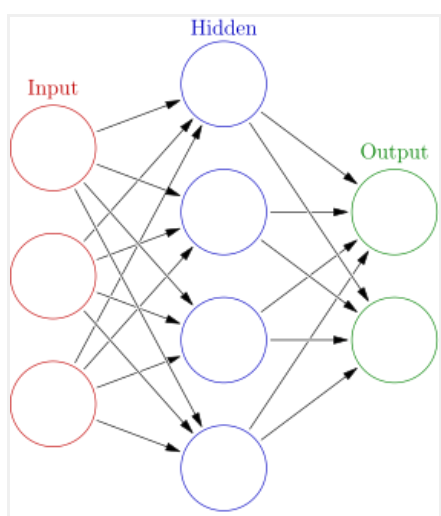


Fig 3.1 Deep Neural Network

Deep neural nets, then, capitalize on the ANN component. They say, if that works so well at improving a model—because each node in the hidden layer makes both associations and grades importance of the input to determajorng the output—then why not stack more and more of these upon each other and benefit even more from the hidden layer?

So, the deep net has multiple hidden layers. ‘Deep’ refers to a model’s layers being multiple layers deep.

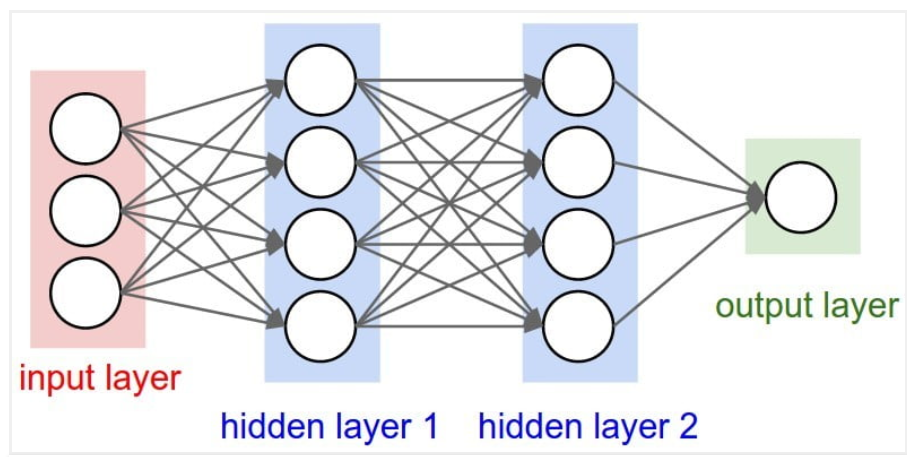


Fig 3.2 Artificial Neural network

**CHAPTER 4**

**PROPOSED SYSTEM**

**4.1 OVERVIEW**

The provided Python code encapsulates a robust graphical user interface (GUI) application developed using the Tkinter library, offering a comprehensive toolkit for researchers and practitioners engaged in addressing the security challenges prevalent in Internet of Things (IoT) networks. With IoT's exponential growth permeating various domains like healthcare, smart homes, and industrial automation, ensuring the security and privacy of interconnected devices has become paramount. The GUI application serves as a practical platform to navigate through the intricacies of dataset handling, preprocessing, model training, performance evaluation, and predictive analysis, all tailored towards fortifying the security posture of IoT deployments. Through seamless integration of deep learning techniques, including Feed Forward Neural Networks (DNN) and Convolutional Neural Networks (CNN), alongside ensemble learning with Random Forest, the application empowers users to develop and assess sophisticated security frameworks adept at detecting and mitigating cyber threats targeting IoT ecosystems. This multifaceted approach not only enhances the resilience and trustworthiness of IoT networks but also fosters a deeper understanding of the evolving threat landscape, paving the way for proactive security measures and informed decision-making. The interface offers intuitive functionalities, facilitating dataset upload with a file dialog, displaying dataset information, and enabling preprocessing steps such as handling missing values and encoding categorical variables. Users can seamlessly transition to model training, where the application orchestrates the training process for DNN, CNN, and CNN combined with Random Forest classifier. This enables users to explore diverse methodologies and evaluate their performance across key metrics like accuracy, precision, recall, and F1 score. Additionally, the application provides a prediction module, allowing users to upload new datasets and predict the likelihood of cyber attacks based on the trained models. Visualizations in the form of ROC graphs and bar graphs further enrich the user experience, offering insights into the comparative performance of different algorithms. By encapsulating these functionalities within a user-friendly interface, the application democratizes access to advanced security techniques, empowering stakeholders across academia, industry, and policymaking to contribute towards the development of resilient IoT ecosystems. Through ongoing collaboration and iterative refinement, the GUI application serves as a catalyst for innovation, driving forward the collective efforts aimed at securing the interconnected devices shaping our digital future.

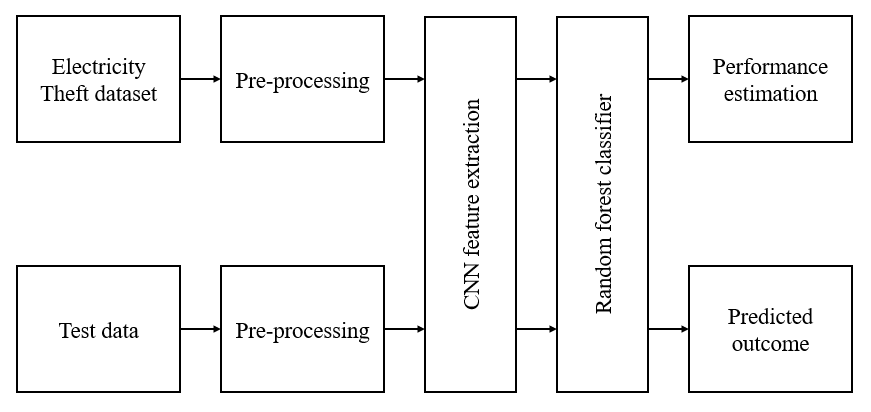
****

Fig. 4.1: Block diagram of proposed system.

**4.2 Data Preprocessing in Machine learning**

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So, for this, we use data pre-processing task.

**Why do we need Data Pre-processing?**

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data pre-processing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

* Getting the dataset
* Importing libraries
* Importing datasets
* Finding Missing Data
* Encoding Categorical Data
* Splitting dataset into training and test set
* Feature scaling

**Splitting the Dataset into the Training set and Test set**

In machine learning data pre-processing, we divide our dataset into a training set and test set. This is one of the crucial steps of data pre-processing as by doing this, we can enhance the performance of our machine learning model.

Suppose if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:

A picture containing shape

Description automatically generated

**Training** **Set**: A subset of dataset to train the machine learning model, and we already know the output.

**Test** **set**: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

**4.3 CNN Classifier**

According to the facts, training and testing of CNN involves in allowing every source data via a succession of convolution layers by a kernel or filter, rectified linear unit (ReLU), max pooling, fully connected layer and utilize SoftMax layer with classification layer to categorize the objects with probabilistic values ranging from.

Convolution layer is the primary layer to extract the features from a source image and maintains the relationship between pixels by learning the features of image by employing tiny blocks of source data. It’s a mathematical function which considers two inputs like source image where and denotes the spatial coordinates i.e., number of rows and columns. d is denoted as dimension of an image (here d=3 since the source image is RGB) and a filter or kernel with similar size of input image and can be denoted as ..

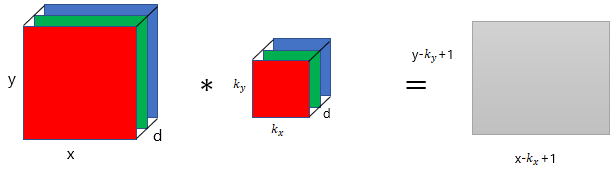
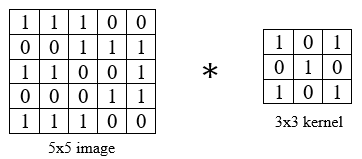


Fig. 4.2: Representation of convolution layer process.

The output obtained from convolution process of input image and filter has a size of , which is referred as feature map. Let us assume an input image with a size of 5×5 and the filter having the size of 3×3. The feature map of input image is obtained by multiplying the input image values with the filter values.



(a)

A picture containing diagram

Description automatically generated

(b)

Fig. 4.3: Example of convolution layer process (a) an image with size 5×5 is convolving with 3×3 kernel (b) Convolved feature map.

**ReLU layer**

Networks those utilizes the rectifier operation for the hidden layers are cited as rectified linear unit (ReLU). This ReLU function is a simple computation that returns the value given as input directly if the value of input is greater than zero else returns zero. This can be represented as mathematically using the function over the set of 0 and the input x as follows:

**Max pooing layer**

This layer mitigates the number of parameters when there are larger size images. This can be called as subsampling or down sampling that mitigates the dimensionality of every feature map by preserving the important information. Max pooling considers the maximum element form the rectified feature map.

**4.4 Random Forest Algorithm**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Diagram

Description automatically generated

Fig. 4.4: Random Forest algorithm.

***Random Forest algorithm***

Step 1: In Random Forest n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

**Important Features of Random Forest**

* **Diversity**- Not all attributes/variables/features are considered while making an individual tree, each tree is different.
* **Immune** **to** **the** **curse** **of** **dimensionality**- Since each tree does not consider all the features, the feature space is reduced.
* **Parallelization**-Each tree is created independently out of different data and attributes. This means that we can make full use of the CPU to build random forests.
* **Train-Test** **split**- In a random forest we don’t have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.
* **Stability**- Stability arises because the result is based on majority voting/ averaging.

**Assumptions for Random Forest**

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random Forest classifier:

* There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
* The predictions from each tree must have very low correlations.

Below are some points that explain why we should use the Random Forest algorithm

* It takes less training time as compared to other algorithms.
* It predicts output with high accuracy, even for the large dataset it runs efficiently.
* It can also maintain accuracy when a large proportion of data is missing.

**Types of Ensembles**

Before understanding the working of the random forest, we must look into the ensemble technique. Ensemble simply means combining multiple models. Thus, a collection of models is used to make predictions rather than an individual model. Ensemble uses two types of methods:

**Bagging**– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest. Bagging, also known as Bootstrap Aggregation is the ensemble technique used by random forest. Bagging chooses a random sample from the data set. Hence each model is generated from the samples (Bootstrap Samples) provided by the Original Data with replacement known as row sampling. This step of row sampling with replacement is called bootstrap. Now each model is trained independently which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as aggregation.

**Diagram

Description automatically generated**

Fig. 4.5: RF Classifier analysis.

**Boosting**– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST.

Diagram

Description automatically generated

Fig. 4.6: Boosting RF Classifier.

**Advantages of proposed system**

* It can be used in classification and regression problems.
* It solves the problem of overfitting as output is based on majority voting or averaging.
* It performs well even if the data contains null/missing values.
* Each decision tree created is independent of the other thus it shows the property of parallelization.

**CHAPTER 5**

**UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

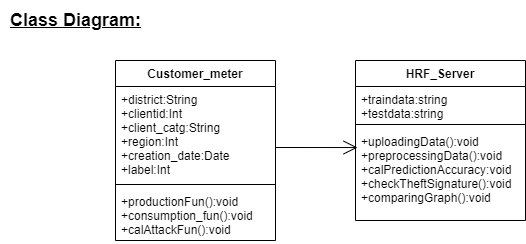
**Goals**

The Primary goals in the design of the UML are as follows:

* Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
* Provide extendibility and specialization mechanisms to extend the core concepts.
* Be independent of particular programming languages and development process.
* Provide a formal basis for understanding the modeling language.
* Encourage the growth of OO tools market.
* Support higher level development concepts such as collaborations, frameworks, patterns and components.
* Integrate best practices.

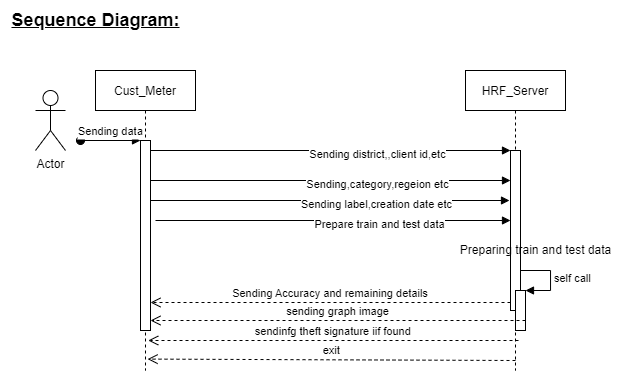
**5.1 Class** **Diagram**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



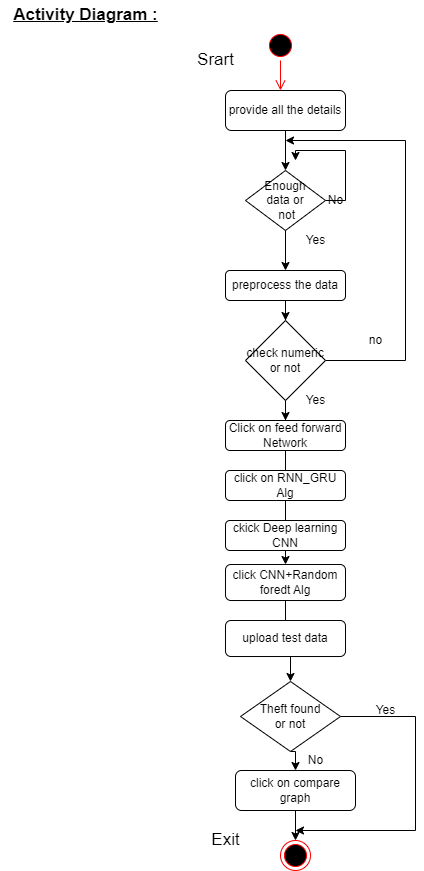
**5.2 Sequence diagram**

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".



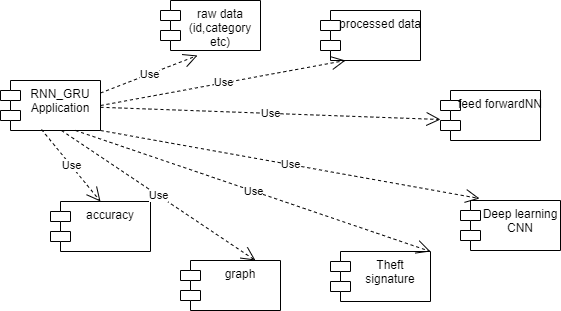
**5.3 Activity diagram**

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions.



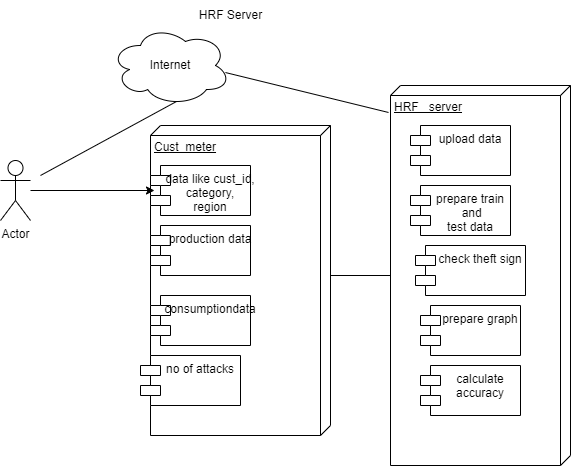
**5,4 Collaboration diagram**

A collaboration diagram groups together the interactions between different objects. The interactions are listed as numbered interactions that help to trace the sequence of the interactions. The collaboration diagram helps to identify all the possible interactions that each object has with other objects.



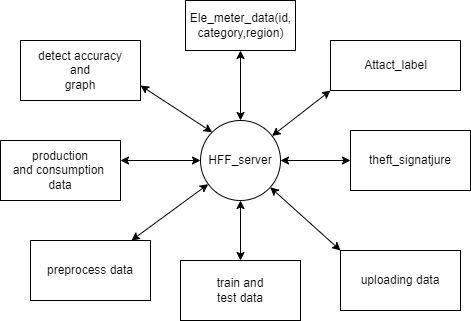
**5.5 Deployment diagram**

The deployment diagram visualizes the physical hardware on which the software will be deployed.

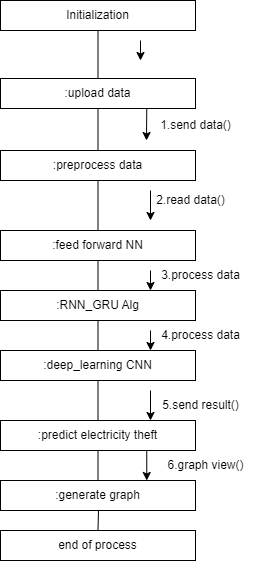


**5.6 Dataflow diagram**

Data flow diagram is graphical representation of flow of data in an information system.



**5.7 Component diagram:** describes the organization and wiring of the physical components in a system.



**CHAPTER 6**

**SOFTWARE ENVIRONMENT**

**What is Python?**

Below are some facts about Python.

* Python is currently the most widely used multi-purpose, high-level programming language.
* Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
* Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
* Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard libraries which can be used for the following –

* 1. Machine Learning
  2. GUI Applications (like Kivy, Tkinter, PyQt etc.)
  3. Web frameworks like Django (used by YouTube, Instagram, Dropbox)
  4. Image processing (like Opencv, Pillow)
  5. Web scraping (like Scrapy, BeautifulSoup, Selenium)
  6. Test frameworks
  7. Multimedia

**Advantages of Python**

Let’s see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don’t have to write the complete code for that manually.

1. Extensible

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

1. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

1. Improved Productivity

The language’s simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

1. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet of Things. This is a way to connect the language with the real world.

1. Simple and Easy

When working with Java, you may have to create a class to print ‘Hello World’. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

1. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. These further aids the readability of the code.

1. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

1. Free and Open-Source

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

1. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn’t the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

1. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

**Advantages of Python Over Other Languages**

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don’t have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

1. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

1. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

**Disadvantages of Python**

So far, we’ve seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let’s now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn’t a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle.

The reason it is not so famous despite the existence of Brython is that it isn’t that secure.

3. Design Restrictions

As you know, Python is dynamically-typed. This means that you don’t need to declare the type of variable while writing the code. It uses duck-typing. But wait, what’s that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java DataBase Connectivity) and ODBC (Open DataBase Connectivity), Python’s database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we’re not kidding. Python’s simplicity can indeed be a problem. Take my example. I don’t do Java, I’m more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

**History of Python**

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde &Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners1, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it. "Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So, I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

**Python Development Steps**

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of lists, dict, str and others. It was also object oriented and had a module system.  
Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it."Some changes in Python 7.3:

* Print is now a function.
* Views and iterators instead of lists
* The rules for ordering comparisons have been simplified. E.g., a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
* There is only one integer type left, i.e., int. long is int as well.
* The division of two integers returns a float instead of an integer. "//" can be used to have the "old" behaviour.
* Text Vs. Data Instead of Unicode Vs. 8-bit

**Purpose**

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

**Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

* Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

**Modules Used in Project**

**TensorFlow**

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.‍

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

**NumPy**

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

* A powerful N-dimensional array object
* Sophisticated (broadcasting) functions
* Tools for integrating C/C++ and Fortran code
* Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

**Pandas**

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

**Matplotlib**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object-oriented interface or via a set of functions familiar to MATLAB users.

**Scikit – learn**

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

* Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

**Install Python Step-by-Step in Windows and Mac**

Python a versatile programming language doesn’t come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

**How to Install Python on Windows and Mac**

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e., operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So, the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here. The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

**Download the Correct version into the system**

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: https://www.python.org

A screenshot of a computer

Description automatically generated with medium confidence

Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.

Graphical user interface, application

Description automatically generated

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

Graphical user interface, application

Description automatically generated

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

Graphical user interface, text

Description automatically generated

* To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
* To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e., Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.

Graphical user interface, text, application

Description automatically generated

Step 2: Before you click on Install Now, make sure to put a tick on Add Python 3.7 to PATH.

Graphical user interface, text, application, chat or text message

Description automatically generated

Step 3: Click on Install NOW After the installation is successful. Click on Close.

Graphical user interface, text, application, chat or text message

Description automatically generated

With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.

Graphical user interface, application

Description automatically generated

Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type python –V and press Enter.

A screenshot of a computer

Description automatically generated with medium confidence

Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.

Application

Description automatically generated with low confidence

Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save

Graphical user interface, text, application, email

Description automatically generated

Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g., enter print (“Hey World”) and Press Enter.

Graphical user interface, text, application, email

Description automatically generated

You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python does not need semicolons at the end of the statements otherwise it won’t work.

**CHAPTER 7**

**SYSTEM REQUIREMENTS SPECIFICATIONS**

**Software Requirements**

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regard to what the areas of strength and deficit are and how to tackle them.

* Python IDLE 3.7 version (or)
* Anaconda 3.7 (or)
* Jupiter (or)
* Google colab

**Hardware Requirements**

Majormum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

Operating system : Windows, Linux

Processor : majormum intel i3

Ram : majormum 4 GB

Hard disk : majormum 250GB

**CHAPTER 8**

**FUNCTIONAL REQUIREMENTS**

**Output Design**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation. The various types of outputs in general are:

* External Outputs, whose destination is outside the organization
* Internal Outputs whose destination is within organization and they are the
* User’s main interface with the computer.
* Operational outputs whose use is purely within the computer department.
* Interface outputs, which involve the user in communicating directly.

**Output Definition**

The outputs should be defined in terms of the following points:

* Type of the output
* Content of the output
* Format of the output
* Location of the output
* Frequency of the output
* Volume of the output
* Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

**Input Design**

Input design is a part of overall system design. The main objective during the input design is as given below:

* To produce a cost-effective method of input.
* To achieve the highest possible level of accuracy.
* To ensure that the input is acceptable and understood by the user.

**Input Stages**

The main input stages can be listed as below:

* Data recording
* Data transcription
* Data conversion
* Data verification
* Data control
* Data transmission
* Data validation
* Data correction

**Input Types**

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

* External inputs, which are prime inputs for the system.
* Internal inputs, which are user communications with the system.
* Operational, which are computer department’s communications to the system?
* Interactive, which are inputs entered during a dialogue.

**Input Media**

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

* Type of input
* Flexibility of format
* Speed
* Accuracy
* Verification methods
* Rejection rates
* Ease of correction
* Storage and handling requirements
* Security
* Easy to use
* Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As

Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

**Error Avoidance**

At this stage care is to be taken to ensure that input data remains accurate form th stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

**Error Detection**

Even though every effort is made to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

**Data Validation**

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

**User Interface Design**

It is essential to consult the system users and discuss their needs while designing the user interface:

**User Interface Systems Can Be Broadly Clasified As:**

* User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
* Computer initiated interfaces

In the computer-initiated interfaces the computer guides the progress of the user/computer dialogue. Information is displayed and the user response of the computer takes action or displays further information.

**User Initiated Intergfaces**

User initiated interfaces fall into two approximate classes:

* Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
* Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms-oriented interface is chosen because it is the best choice.

**Computer-Initiated Interfaces**

The following computer – initiated interfaces were used:

* The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.
* Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

**Error Message Design**

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different inputs.

**Performance Requirements**

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

* The system should be able to interface with the existing system

**CHAPTER 9**

**SOURCE CODE**

from tkinter import \*

import tkinter

from tkinter import filedialog

import numpy as np

from tkinter.filedialog import askopenfilename

import pandas as pd

from tkinter import simpledialog

import pandas as pd

import numpy as np

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import normalize

from keras.models import Sequential, Model

from keras.layers import Dense, Dropout, Activation

from keras.utils.np\_utils import to\_categorical

from keras.models import model\_from\_json

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import precision\_score

from sklearn.metrics import recall\_score

from sklearn.metrics import f1\_score

from sklearn.metrics import accuracy\_score

from sklearn.model\_selection import train\_test\_split

from sklearn import svm

import os

import matplotlib.pyplot as plt

from sklearn.metrics import roc\_curve

from sklearn.metrics import roc\_auc\_score

from sklearn import metrics

from keras.layers import MaxPooling2D

from keras.layers import Flatten

from keras.layers import Convolution2D

from keras.layers import Bidirectional,GRU

main = tkinter.Tk()

main.title("Deep Ensemble Framework with Supervised Learning for Secure IoT Network")

main.geometry("1000x650")

global filename

global dnn\_model

global X, Y

global le

global dataset

accuracy = []

precision = []

recall = []

fscore = []

global classifier

global cnn\_model

def uploadDataset():

global filename

global dataset

filename = filedialog.askopenfilename(initialdir = "Dataset")

text.delete('1.0', END)

text.insert(END,filename+' Loaded\n')

dataset = pd.read\_csv(filename)

text.insert(END,str(dataset.head())+"\n\n")

def preprocessDataset():

global X, Y

global le

global dataset

le = LabelEncoder()

text.delete('1.0', END)

dataset.fillna(0, inplace = True)

dataset['client\_id'] = pd.Series(le.fit\_transform(dataset['client\_id'].astype(str)))

dataset['label'] = dataset['label'].astype('uint8')

print(dataset.info())

dataset.drop(['creation\_date'], axis = 1,inplace=True)

text.insert(END,str(dataset.head())+"\n\n")

dataset = dataset.values

X = dataset[:,0:dataset.shape[1]-1]

Y = dataset[:,dataset.shape[1]-1]

Y = Y.astype('uint8')

indices = np.arange(X.shape[0])

np.random.shuffle(indices)

X = X[indices]

Y = Y[indices]

Y = Y.astype('uint8')

text.insert(END,"Total records found in dataset to train Deep Learning : "+str(X.shape[0])+"\n\n")

def rocGraph(testY, predict, algorithm):

random\_probs = [0 for i in range(len(testY))]

p\_fpr, p\_tpr, \_ = roc\_curve(testY, random\_probs, pos\_label=1)

plt.plot(p\_fpr, p\_tpr, linestyle='--', color='orange',label="True classes")

ns\_fpr, ns\_tpr, \_ = roc\_curve(testY, predict,pos\_label=1)

plt.plot(ns\_fpr, ns\_tpr, linestyle='--', label='Predicted Classes')

plt.title(algorithm+" ROC Graph")

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive rate')

plt.show()

def runCNN():

global X, Y

Y1 = to\_categorical(Y)

Y1 = Y1.astype('uint8')

X1 = np.reshape(X, (X.shape[0], X.shape[1], 1, 1))

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X1, Y1, test\_size=0.2, random\_state=0)

global cnn\_model

if os.path.exists('model/cnn\_model.json'):

with open('model/cnn\_model.json', "r") as json\_file:

loaded\_model\_json = json\_file.read()

cnn\_model = model\_from\_json(loaded\_model\_json)

json\_file.close()

cnn\_model.load\_weights("model/cnn\_model\_weights.h5")

cnn\_model.\_make\_predict\_function()

else:

counts = np.bincount(Y1[:, 0])

weight\_for\_0 = 1.0 / counts[0]

weight\_for\_1 = 1.0 / counts[1]

class\_weight = {0: weight\_for\_0, 1: weight\_for\_1}

cnn\_model = Sequential()

cnn\_model.add(Convolution2D(32, 1, 1, input\_shape = (X\_train.shape[1], X\_train.shape[2], X\_train.shape[3]), activation = 'relu'))

cnn\_model.add(MaxPooling2D(pool\_size = (1, 1)))

cnn\_model.add(Convolution2D(32, 1, 1, activation = 'relu'))

cnn\_model.add(MaxPooling2D(pool\_size = (1, 1)))

cnn\_model.add(Flatten())

cnn\_model.add(Dense(output\_dim = 256, activation = 'relu'))

cnn\_model.add(Dense(output\_dim = y\_train.shape[1], activation = 'softmax'))

cnn\_model.compile(optimizer = 'adam', loss = 'categorical\_crossentropy', metrics = ['accuracy'])

hist = cnn\_model.fit(X\_train, y\_train, batch\_size=64, epochs=20, shuffle=True, verbose=2, validation\_data=(X\_test, y\_test),class\_weight=class\_weight)

cnn\_model.save\_weights('model/cnn\_model\_weights.h5')

model\_json = cnn\_model.to\_json()

with open("model/cnn\_model.json", "w") as json\_file:

json\_file.write(model\_json)

json\_file.close()

y\_test = np.argmax(y\_test, axis=1)

predict = cnn\_model.predict(X\_test)

predict = np.argmax(predict, axis=1)

p = precision\_score(y\_test, predict,average='macro') \* 100

r = recall\_score(y\_test, predict,average='macro') \* 100

f = f1\_score(y\_test, predict,average='macro') \* 100

a = accuracy\_score(y\_test,predict)\*100

accuracy.append(a)

precision.append(p)

recall.append(r)

fscore.append(f)

text.insert(END,"CNN Precision : "+str(p)+"\n")

text.insert(END,"CNN Recall : "+str(r)+"\n")

text.insert(END,"CNN FMeasure : "+str(f)+"\n")

text.insert(END,"CNN Accuracy : "+str(f)+"\n\n")

rocGraph(y\_test, predict, "CNN")

def runDNN():

text.delete('1.0', END)

global X, Y

global dnn\_model

accuracy.clear()

precision.clear()

recall.clear()

fscore.clear()

Y1 = to\_categorical(Y)

Y1 = Y1.astype('uint8')

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y1, test\_size=0.2, random\_state=0)

if os.path.exists('model/model.json'):

with open('model/model.json', "r") as json\_file:

loaded\_model\_json = json\_file.read()

dnn\_model = model\_from\_json(loaded\_model\_json)

json\_file.close()

dnn\_model.load\_weights("model/model\_weights.h5")

dnn\_model.\_make\_predict\_function()

print(dnn\_model.summary())

else:

counts = np.bincount(Y1[:, 0])

weight\_for\_0 = 1.0 / counts[0]

weight\_for\_1 = 1.0 / counts[1]

class\_weight = {0: weight\_for\_0, 1: weight\_for\_1}

dnn\_model = Sequential() #creating RNN model object

dnn\_model.add(Dense(256, input\_dim=X.shape[1], activation='relu', kernel\_initializer = "uniform")) #defining one layer with 256 filters to filter dataset

dnn\_model.add(Dense(128, activation='relu', kernel\_initializer = "uniform"))#defining another layer to filter dataset with 128 layers

dnn\_model.add(Dense(y\_train.shape[1], activation='softmax',kernel\_initializer = "uniform")) #after building model need to predict two classes such as normal or Dyslipidemia disease

dnn\_model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy']) #while filtering and training dataset need to display accuracy

print(dnn\_model.summary()) #display rnn details

hist = cnn\_model.fit(X\_train, y\_train, epochs=20, batch\_size=64,class\_weight=class\_weight)

dnn\_model.save\_weights('model/model\_weights.h5')

model\_json = dnn\_model.to\_json()

with open("model/model.json", "w") as json\_file:

json\_file.write(model\_json)

json\_file.close()

y\_test = np.argmax(y\_test, axis=1)

predict = dnn\_model.predict(X\_test)

predict = np.argmax(predict, axis=1)

p = precision\_score(y\_test, predict,average='macro') \* 100

r = recall\_score(y\_test, predict,average='macro') \* 100

f = f1\_score(y\_test, predict,average='macro') \* 100

a = accuracy\_score(y\_test,predict)\*100

accuracy.append(a)

precision.append(p)

recall.append(r)

fscore.append(f)

text.insert(END,"DNN Precision : "+str(p)+"\n")

text.insert(END,"DNN Recall : "+str(r)+"\n")

text.insert(END,"DNN FMeasure : "+str(f)+"\n")

text.insert(END,"DNN Accuracy : "+str(f)+"\n\n")

rocGraph(y\_test, predict, "DNN")

def runCNNRandomForest():

global cnn\_model

global classifier

global X, Y

global cnn\_model

print(cnn\_model.summary())

X1 = np.reshape(X, (X.shape[0], X.shape[1], 1, 1))

extract = Model(cnn\_model.inputs, cnn\_model.layers[-2].output)

XX = extract.predict(X1)

print(XX.shape)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(XX, Y, test\_size=0.2, random\_state=0)

X\_train, X\_test1, y\_train, y\_test1 = train\_test\_split(X\_test, y\_test, test\_size=0.2, random\_state=0)

rfc = RandomForestClassifier(n\_estimators=50)

rfc.fit(X\_test, y\_test)

classifier = rfc

predict = rfc.predict(X\_test1)

p = precision\_score(y\_test1, predict,average='macro') \* 100

r = recall\_score(y\_test1, predict,average='macro') \* 100

f = f1\_score(y\_test1, predict,average='macro') \* 100

a = accuracy\_score(y\_test1,predict)\*100

accuracy.append(a)

precision.append(p)

recall.append(r)

fscore.append(f)

text.insert(END,"CNN with Random Forest Precision : "+str(p)+"\n")

text.insert(END,"CNN with Random Forest Recall : "+str(r)+"\n")

text.insert(END,"CNN with Random Forest FMeasure : "+str(f)+"\n")

text.insert(END,"CNN with Random Forest Accuracy : "+str(f)+"\n\n")

rocGraph(y\_test1, predict, "CNN + Random Forest")

def predict():

global classifier

global cnn\_model

text.delete('1.0', END)

filename = filedialog.askopenfilename(initialdir = "Dataset")

test = pd.read\_csv(filename)

test.fillna(0, inplace = True)

test = test.values

data = test

test = np.reshape(test, (test.shape[0], test.shape[1], 1, 1))

extract = Model(cnn\_model.inputs, cnn\_model.layers[-2].output)

test = extract.predict(test)

predict = classifier.predict(test)

for i in range(len(predict)):

if predict[i] == 1:

text.insert(END,str(data[i])+" ===> Record Detected as Cyber Attack\n\n")

if predict[i] == 0:

text.insert(END,str(data[i])+" ===> Record Not Detected as Cyber Attack\n\n")

def graph():

df = pd.DataFrame([['DNN','Precision',precision[0]],['DNN','Recall',recall[0]],['DNN','F1 Score',fscore[0]],['DNN','Accuracy',accuracy[0]],

['CNN','Precision',precision[1]],['CNN','Recall',recall[1]],['CNN','F1 Score',fscore[1]],['CNN','Accuracy',accuracy[1]],

['CNN+RF','Precision',precision[2]],['CNN+RF','Recall',recall[2]],['CNN+RF','F1 Score',fscore[2]],['CNN+RF','Accuracy',accuracy[2]],

],columns=['Parameters','Algorithms','Value'])

df.pivot("Parameters", "Algorithms", "Value").plot(kind='bar')

plt.show()

def close():

main.destroy()

font = ('times', 16, 'bold')

title = Label(main, text='Deep Ensemble Framework with Supervised Learning for Secure IoT Network', justify=LEFT)

title.config(bg='AliceBlue', fg='black')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=100,y=5)

title.pack()

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Dataset", command=uploadDataset)

uploadButton.place(x=20,y=100)

uploadButton.config(font=font1)

preprocessButton = Button(main, text="Preprocess Dataset", command=preprocessDataset)

preprocessButton.place(x=200,y=100)

preprocessButton.config(font=font1)

cnnButton = Button(main, text="Feed Forward Neural Network", command=runDNN)

cnnButton.place(x=400,y=100)

cnnButton.config(font=font1)

cnnsvmButton = Button(main, text="Deep Learning CNN", command=runCNN)

cnnsvmButton.place(x=700,y=100)

cnnsvmButton.config(font=font1)

rfButton = Button(main, text="CNN + Random Forest", command=runCNNRandomForest)

rfButton.place(x=20,y=150)

rfButton.config(font=font1)

predictButton = Button(main, text="Predict Cyber Attack", command=predict)

predictButton.place(x=250,y=150)

predictButton.config(font=font1)

graphButton = Button(main, text="Performance Evaluation", command=graph)

graphButton.place(x=450,y=150)

graphButton.config(font=font1)

exitButton = Button(main, text="Exit", command=close)

exitButton.place(x=700,y=150)

exitButton.config(font=font1)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=120)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=200)

text.config(font=font1)

main.config(bg='LightSteelBlue4')

main.mainloop()

**CHAPTER 10**

**RESULTS AND DISCUSSION**

**10.1 Implementation Description**

The Python code implements a GUI application using the Tkinter library to facilitate the Deep Ensemble Framework with Supervised Learning for Secure IoT Network.

* GUI Setup

The code initializes the main GUI window with a title and dimensions.

* Buttons and Functionality:

Upload Dataset: This button triggers a file dialog allowing the user to select a dataset file. Upon selection, the code loads the dataset, displays its first few rows in the text area, and stores it for further processing.

* Preprocess Dataset:

This button preprocesses the loaded dataset by handling missing values, encoding categorical variables, and splitting it into features and labels.

* Feed Forward Neural Network (DNN)

Initiates the training and evaluation of a DNN model on the preprocessed dataset. The model's performance metrics such as accuracy, precision, recall, and F1 score are displayed in the text area.

* Deep Learning CNN

Triggers the training and evaluation of a Convolutional Neural Network (CNN) model on the preprocessed dataset. Similar to the DNN, the CNN's performance metrics are displayed.

* CNN + Random Forest

Combines the features extracted from the CNN model with a Random Forest classifier for training and evaluation. Performance metrics are displayed accordingly.

* Predict Cyber Attack:

Allows the user to upload a new dataset file for prediction. The code preprocesses the data, extracts features using the CNN model, and predicts whether each record represents a cyber-attack or not.

* Performance Evaluation:

Generates a bar graph visualizing the performance evaluation metrics (accuracy, precision, recall, and F1 score) of different algorithms (DNN, CNN, CNN + Random Forest).

* Exit: Closes the GUI application.
* Text Area:

A text area is provided within the GUI to display messages, dataset information, model performance metrics, and prediction results.

* Model Training and Evaluation

The code trains and evaluates various machine learning models (DNN, CNN, CNN + Random Forest) using the preprocessed dataset. Performance metrics such as accuracy, precision, recall, and F1 score are calculated and displayed for each model.

* Visualization

The code utilizes matplotlib to generate visualizations such as ROC graphs and bar graphs to aid in the interpretation of model performance.

* Integration with Deep Learning Libraries

The implementation leverages deep learning libraries such as Keras for model building, training, and evaluation.

* Error Handling: While not explicitly shown in the provided code snippet, it's common practice to include error handling mechanisms to gracefully handle exceptions and errors that may arise during dataset loading, preprocessing, or model training.

**10.2 Dataset Description**

The dataset consists of several columns representing different attributes of clients in an IoT network. Here's a description of each column based on the provided title:

district: This column likely represents the district or location associated with each client in the IoT network. It appears to be a categorical variable indicating the geographical area or jurisdiction where the client is located.

client\_id: This column contains unique identifiers assigned to each client in the IoT network. It serves as a primary key or identifier for distinguishing between individual clients.

client\_catg: This column represents the category or type of client within the IoT network. It appears to be a categorical variable categorizing clients based on certain criteria or attributes, such as their usage patterns, industry sector, or subscription level.

region: This column indicates the region or area within which each client operates or is located. Similar to the 'district' column, it likely represents a geographical classification of clients based on their geographic location or operational area.

creation\_date: This column contains the creation date or registration date associated with each client in the IoT network. It represents the date when each client was onboarded or registered within the system.

label: This column serves as the target variable or label for the dataset. It indicates whether each client has been labeled as a potential security threat or not. In the provided dataset, a value of '0' likely signifies that the client is not flagged as a security threat, while other values may indicate otherwise.

**10.3 Results and Description**

To implement this project, we have used Smart Meter electricity recording dataset and below are the details of that dataset

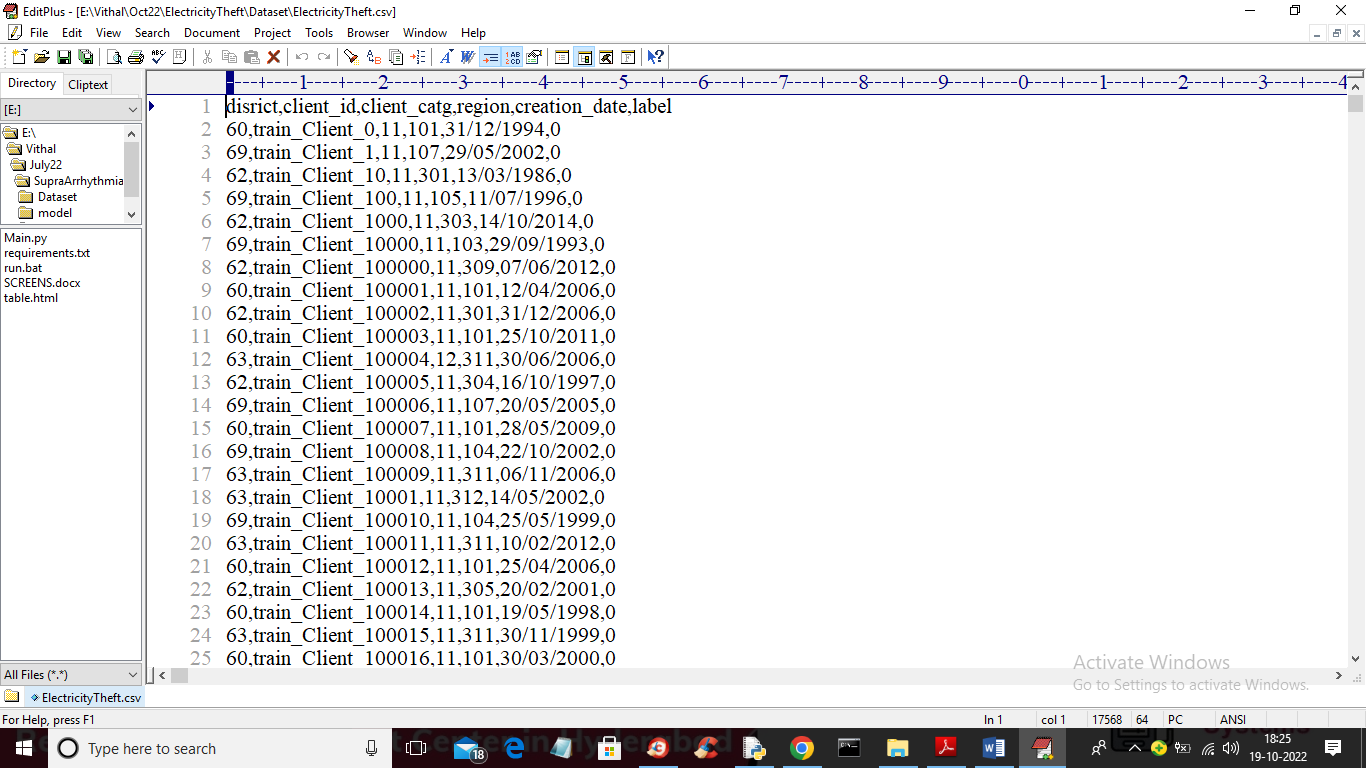


Fig 10.1 Perform a Dataset

In above screen first row represents dataset column names and remaining rows contains dataset values which contains IoT device details and last column contains class label as 0 or 1 where 0 means No Attack and 1 means Attack.

To implement this project, we have designed following modules

1. Upload Dataset: using this module we will upload dataset to application
2. Preprocess Dataset: using this module we will read dataset and then remove missing values and then convert all non-numeric data into numeric as deep learning accept only numeric data. Processed dataset will be split into train and test where 80% dataset used for training and 20% for testing
3. Feed Forward Neural Network: processed train data will be input to DNN algorithm to train attack detection model and this model will be applied on test data to calculate prediction accuracy.
4. Deep Learning CNN: processed train data will be input to CNN algorithm to train cyberattack detection model and this model will be applied on test data to calculate prediction accuracy.
5. CNN + Random Forest: using this module we will extract features from CNN and then retrain with Random Forest algorithm to build a hybrid model and then test data will be applied on hybrid model to calculate its accuracy
6. Predict Cyberattack: using this module we will upload test data and then proposed algorithm will predict weather test data is normal or contains attack signatures
7. Performance Evaluation: using this module we will plot comparison graph of all algorithms

**SCREEN SHOTS**

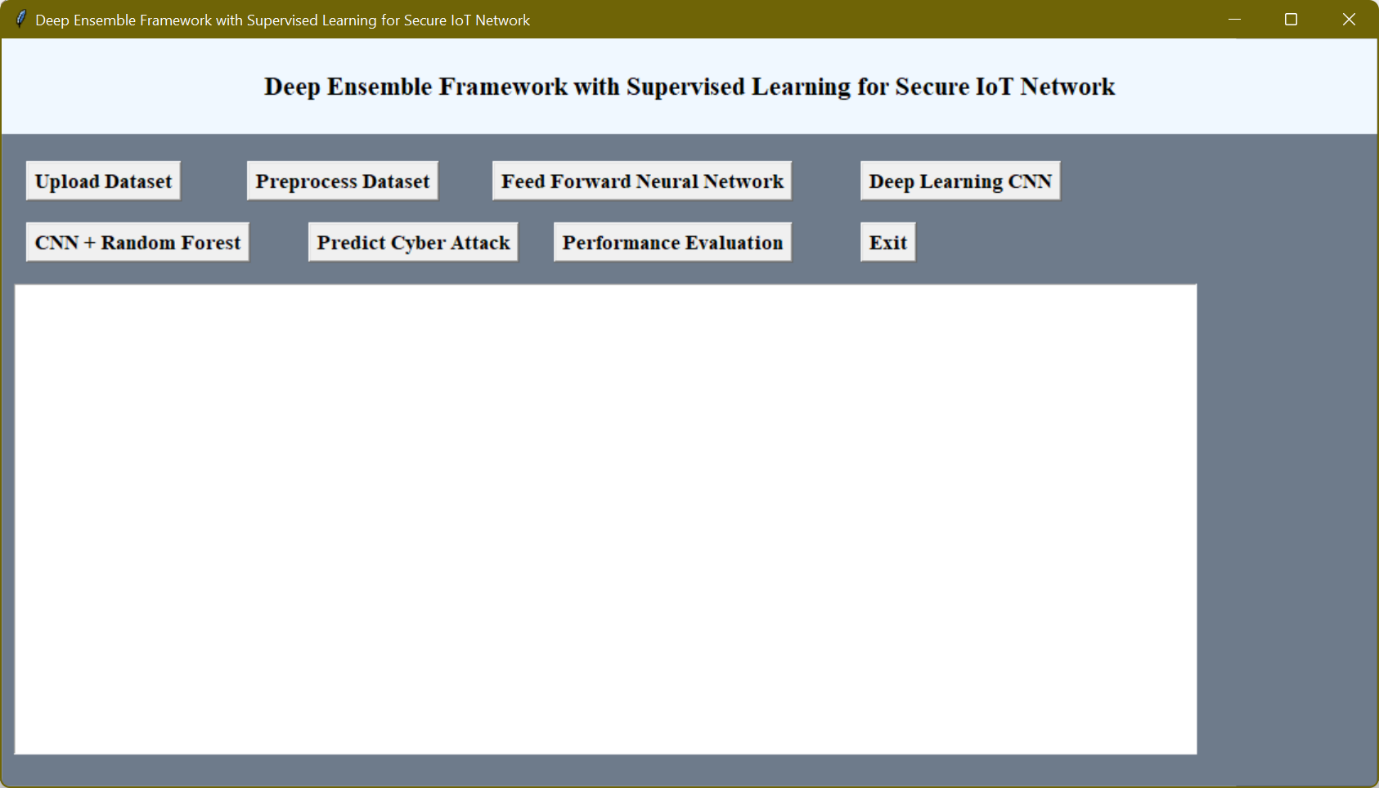


Fig 10.2 Upload Dataset Description

In above screen click on ‘Upload Dataset’ button to upload IoT device dataset and get below output

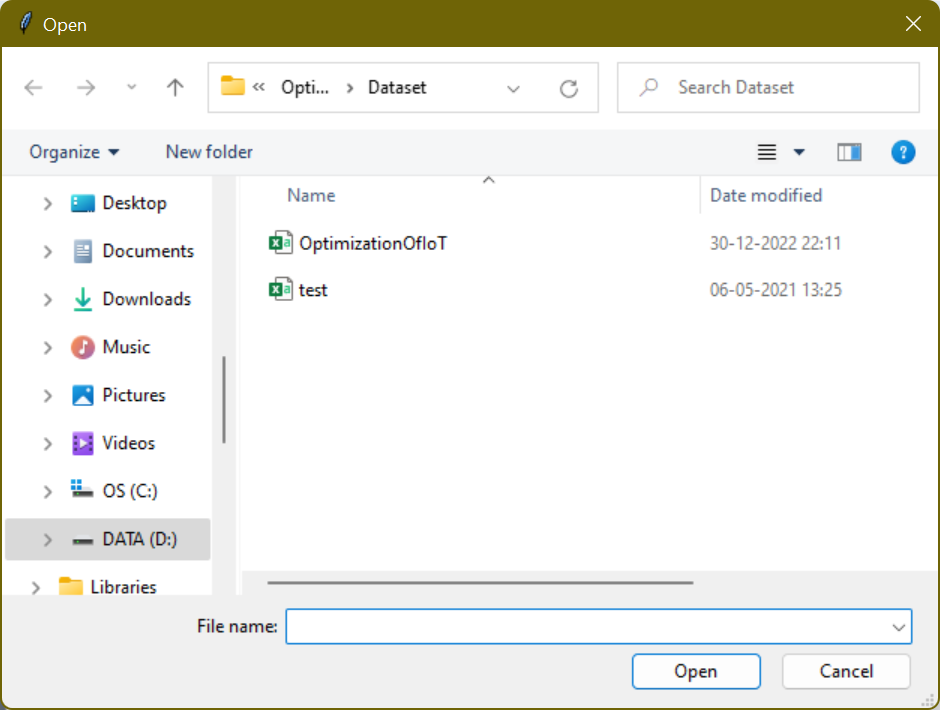


Fig 10.3 dataset of Optimization Of IoT

In above screen selecting and uploading ‘OptimizationOfIoT’ dataset and then click on ‘Open’ button to load dataset and get below output

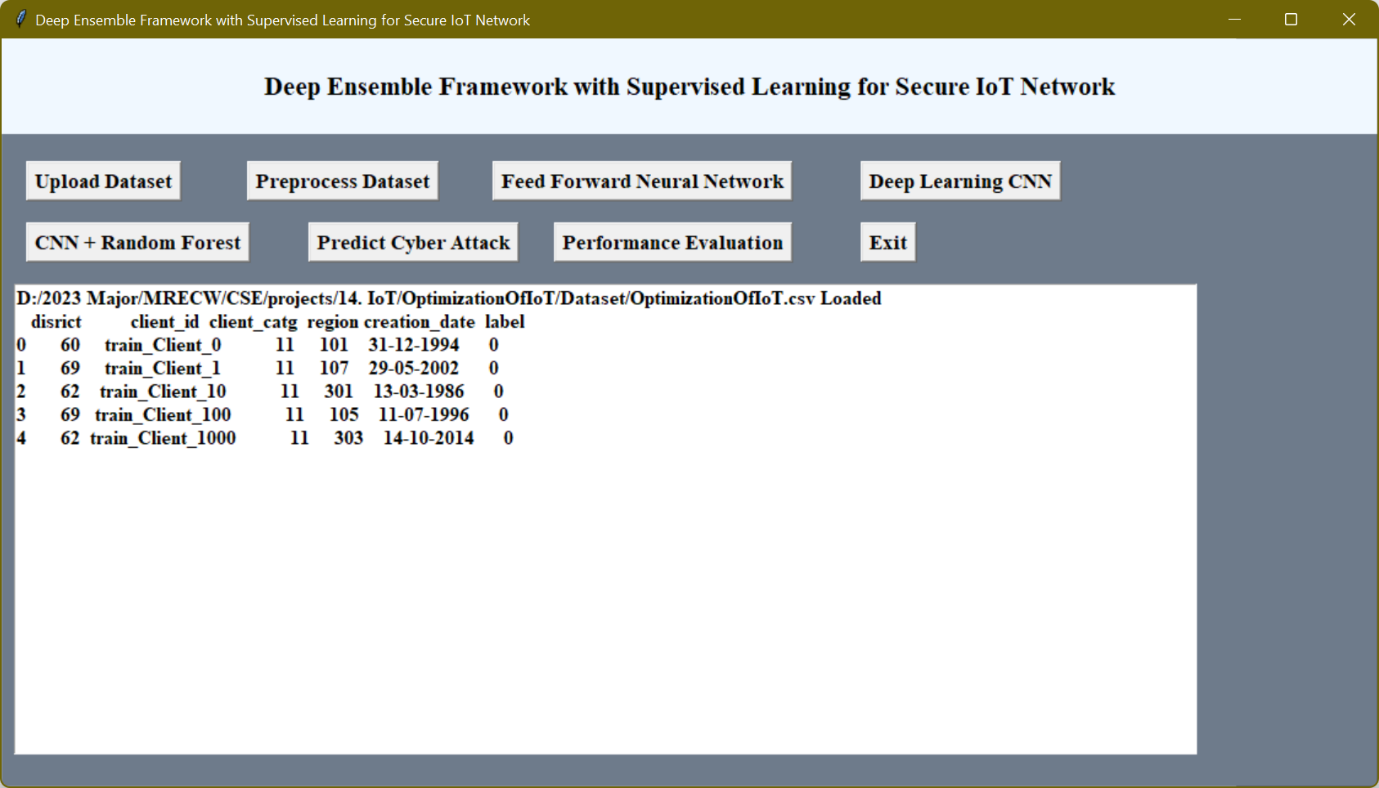


Fig 10.4 Perform the Preprocessing dataset of Iot

In above screen dataset loaded and now click on ‘Preprocess Dataset’ button to clean dataset and get below output

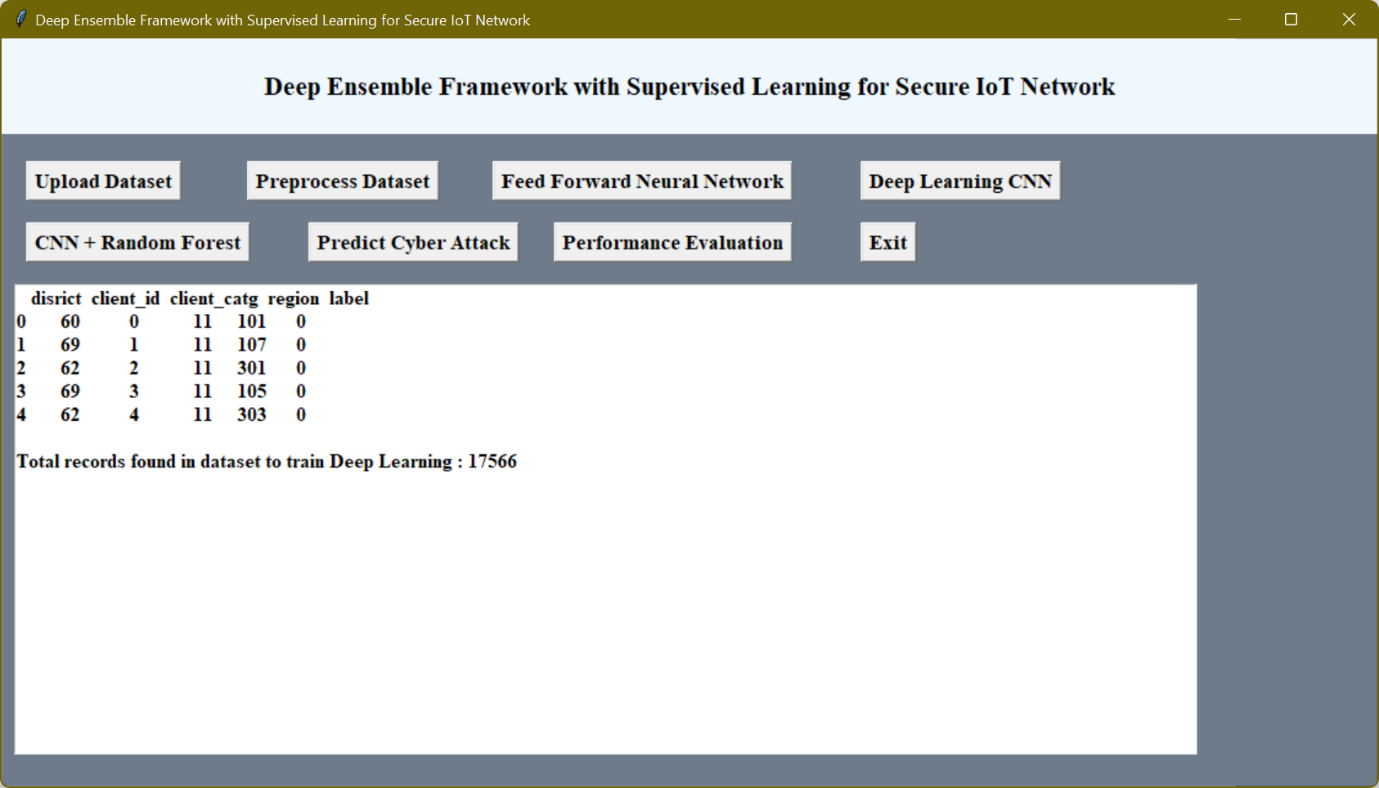


Fig 10.5 perform Of feed Forward neural Network

In above screen all non-numeric data converted to numeric format and now click on ‘Feed Forward Neural Network’ button to train DNN and get below output

Graphical user interface, text

Description automatically generated

Fig 10.6 Perform of Deep Learing CNN

Chart, line chart

Description automatically generated

Fig 10.7 DNN ROC Graph

In above screen with DNN feed forward algorithm we got 93.99% accuracy and in ROC graph x-graph represents False Positive Rate and y-axis

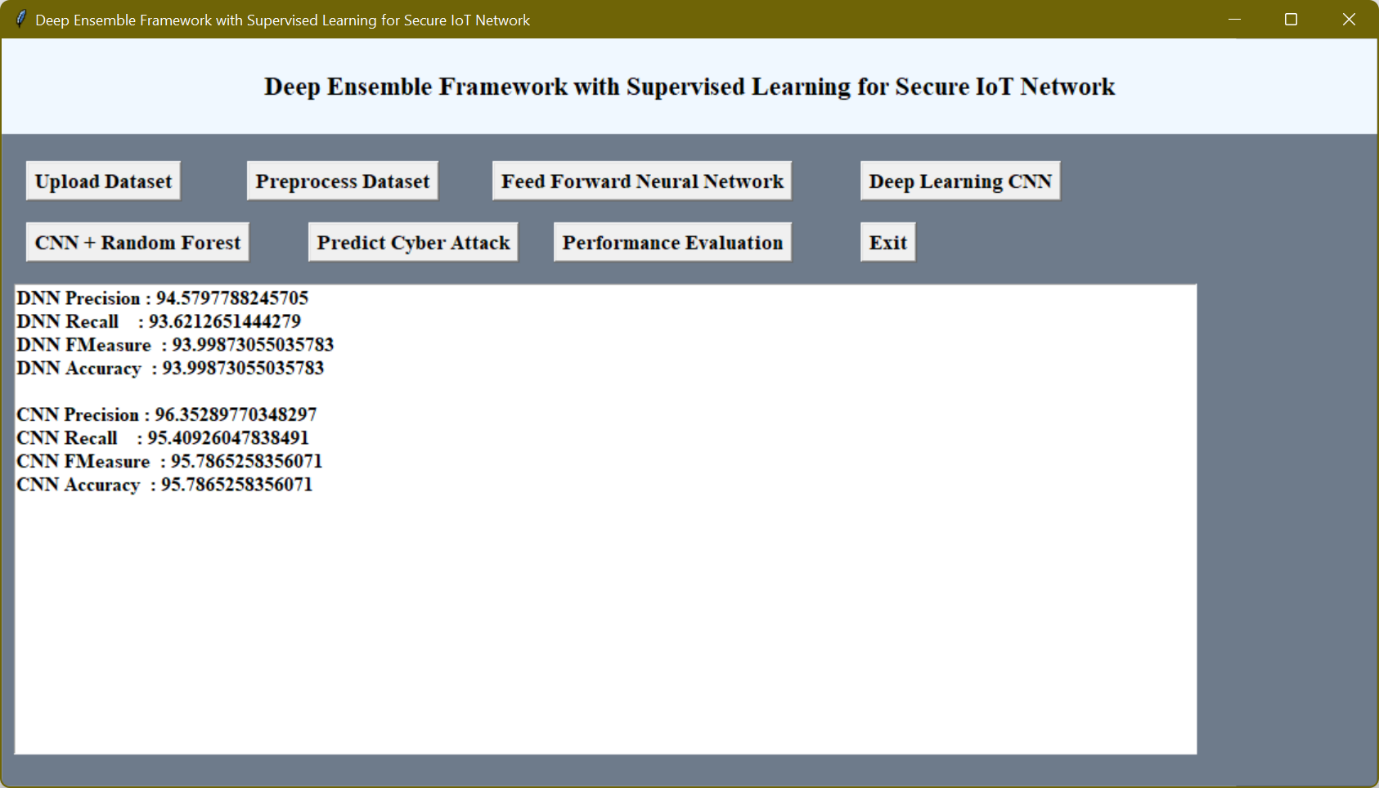


Fig 10.8 Optimization Of Iot Perform Evolution

Chart, bar chart

Description automatically generated

Fig 10.9 Performing The Optimizatin Of Iot

In above graph x-axis represents algorithm names with each different colour bar represents different metric such as ‘accuracy, precision, recall and FSCORE’ and Y-axis represents score values. In all algorithms proposed optimized ensemble framework got high performance.

**CHAPTER 11**

**CONCLUSION**

Global energy crises are increasing every moment. Everyone has the attention towards more and more energy production and also trying to save it. Electricity can be produced through many ways which is then synchronized on a main grid for usage. Weather losses are technical or non-technical. Technical losses can abstract be calculated easily, as we discussed in section of mathematical modeling that how to calculate technical losses. Whereas nontechnical losses can be evaluated if technical losses are known. Theft in electricity produce non-technical losses. To reduce or control theft one can save his economic resources. Smart meter can be the best option to majormize electricity theft, because of its high security, best efficiency, and excellent resistance towards many of theft ideas in electromechanical meters. So, in this paper we have mostly concentrated on theft issues. Therefore, this project evaluated performance of various deep learning algorithms such as deep feed forward neural network (DNN), recurrent neural network with gated recurrent unit (RNN-GRU) and convolutional neural network (CNN) for electricity cyber-attack detection.

**REFERENCES**

|  |  |  |  |
| --- | --- | --- | --- |
|  | [1] Jahangeer, Asma, Et Al. "A Review On The Security Of Iot Network From Network Layer’s Perspective." IEEE Access (2023)  [2] Goulart, Ana, Et Al. "On Wide-Area Iot Networks, Lightweight Security And Their Applications—A Practical Review." Electronics 11.11 (2022):  1762.  [3] Mahamat, Michaël, Ghada Jaber, And Abdelmadjid Bouabdallah. "Achieving Efficient Energy-Aware Security In Iot Networks: A Survey Of Recent Solutions And Research Challenges." Wireless Networks 29.2 (2023): 787-808.  [4] Aldhaheri, Alyazia, Et Al. "Deep Learning For Cyber Threat Detection In Iot Networks: A Review." Internet Of Things And Cyber-Physical Systems (2023).  [5] Cai, J., Liang, W., Li, X., Li, K., Gui, Z., & Khan, M. K. (2023). Gtxchain: A Secure Iot Smart Blockchain Architecture Based On Graph Neural Network. IEEE Internet Of Things Journal.  [6] Tibrewal, Ishani, Manas Srivastava, And Amit Kumar Tyagi. "Blockchain Technology For Securing Cyber-Infrastructure And Internet Of Things Networks." Intelligent Interactive Multimedia Systems For E-Healthcare Applications (2022): 337-350.  [7] Sagu, Amit, Et Al. "Design Of Metaheuristic Optimization Algorithms For Deep Learning Model For Secure Iot Environment." Sustainability 15.3 (2023): 2204.  [8] Choi, Jongseok, Et Al. "Secure Iot Framework And 2d Architecture For End-To-End Security." The Journal Of Supercomputing 74 (2018): 3521-3535.  [9] Al Hwaitat, Ahmad K., Et Al. "A New Blockchain-Based Authentication Framework For Secure Iot Networks." Electronics 12.17 (2023): 3618.  [10] Huang, Xin, Et Al. "Seciot: A Security Framework For The Internet Of Things." Security And Communication Networks 9.16 (2016): 3083-3094.  [11] Chakrabarty, Shaibal, And Daniel W. Engels. "Secure Smart Cities Framework Using Iot And Ai." 2020 IEEE Global Conference On Artificial Intelligence And Internet Of Things (Gcaiot). IEEE, 2020.  [12] Abou El Houda, Zakaria, Lyes Khoukhi, And Bouziane Brik. "A Low-Latency Fog-Based Framework To Secure Iot Applications Using Collaborative Federated Learning." 2022 IEEE 47th Conference On Local Computer Networks (Lcn). IEEE, 2022  [13] Jothi, B., And M. Pushpalatha. "Wils-Trs—A Novel Optimized Deep Learning Based Intrusion Detection Framework For Iot Networks." Personal And Ubiquitous Computing 27.3 (2023): 1285-1301  [14] Balaswamy, K., Et Al. "A Secure Framework For Protecting Clinical Data In Medical Iot Environment." Smart Healthcare System Design: Security And Privacy Aspects (2022): 203-234.  [15] Subhashini, R., And Alex Khang. "The Role Of Internet Of Things (Iot) In Smart City Framework." Smart Cities. Crc Press, 2023. 31-56.  [16] Abu Al‐Haija, Qasem, Ahmad Al Badawi, And Giridhar Reddy Bojja. "Boost‐Defence For Resilient Iot Networks: A Head‐To‐Toe Approach." Expert Systems 39.10 (2022): E12934.  [17] Bhardwaj, Akashdeep, Et Al. "Iaf: Iot Attack Framework And Unique Taxonomy." Journal Of Circuits, Systems And Computers 31.02 (2022): 2250029  [18] Awotunde, Joseph Bamidele, And Sanjay Misra. "Feature Extraction And Artificial Intelligence-Based Intrusion Detection Model For A Secure Internet Of Things Networks." Illumination Of Artificial Intelligence In Cybersecurity And Forensics. Cham: Springer International Publishing, 2022. 21-44.  [19] Saqib, Manasha, Bhat Jasra, And Ayaz Hassan Moon. "A Lightweight Three Factor Authentication Framework For Iot Based Critical Applications." Journal Of King Saud University-Computer And Information Sciences 34.9 (2022): 6925-6937  [20] Ting, Li, Et Al. "A Secure Framework For Iot-Based Smart Climate Agriculture System: Toward Blockchain And Edge Computing." Journal Of Intelligent Systems 31.1 (2022): 221-236.  [21] Chee, Kok Onn, et al. "IoTSecSim: A framework for modelling and simulation of security in Internet of Things." *Computers & Security* 136 (2024): 103534.  [22] Alwahedi, Fatima, et al. "Machine learning techniques for IoT security: Current research and future vision with generative AI and large language models." *Internet of Things and Cyber-Physical Systems* (2024).  [23] Saied, Mohamed, Shawkat Guirguis, and Magda Madbouly. "Review of artificial intelligence for enhancing intrusion detection in the internet of things." *Engineering Applications of Artificial Intelligence* 127 (2024): 107231.  [24] Sharma, Bhawana, et al. "Explainable artificial intelligence for intrusion detection in IoT networks: A deep learning based approach." *Expert Systems with Applications* 238 (2024): 121751.  [25] Bilakanti, Harini, et al. "Anomaly detection in IoT environment using machine learning." *Security and Privacy* (2024): e366.  [26] Mansour, Salah-Eddine, et al. "Enhancing Security Mechanisms for IoT-Fog Networks." *Journal of Robotics and Control (JRC)* 5.1 (2024): 152-159.  [27] Lilhore, Umesh Kumar, Surjeet Dalal, and Sarita Simaiya. "A cognitive security framework for detecting intrusions in IoT and 5G utilizing deep learning." *Computers & Security* 136 (2024): 103560.  [28]Sami, Teba Mohammed Ghazi, Subhi RM Zeebaree, and Sarkar Hasan Ahmed. "A Novel Multi-Level Hashing Algorithm to Enhance Internet of Things Devices’ and Networks’ Security." *International Journal of Intelligent Systems and Applications in Engineering* 12.1s (2024): 676-696.  [29] Ali, Saqib, Qianmu Li, and Abdullah Yousafzai. "Blockchain and federated learning-based intrusion detection approaches for edge-enabled industrial IoT networks: a survey." Ad Hoc Networks 152 (2024): 103320.  [30] Singh, Sushil Kumar, et al. "GRU-based digital twin framework for data allocation and storage in IoT-enabled smart home networks." Future Generation Computer Systems 153 (2024): 391-402.  [31] Kizza, Joseph Migga. "Internet of things (iot): growth, challenges, and security." Guide to Computer Network Security. Cham: Springer International Publishing, 2024. 557-573.  [32] Khalid, Mizna, et al. "Towards SDN-based smart contract solution for IoT access control." Computer Communications 198 (2023): 1-31.  [33] Khan, AB Feroz, et al. "DDoS attack modeling and resistance using trust based protocol for the security of Internet of Things." Journal of Engineering Research 11.2 (2023): 100058.  [34] Xie, Hui, et al. "TEBDS: A Trusted Execution Environment-and-Blockchain-supported IoT data sharing system." Future Generation Computer Systems 140 (2023): 321-330.  [35] Dhar, Shalini, Ashish Khare, and Rajani Singh. "Advanced security model for multimedia data sharing in Internet of Things." Transactions on Emerging Telecommunications Technologies 34.11 (2023): e4621.  [36] Rekha, Shashi, et al. "Study of security issues and solutions in Internet of Things (IoT)." Materials Today: Proceedings 80 (2023): 3554-3559.  [37] Ragesh, G. K., and Ajay Kumar. "Trust-based secure routing and message delivery protocol for signal processing attacks in IoT applications." The Journal of Supercomputing 79.3 (2023): 2882-2909.  [38] Li, Xuanang, et al. "PSAP-WSN: a provably secure authentication protocol for 5g-based wireless sensor networks." CMES-Computer Modeling in Engineering & Sciences 135.1 (2023): 711.  [39] Fathy, Cherine, and Hassan M. Ali. "A secure IoT-based irrigation system for precision agriculture using the expeditious cipher." Sensors 23.4 (2023): 2091.  [40] Khan, Wali Ullah, et al. "Joint optimization for secure ambient backscatter communication in NOMA-enabled IoT networks." Digital Communications and Networks 9.1 (2023): 264-269.   |  | | --- | |  | |  | |

[41] Das, A.; McFarlane, A. Non-linear dynamics of electric power losses, electricity consumption, and GDP in Jamaica. Energy Econ. 2019, 84, 104530.

[42] Bashkari, S.; Sami, A.; Rastegar, M. Outage Cause Detection in Power Distribution Systems based on Data Majorng. IEEE Trans. Ind. Inf. 2020.

[43] Bank, T.W. Electric Power Transmission and Distribution Losses (% of output); IEA: Paris, France, 2016.

[44] Zheng, Z.; Yang, Y.; Niu, X.; Dai, H.-N.; Zhou, Y. Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids. IEEE Trans. Ind. Inform. 2018, 14, 1606–1615.

[45] Hasan, M.N., Toma, R.N., Nahid, A.A., Islam, M.M. and Kim, J.M., 2019. Electricity theft detection in smart grid systems: A CNN-LSTM based approach. Energies, 12(17), p.3310.

[46] K. Zheng, Q. Chen, Y. Wang, C. Kang and Q. Xia, "A Novel Combined Data-Driven Approach for Electricity Theft Detection," in IEEE Transactions on Industrial Informatics, vol. 15, no. 3, pp. 1809-1819, March 2019, doi: 10.1109/TII.2018.2873814.

[46] Li, S., Han, Y., Yao, X., Yingchen, S., Wang, J. and Zhao, Q., 2019. Electricity theft detection in power grids with deep learning and random forests. Journal of Electrical and Computer Engineering, 2019.

[47] M. Nabil, M. Ismail, M. M. E. A. Mahmoud, W. Alasmary and E. Serpedin, "PPETD: Privacy-Preserving Electricity Theft Detection Scheme with Load Monitoring and Billing for AMI Networks," in IEEE Access, vol. 7, pp. 96334-96348, 2019, doi: 10.1109/ACCESS.2019.2925322.

[48] Khan, Z.A., Adil, M., Javaid, N., Saqib, M.N., Shafiq, M. and Choi, J.G., 2020. Electricity theft detection using supervised learning techniques on smart meter data. Sustainability, 12(19), p.8023.

[49] Kocaman, B., Tümen, V. Detection of electricity theft using data processing and LSTM method in distribution systems. Sādhanā 45, 286 (2020). https://doi.org/10.1007/s12046-020-01512-0

[50] Li, B., Xu, K., Cui, X., Wang, Y., Ai, X., Wang, Y. (2018). Multi-scale DenseNet-Based Electricity Theft Detection. In: Huang, DS., Bevilacqua, V., Premaratne, P., Gupta, P. (eds) Intelligent Computing Theories and Application. ICIC 2018. Lecture Notes in Computer Science (), vol 10954. Springer, Cham. https://doi.org/10.1007/978-3-319-95930-6\_17

[51] A. Aldegheishem, M. Anwar, N. Javaid, N. Alrajeh, M. Shafiq and H. Ahmed, "Towards Sustainable Energy Efficiency with Intelligent Electricity Theft Detection in Smart Grids Emphasising Enhanced Neural Networks," in IEEE Access, vol. 9, pp. 25036-25061, 2021, doi: 10.1109/ACCESS.2021.3056566.