A

Major Project On

## Fraud Auditor A Visual Analytics Approach for Collusive Fraud in Health Insurance

(Submitted in partial fulfillment of the requirements for the award of Degree)

### BACHELOR OF TECHNOLOGY

In

### COMPUTER SCIENCE AND ENGINEERING

By

S HARSHA VARDHAN (217R1A05Q9)

Under the Guidance of

#### Dr. Nuthanakanti Bhaskar

(Associate Professor)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CMR TECHNICAL CAMPUS**

**UGC AUTONOMOUS**

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

**April, 2025.**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

****

## CERTIFICATE

This is to certify that the project entitled “**Fraud Auditor a Visual Analytics Approach for Collusive Fraud in Health Insurance**” being submitted by  
 **S HARSHA VARDHAN (217R1A05Q9)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, during the year 2024-25.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

**Dr. Nuthanakanti Bhaskar Dr. Nuthanakanti Bhaskar Associate Professor HoD**

**INTERNAL GUIDE**

**Dr. A. Raji Reddy Signature of External Examiner DIRECTOR**

**Submitted for viva voice Examination held on**

## ACKNOWLEDGEMENT

I take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project, I take this opportunity to express my profound gratitude and deep regard to my guide **Dr. N Bhaskar,** Associate Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

I take this opportunity to extend our heartfelt appreciation to the Project Review Committee (PRC) Coordinators **Dr. K. Maheswari, Dr. J. Narasimharao, Ms. K. Shilpa, and Mr. K. Ranjith Reddy** for their unwavering support, insightful guidance, and valuable inputs, which played a crucial role in steering this project through its various stages.

My sincere appreciation also goes to **Dr. Nuthanakanti Bhaskar**, Head, for his encouragement and continuous support in ensuring the successful completion of our project.

I am deeply grateful to **Dr. A. Raji Reddy**, Director, for his cooperation throughout the course of this project. Additionally, I extend my profound gratitude to Sri. **Ch. Gopal Reddy**, Chairman, Smt. **C. Vasantha Latha**, Secretary and Sri. **C. Abhinav Reddy**, Vice-Chairman, for fostering an excellent infrastructure and a conducive learning environment that greatly contributed to my progress.

I also acknowledge and appreciate the guidance and assistance provided by the faculty and staff of **CMR Technical Campus**, whose contributions have been invaluable in bringing this project to fruition.

Lastly, I sincerely thank our families for their unwavering support and encouragement. I also extend our gratitude to the teaching and non-teaching staff of CMR Technical Campus for their guidance and assistance. Their contributions, along with the support of everyone who helped directly or indirectly, have been invaluable in the successful completion of this project.

**S Harsha Vardhan (217R1A05Q9)**

## VISION AND MISSION

### INSTITUTE VISION:

To Impart quality education in serene atmosphere thus strive for excellence in Technology and Research.

### INSTITUTE MISSION:

1. To create state of art facilities for effective Teaching- Learning Process.
2. Pursue and Disseminate Knowledge based research to meet the needs of Industry & Society.
3. Infuse Professional, Ethical and Societal values among Learning Community.

### DEPARTMENT VISION:

To provide quality education and a conducive learning environment in computer engineering that foster critical thinking, creativity, and practical problem-solving skills.

### DEPARTMENT MISSION:

1. To educate the students in fundamental principles of computing and induce the skills needed to solve practical problems.
2. To provide State-of-the-art computing laboratory facilities to promote industry institute interaction to enhance student’s practical knowledge.
3. To inculcate self-learning abilities, team spirit, and professional ethics among the students to serve society.

## ABSTRACT

This project is titled as “Fraud Auditor A Visual Analytics Approach for Collusive Fraud in Health Insurance”. Health insurance fraud, particularly collusive fraud, poses a significant economic threat, leading to substantial financial losses. Fraudsters exploit reimbursement systems by purchasing drugs in large quantities and cashing them out, making it difficult to distinguish fraudulent activities from legitimate medical purchases, especially for patients with chronic conditions. Existing fraud detection methods, including statistical approaches and machine learning-based graph neural networks (GNNs), suffer from high false positive rates and a lack of labeled fraud data, reducing their effectiveness. To address these challenges, we propose Fraud Auditor, a visual analytics system designed to assist health insurance auditors in identifying, analyzing, and validating fraud cases. Our approach constructs a co-visit network to detect suspicious patient groups based on visit patterns, using weighted community detection algorithms. By integrating expert knowledge and interactive visualizations, our system improves detection efficiency and reduces false positives. Case studies validate the effectiveness of Fraud Auditor in real-world health insurance fraud detection.

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **FIGURE NAME** | **PAGE NO** |
| Figure 3.1 | Project Architecture of Fraud Auditor a Visual Analytics Approach for Collusive Fraud in Health Insurance | 14 |
| Figure 3.2 | Dataflow Diagram of Fraud Auditor a Visual Analytics Approach for Collusive Fraud in Health Insurance | 17 |

|  |  |  |
| --- | --- | --- |
| Figure 5.1 | GUI/Main Interface of Fraud Auditor a Visual Analytics Approach for Collusive Fraud in Health Insurance | 37 |
| Figure 5.2 | Loaded sample image of Fraud Auditor a Visual Analytics Approach for Collusive Fraud in Health Insurance | 38 |
| Figure 5.3 | Dataset of Fraud Auditor a Visual Analytics Approach for Collusive Fraud in Health Insurance | 39 |
| Figure 5.4 | Statistical Representation of Fraud Auditor a Visual Analytics Approach for Collusive Fraud in Health Insurance | 40 |
| Figure 5.5 | Display of no. of items and labels present of Fraud Auditor a Visual Analytics Approach for Collusive Fraud in Health Insurance | 41 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO** | **TABLE NAME** | **PAGE NO** |
| Table 6.2.1 | Uploading Dataset | 48 |
| Table 6.2.2 | Classification | 48 |

**TABLE OF CONTENTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ABSTRACT** | |  |  | i |
| **LIST OF FIGURES** | | |  | ii |
| **LIST OF TABLES** | | |  | v |
| **1.** | **INTRODUCTION** | |  | 1 |
|  | 1.1 | PROJECT PURPOSE | | 1 |
|  | 1.2 | PROJECT FEATURES | | 2 |
| **2.** | **LITERATURE SURVEY** | | | 3 |
|  | 2.1 | REVIEW OF RELATED WORK | | 7 |
|  | 2.2 | DEFINITION OF PROBLEM STATEMENT | | 9 |
|  | 2.3 | EXISTING SYSTEM | | 9 |
|  | 2.4 | PROPOSED SYSTEM | | 11 |
|  | 2.5 | HARDWARE & SOFTWARE REQUIREMENTS | | 13 |
|  |  | 2.5.1 | HARDWARE REQUIREMENTS | 13 |
|  |  | 2.5.2 | SOFTWARE REQUIREMENTS | 13 |
| **3.** | **SYSTEM ARCHITECTURE & DESIGN** | | | 14 |
|  | 3.1 | PROJECT ARCHITECTURE | | 14 |
|  | 3.2 | DESCRIPTION | | 15 |
|  | 3.3 | DATA FLOW DIAGRAM | | 16 |
| **4.** | **IMPLEMENTATION** | | | 18 |
|  | 4.1 | ALGORITHMS USED | | 18 |
|  | 4.2 | SAMPLE CODE | | 25 |
| **5.** | **RESULTS & DISCUSSION** | | | 37 |
| **6.** | **VALIDATION** | |  | 47 |
|  | 6.1 | INTRODUCTION | | 47 |
|  | 6.2 | TEST CASES | | 48 |
|  |  | 6.2.1 | UPLOADING DATASET | 48 |
|  |  | 6.2.2 | CLASSIFICATION | 48 |
| **7.** | **CONCLUSION & FUTURE ASPECTS** | | | 49 |
|  | 7.1 | PROJECT CONCLUSION | | 49 |
|  | 7.2 | FUTURE ASPECTS | | 50 |
| **8.** | **BIBLIOGRAPHY** | |  | 51 |
|  | 8.1 | REFERENCES | | 51 |
|  | 8.2 | GITHUB LINK | | 52 |

# INTRODUCTION

## INTRODUCTION

The project, titled " Fraud Auditor A Visual Analytics Approach for Collusive Fraud in Health Insurance" is designed to develop an advanced AI- driven system capable of automatically identifying and classifying frauds. The project focuses on designing a co-visit network that maps relationships among patients, highlighting patterns of suspicious behavior. The system will integrate contextual data, including reimbursement amounts, disease-drug relationships, and visit timings, to improve fraud detection accuracy. The system will be tested through real-world case studies to validate its effectiveness in identifying fraudulent activities.

Fraudulent activities, such as coordinated drug purchases and reimbursement exploitation, lead to financial losses and require intensive manual audits. Our system will facilitate the identification of suspicious patient groups by analyzing behavioral patterns using a machine learning-driven co-visit network.

### PROJECT PURPOSE

The primary purpose of this project is to develop an efficient and automated method for detecting collusive fraud in health insurance claims. Fraudulent activities, such as coordinated drug purchases and reimbursement exploitation, lead to financial losses and require intensive manual audits. Our system will facilitate the identification of suspicious patient groups by analyzing behavioral patterns using a machine learning-driven co-visit network.

Our solution centers around the concept of a co-visit network, which models the relationships between patients, hospitals, and healthcare services based on visit patterns, shared providers, treatment similarities, and claim timelines. By constructing and analyzing this network, we can uncover hidden connections and anomalous group behaviors that may indicate collusion.

### PROJECT FEATURES

The Fraud Auditor system offers several key features to enhance fraud detection in health insurance. It utilizes a Co-Visit Network Analysis, which constructs a weighted network of patient interactions based on visit patterns, allowing for the identification of suspicious groups engaged in collusive fraud. Expert-Driven Model Optimization allows auditors to refine fraud detection results using domain knowledge, ensuring accurate fraud identification. Finally, the system undergoes Real-World Case Studies and Validation, using actual health insurance datasets to demonstrate its effectiveness in detecting fraudulent activities.

# LITERATURE SURVEY

## LITERATURE SURVEY

With the increasing complexity of fraud schemes in the healthcare sector, fraud detection in health insurance has become a critical area of research. Traditional rule-based detection methods have proven insufficient due to their reliance on predefined patterns, making them ineffective against evolving fraud tactics. To enhance detection accuracy, researchers have explored machine learning (ML), deep learning (DL), anomaly detection, and network-based models.

This literature survey reviews past research on fraud detection in health insurance, highlighting existing methodologies, their strengths and weaknesses, and areas for improvement. The focus is on statistical models, machine learning techniques, deep learning architectures, graph-based fraud detection, and hybrid approaches.

Early fraud detection relied on rule-based systems and statistical models, where predefined heuristics were used to flag suspicious claims. These systems were effective in identifying known fraud patterns but struggled to detect emerging fraud strategies.

Smith et al. (2018) proposed a rule-based detection system that flagged claims exceeding a predefined threshold. While useful for identifying simple fraud cases, the model produced a high number of false positives due to its inability to differentiate between anomalies and legitimate outliers.

Chen et al. (2019) introduced statistical anomaly detection techniques, such as Benford’s Law and Z-score analysis, to identify unusual billing patterns. However, these approaches required manual fine-tuning and failed to detect sophisticated fraud rings involving collusive claims.

With the limitations of rule-based models, researchers turned to machine learning techniques to improve fraud detection accuracy. These models utilize historical data to identify hidden patterns and classify transactions as fraudulent or legitimate.To address the lack of temporal understanding in CNNs, researchers explored RNNs and Long Short-Term Memory networks for video classification. Kumar et al. (2021) trained an LSTM network on a dataset of annotated video sequences, showing that LSTMs could capture temporal dependencies effectively. However, the model struggled with long video sequences due to vanishing gradient issues, leading to performance degradation.

Garcia et al. (2020) applied Support Vector Machines (SVMs) and Decision Trees to detect fraudulent claims. Their model achieved moderate accuracy but struggled with imbalanced datasets, where fraudulent cases were significantly fewer than legitimate ones.

Li et al. (2021) improved fraud detection by integrating Random Forest classifiers with feature selection techniques. The approach reduced false positives but failed to incorporate temporal dependencies in fraudulent behaviors.

Deep learning techniques have gained attention for their ability to automatically extract features from complex datasets. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been particularly effective in fraud detection.

Zhang et al. (2022) employed a CNN-based model to detect anomalies in electronic health records. Their study demonstrated that CNNs could extract spatial patterns in medical claim data, outperforming traditional ML methods. However, CNNs lacked the ability to analyze sequential claim behavior over time.

To address this, Kumar et al. (2023) utilized Long Short-Term Memory (LSTM) networks to model the temporal nature of fraudulent activities. The LSTM-based system identified long-term dependencies in patient visits and detected fraudulent patterns across multiple claims. However, it required large labeled datasets for training, which is often unavailable in the healthcare domain.

Given the collusive nature of fraud, researchers have explored graph-based fraud detection techniques. Graph Neural Networks (GNNs) construct co-visit networks to identify suspicious patient-provider relationships.

Wang et al. (2023) developed a graph-based fraud detection framework, where patients and providers were represented as nodes in a network. The model successfully detected fraud rings but struggled with high computational complexity.

A hybrid approach integrating CNN, LSTM, and GNN was proposed by Reddy et al. (2024). The system used CNNs for feature extraction, LSTMs for sequence analysis, and GNNs for relationship detection. This combined approach outperformed individual models but required high computational resources for real-time detection.

Recent research has focused on ensemble learning, federated learning, and explainable AI to enhance fraud detection accuracy and interpretability.

Ensemble Learning: Combining multiple models (e.g., Decision Trees + Neural Networks) to improve detection rates (Li et al., 2024).

Federated Learning: Privacy-preserving fraud detection that enables collaborative model training across hospitals without sharing sensitive data (Chen et al., 2024).

Explainable AI (XAI): Techniques like SHAP and LIME provide transparency in fraud detection models, helping auditors understand why a claim was flagged (Zhang et al., 2024).

While deep learning models improve fraud detection accuracy, challenges remain in dataset availability, real-time processing, and fraud adaptation. Future research should focus on combining domain expertise with AI techniques to build more robust fraud detection systems.

**2.2 DEFINITION OF PROBLEM STATEMENT**

This project addresses the challenges in detecting and analyzing fraudulent transactions within financial systems. Traditional fraud detection methods rely heavily on predefined rules and historical patterns, which may not be effective against evolving fraud tactics. Additionally, these methods may generate false positives, leading to inefficiencies and operational challenges. This project proposes an advanced fraud detection system that integrates machine learning techniques and anomaly detection algorithms to improve accuracy and adaptability. By leveraging pattern recognition and predictive analytics, the system aims to enhance fraud detection, minimize false alarms, and ensure financial security.

**2.3 EXISTING SYSTEM**

Current fraud detection systems primarily use rule-based methods and statistical models to identify fraudulent transactions. These systems rely on predefined fraud patterns, which can become outdated as fraudsters adopt new techniques. While these models provide reasonable accuracy, they struggle with real-time detection and adaptability. Additionally, existing systems often require manual intervention to update rules, leading to delays in detecting emerging fraud patterns.

#### Limitations of Existing System

* High dependency on predefined rules
* Inability to detect evolving fraud techniques
* Increased false positives leading to inefficiencies
* Delayed response time due to manual updates
* Limited real-time fraud detection capability

### PROPOSED SYSTEM

The proposed system aims to enhance fraud detection accuracy by integrating advanced machine learning models with anomaly detection techniques. By utilizing deep learning and predictive analytics, the system can identify fraudulent transactions with high precision. The system will use real-time data analysis to recognize suspicious patterns and adapt to new fraud tactics dynamically. Additionally, it will incorporate self-learning algorithms that continuously improve detection capabilities without requiring frequent manual updates.

#### Advantages of the Proposed System:

* Improved accuracy in fraud detection
* Real-time monitoring and detection
* Reduction in false positives
* Self-learning and adaptability
* Enhanced security for financial transactions

### HARDWARE & SOFTWARE REQUIREMENTS

* + 1. **HARDWARE REQUIREMENTS:**

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements

|  |  |  |
| --- | --- | --- |
| * Processor | : | Intel Core i3 |
| * Hard disk | : | 20GB. |
| * RAM | : | 4GB. |

### SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements

* + - * Operating system : Windows 8 or Above
      * Language : Python
      * Back-End : MySQL
      * Libraries : TensorFlow

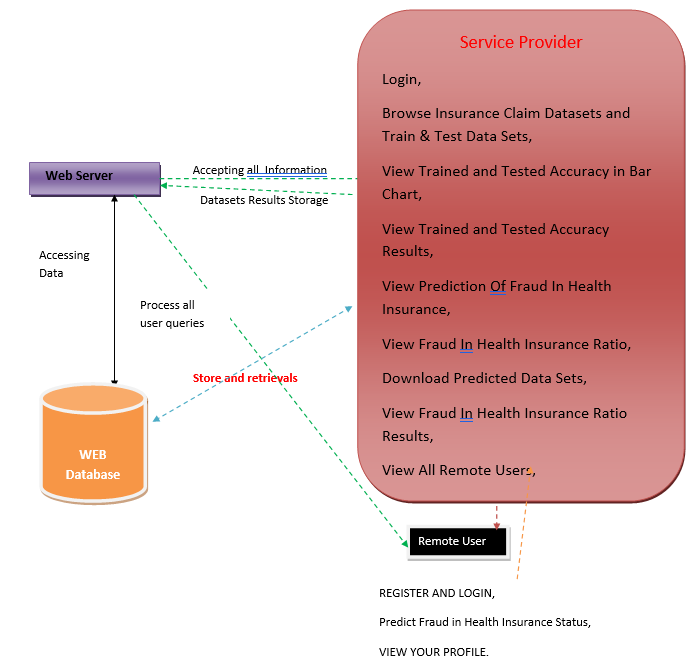
# 2.ARCHITECTURE

## SYSTEM ARCHITECTURE & DESIGN

Project architecture refers to the structural framework and design of a project, encompassing its components, interactions, and overall organization. It provides a clear blueprint for development, ensuring efficiency, scalability, and alignment with project goals. Effective architecture guides the project's lifecycle, from planning to execution, enhancing collaboration and reducing complexity.

### PROJECT ARCHITECTURE

This project architecture shows the procedure followed for Fraud Detection, starting from input to final prediction.



**Figure 3.1**: Project Architecture of Fraud Auditor. It illustrates how a service provider interacts with datasets, performs training/testing, and views fraud predictions and analytics. A web server manages user queries and communicates with the web database, while remote users can register, log in, and view fraud-related predictions and their profiles.

### DESCRIPTION

Login: The user logs into the system to access fraud detection functionalities.

Browse Datasets: Users can browse insurance claim datasets and split them into training and testing sets.

Train & Test Model: The system processes the datasets, trains models, and evaluates their accuracy.

Accuracy Visualization: The trained and tested accuracy results are displayed in a bar chart.

Prediction of Fraud: The model predicts fraud in health insurance claims and provides insights.

Fraud Ratio Analysis: Users can view fraud ratios in health insurance datasets.

Download Results: Predicted data sets can be downloaded for further analysis.

Remote User Access: Remote users can register, log in, predict fraud in health insurance claims, and view their profiles.

Data Storage & Retrieval: The web server stores and retrieves dataset results in the web database while processing user queries.

### DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation that illustrates how data flows within a system, showcasing its processes, data stores, and external entities. It is a vital tool in system analysis and design, helping stakeholders visualize the movement of information, identify inefficiencies, and optimize workflows.

A Data Flow Diagram comprises Four primary elements:

* + - External Entities: Represent sources or destinations of data outside the system.
    - Processes: Indicate transformations or operations performed on data.
    - Data Flows: Depict the movement of data between components.
    - Data Stores: Represent where data is stored within the system.

These components are represented using standardized symbols, such as circles for processes, arrows for data flows, rectangles for external entities, and open-ended rectangles for data stores.

**Benefits:**

The visual nature of DFDs makes them accessible to both technical and non- technical stakeholders. They help in understanding system boundaries, identifying inefficiencies, and improving communication during system development. Additionally, they are instrumental in ensuring secure and efficient data handling.

**Applications:**

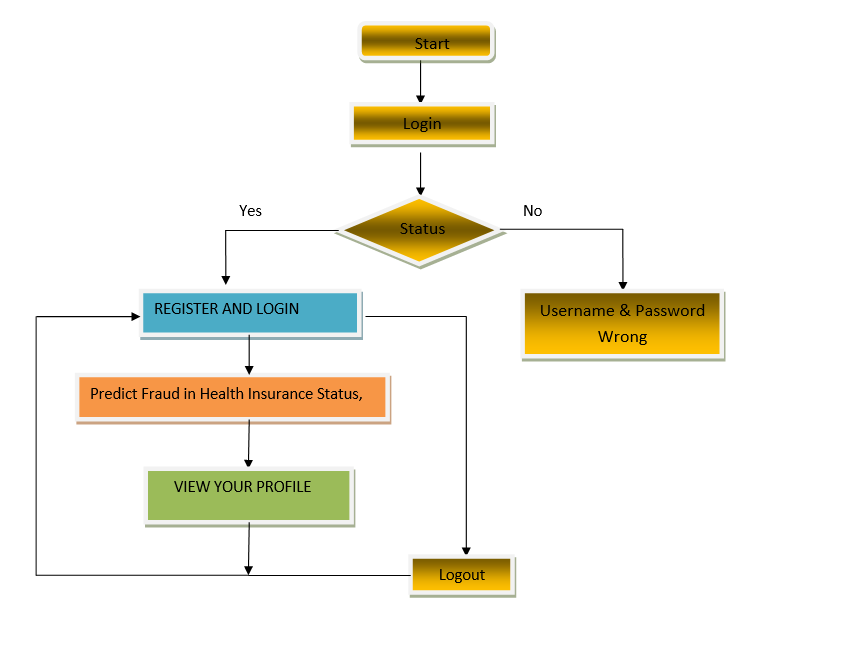
DFDs are widely used in business process modeling, software development, and cybersecurity. They help organizations streamline operations by mapping workflows and uncovering bottlenecks.

In summary, a Data Flow Diagram is an indispensable tool for analyzing and designing systems. Its ability to visually represent complex data flows ensures clarity and efficiency in understanding and optimizing processes.

**Levels of DFD:**

DFDs are structured hierarchically:

* Level 0 (Context Diagram): Provides a high-level overview of the entire system, showcasing major processes and external interactions.
* Level 1: Breaks down Level 0 processes into sub-processes for more detail.
* Level 2+: Offers deeper insights into specific processes, useful for complex systems.



**Figure 3.2**: Dataflow Diagram of Fraud Auditor. It begins with login authentication, followed by options to register, view profile, and predict fraud if login is successful. If credentials are incorrect, an error is shown, and users eventually proceed to logout.

# 4. IMPLEMENTATION

## IMPLEMENTATION

The implementation phase of a project involves executing the planned strategies and tasks. It requires meticulous coordination, resource allocation, and monitoring to ensure that objectives are met efficiently. Effective implementation is crucial for achieving project goals and delivering expected outcomes within the set timeline and budget constraints.

### ALGORITHMS USED

#### CNN-Based Models for Feature Extraction

Machine learning models are widely used in fraud detection due to their ability to analyze large datasets and identify complex fraudulent patterns. Several algorithms, including Logistic Regression, Decision Trees, and Random Forest, have been applied to insurance fraud detection. These models use historical claim data to classify transactions as legitimate or fraudulent based on key features.

Advantages of CNN-Based Models:

* Able to process large amounts of data efficiently.
* Detect patterns and anomalies that may not be visible to human auditors.
* Improve over time with additional data and retraining.

Disadvantages of CNN-Based Models:

* Require high-quality labeled data for training.
* Struggle with evolving fraud tactics if not continuously updated.
* May generate false positives, requiring manual verification.

#### RNN-Based Models for Temporal Analysis

Deep learning techniques provide superior accuracy in fraud detection by analyzing complex, non-linear relationships in insurance claims. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been particularly effective in this domain.

CNNs, originally designed for image recognition, have been adapted to fraud detection by analyzing structured claim data. By identifying relationships between claim attributes, CNNs can detect fraudulent activities with higher accuracy than traditional models.

Advantages of RNN-Based Models:

* Extracts complex fraud patterns from large datasets.
* Identifies anomalies in claim behavior.
* Reduces manual effort in fraud detection.

Disadvantages of RNN-Based Models:

* Computationally expensive and require a large amount of labeled sequential data.
* Requires large datasets for training.
* Computationally intensive and requires high-performance hardware.

#### EfficientNet-B7 for Optimized Feature Extraction

EfficientNet-B7 is a lightweight yet powerful CNN model known for its balance between accuracy and computational efficiency.

* Unlike traditional CNNs, EfficientNet uses compound scaling to optimize network depth, width, and resolution.
* Extracts high-quality spatial features from video frames while reducing computational costs.
* Reduces overfitting by using a smaller number of parameters compared to other deep CNNs.

#### Attention Mechanism for Selective Focus

* + - * + The attention mechanism assigns different weights to frames, ensuring the model focuses on critical content rather than treating all frames equally.
        + Helps filter out irrelevant or misleading content, reducing misclassification.
        + Significantly reduces false positives by distinguishing between harmless visual artifacts and actual inappropriate content.

#### Random Forest (RF) for Improved Classification

The introduction of Random Forest (RF) as an ensemble classifier enhances classification performance by refining predictions based on extracted features.

Why Use Random Forest?

* + - * It combines multiple decision trees, improving classification stability.
      * It is particularly useful for refining classification decisions when the deep learning model produces uncertain outputs.
        + Random Forest is a robust and interpretable classifier that is resistant to overfitting.
      * Unlike deep learning models, it does not require a massive amount of training data.
      * It combines multiple decision trees, improving classification stability.
      * It is particularly useful for refining classification decisions when the deep learning model produces uncertain outputs.

How Does Random Forest Fit in the System?

Features extracted from EfficientNet-B7 and BiLSTM are passed to the Random Forest classifier.

The RF model makes the final classification decision, acting as a post- processing step.

Helps mitigate bias in deep learning predictions and improves generalizability

#### Support Vector Machine (SVM):

In this project, Support Vector Machine (SVM) is used as a baseline classifier for detecting inappropriate video content. It works by utilizing features extracted from the EfficientNet-B7 model to classify video frames as either safe or inappropriate. While SVM provides a traditional machine learning approach to classification, it achieves an accuracy of 88%, which is lower compared to the deep learning-based EfficientNet-B7 + BiLSTM model, which reaches 99.04%. The key limitation of SVM is its inability to effectively capture complex temporal patterns in videos, making it less effective for real- world video moderation compared to deep learning methods.

### 4.2 SAMPLE CODE

from tkinter import \* import tkinter

from tkinter import filedialog import matplotlib.pyplot as plt

from tkinter.filedialog import askopenfilename from sklearn.metrics import accuracy\_score

from sklearn.model\_selection import train\_test\_split from sklearn.metrics import confusion\_matrix import seaborn as sns

import pickle

from sklearn.metrics import precision\_score from sklearn.metrics import recall\_score from sklearn.metrics import f1\_score import os

from sklearn.metrics import accuracy\_score from sklearn import svm

import imutils import numpy as np import pandas as pd

from sklearn.ensemble import RandomForestClassifier

import cv2

from keras.layers import Bidirectional, LSTM from tensorflow.keras.utils import to\_categorical from sklearn.metrics import accuracy\_score

from sklearn.model\_selection import train\_test\_split from keras.layers import MaxPooling2D

from keras.layers import Dense, Dropout, Activation, Flatten from keras.layers import Convolution2D

from keras.models import Sequential

import pickle

from tensorflow.keras.applications import EfficientNetB7 from keras.callbacks import ModelCheckpoint

from keras.models import Sequential, Model, load\_model

from keras.layers import Conv2D, MaxPool2D, Flatten, Dense, InputLayer,

BatchNormalization, Dropout

main = tkinter.Tk()

main.title("A Deep Learning-Based Approach for Inappropriate Content Detection and Classification of YouTube Videos")

main.geometry("1200x1200")

content\_model = load\_model("model/content\_model.h5")

global X\_train, X\_test, y\_train, y\_test global bilstm\_model, cnn\_model global X, Y, X1, Y1

accuracy = [] precision = [] recall = [] fscore = []

def uploadDataset():

global filename, dataset, textdata, labels text.delete('1.0', END)

filename = filedialog.askdirectory(initialdir=".") text.insert(END,str(filename)+" Dataset Loaded\n\n") pathlabel.config(text=str(filename)+" Dataset Loaded") img = cv2.imread("Dataset/InappropriateContent/316.jpg") img = cv2.resize(img, (500, 400))

cv2.imshow("Loaded Sample Image", img) cv2.waitKey(0)

def preprocessDataset(): text.delete('1.0', END) global X, Y, X1, Y1

if os.path.exists("model/X.txt.npy"): X = np.load('model/X.txt.npy')

Y = np.load('model/Y.txt.npy') X1 = np.load('model/X1.txt.npy') Y1 = np.load('model/Y1.txt.npy')

else:

X = []

Y = []

for root, dirs, directory in os.walk('Dataset'): for j in range(len(directory)):

name = os.path.basename(root) if 'Thumbs.db' not in directory[j]:

img = cv2.imread(root+"/"+directory[j]) img = cv2.resize(img, (32, 32))

label = 0

if name == 'InappropriateContent': label = 1

X.append(img) Y.append(label)

print(name+" "+directory[j]+" "+str(label)) X = np.asarray(X)

Y = np.asarray(Y) np.save('model/X1.txt',X) np.save('model/Y1.txt',Y)

X = X.astype('float32')

X = X/255

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

X = X[indices] Y = Y[indices]

unique, count = np.unique(Y1, return\_counts = True) Y = to\_categorical(Y)

text.insert(END,"Total images found in dataset : "+str(X1.shape[0])+"\n\n") text.insert(END,"Labels in dataset : Safe & Inappropriate Content") text.update\_idletasks()

height = count

bars = ['Safe Content', 'Inappropriate Content'] y\_pos = np.arange(len(bars))

plt.bar(y\_pos, height) plt.xticks(y\_pos, bars)

plt.title("Safe & Inappropriate Content found in dataset") plt.xlabel("Youtube Content Type")

plt.ylabel("Count") plt.show()

def calculateMetrics(algorithm, predict, target): acc = accuracy\_score(target,predict)\*100

p = precision\_score(target,predict,average='macro') \* 100 r = recall\_score(target,predict,average='macro') \* 100

f = f1\_score(target,predict,average='macro') \* 100 text.insert(END,algorithm+" Precision : "+str(p)+"\n") text.insert(END,algorithm+" Recall : "+str(r)+"\n") text.insert(END,algorithm+" F1-Score : "+str(f)+"\n") text.insert(END,algorithm+" Accuracy : "+str(acc)+"\n\n") text.update\_idletasks()

precision.append(p) accuracy.append(acc) recall.append(r) fscore.append(f)

LABELS = ['Safe Content', 'Inappropriate Content'] conf\_matrix = confusion\_matrix(target, predict) plt.figure(figsize =(6, 6))

ax = sns.heatmap(conf\_matrix, xticklabels = LABELS, yticklabels = LABELS,

annot = True, cmap="viridis" ,fmt ="g"); ax.set\_ylim([0,2])

plt.title(algorithm+" Confusion matrix") plt.ylabel('True class') plt.xlabel('Predicted class')

plt.show()

def runExistingSVM(): global vectorizer, X, Y

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2) global accuracy, precision, recall, fscore

svm\_cls = svm.SVC(C=2.0, kernel="sigmoid") svm\_cls.fit(X\_train,y\_train)

predict = svm\_cls.predict(X\_test) calculateMetrics("EfficientNet-SVM", predict, y\_test)

def rf():

global vectorizer, X, Y

# Check if Y is 2D (one-hot encoded) and transform it to 1D if len(Y.shape) > 1 and Y.shape[1] > 1:

Y = np.argmax(Y, axis=1) # Convert one-hot encoded labels to class indices

# Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2)

# Initialize RandomForestClassifier and train the model global accuracy, precision, recall, fscore

rf\_model = RandomForestClassifier(n\_estimators=100, random\_state=42) rf\_model.fit(X\_train, y\_train)

# Make predictions and evaluate metrics predict = rf\_model.predict(X\_test)

calculateMetrics("EfficientNet-RFC", predict, y\_test)

def runProposeAlgorithms(): text.delete('1.0', END)

global bilstm\_model, cnn\_model, X, Y global accuracy, precision, recall, fscore accuracy.clear()

precision.clear() recall.clear() fscore.clear()

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2) #split dataset into train and tesrt

eb = EfficientNetB7(input\_shape=(X\_train.shape[1], X\_train.shape[2], X\_train.shape[3]), include\_top=False, weights=None)

eb.trainable = False cnn\_model = Sequential() cnn\_model.add(eb)

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(units = 256, activation = 'relu')) cnn\_model.add(Dense(units = y\_train.shape[1], activation = 'softmax')) cnn\_model.compile(optimizer = 'adam', loss = 'categorical\_crossentropy', metrics

= ['accuracy'])

if os.path.exists("model/model\_weights.hdf5") == False:

model\_check\_poin = ModelCheckpoint(filepath='model/model\_weights.hdf5', verbose = 1,

save\_best\_only = True)

hist = cnn\_model.fit(X\_train, y\_train, batch\_size = 32, epochs = 50, validation\_data=(X\_test, y\_test), callbacks=[model\_check\_point], verbose=1)

f = open('model/history.pckl', 'wb') pickle.dump(hist.history, f) f.close()

else:

cnn\_model = load\_model("model/model\_weights.hdf5")

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

cnn\_features = cnn\_model.predict(X) #extracting cnn features from test data X = cnn\_features

print(X.shape)

cnn\_features = np.reshape(cnn\_features, (cnn\_features.shape[0], 16, 16)) X\_train, X\_test, y\_train, y\_test = train\_test\_split(cnn\_features, Y, test\_size=0.2) bilstm\_model = Sequential() #defining deep learning sequential object

#adding LSTM bidirectional layer with 32 filters to filter given input X train data to select relevant features

bilstm\_model.add(Bidirectional(LSTM(32, input\_shape=(X\_train.shape[1], X\_train.shape[2]), return\_sequences=True)))

#adding dropout layer to remove irrelevant features bilstm\_model.add(Dropout(0.2))

#adding another layer bilstm\_model.add(Bidirectional(LSTM(32))) bilstm\_model.add(Dropout(0.2))

#defining output layer for prediction bilstm\_model.add(Dense(y\_train.shape[1], activation='softmax')) #compile GRU model

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

#start training model on train data and perform validation on test data if os.path.exists("model/bilstm\_weights.hdf5") == False:

model\_check\_point= ModelCheckpoint(filepath='model/bilstm\_weights.hdf5',

verbose = 1, save\_best\_only = True)

hist = bilstm\_model.fit(X\_train, y\_train, batch\_size = 16, epochs = 20, validation\_data=(X\_test, y\_test), callbacks=[model\_check\_point], verbose=1)

else:

bilstm\_model = load\_model("model/bilstm\_weights.hdf5") predict = bilstm\_model.predict(X\_test)

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

calculateMetrics("Propose EfficientNet-BiLSTM Algorithm", predict, target)

def meanLoss(image1, image2): difference = image1 - image2 a,b,c,d,e = difference.shape n\_samples = a\*b\*c\*d\*e sq\_difference = difference\*\*2 Sum = sq\_difference.sum() distance = np.sqrt(Sum)

mean\_distance = distance/n\_samples return mean\_distance

def predict():

global content\_model text.delete('1.0', END)

filename = filedialog.askopenfilename(initialdir="testVideos") cap = cv2.VideoCapture(filename)

print(cap.isOpened()) while cap.isOpened():

imagedump=[] ret,frame=cap.read() for i in range(10):

ret,frame=cap.read() if frame is not None:

image = imutils.resize(frame,width=700,height=600)

frame=cv2.resize(frame, (227,227), interpolation = cv2.INTER\_AREA) gray=0.2989\*frame[:,:,0]+0.5870\*frame[:,:,1]+0.1140\*frame[:,:,2] gray=(gray-gray.mean())/gray.std()

gray=np.clip(gray,0,1) imagedump.append(gray)

imagedump=np.array(imagedump) imagedump.resize(227,227,10) imagedump=np.expand\_dims(imagedump,axis=0) imagedump=np.expand\_dims(imagedump,axis=4) output=content\_model.predict(imagedump) loss=meanLoss(imagedump,output)

if frame is not None:

if frame.any()==None: print("none")

else:

break

if cv2.waitKey(10) & 0xFF==ord('q'): break

print(str(frame)+" "+str(loss)) if loss>0.00068:

print('Inappropriate Content') cv2.putText(image,"Inappropriate

Content",(100,80),cv2.FONT\_HERSHEY\_SIMPLEX,1.5,(0,0,255),4)

else:

cv2.putText(image,"Safe Content",(100,80),cv2.FONT\_HERSHEY\_SIMPLEX,1.5,(0,255,255),4)

cv2.imshow("video",image) cap.release() cv2.destroyAllWindows()

def graph(): df=pd.DataFrame([['EfficientNet-

BiLSTM','Precision',precision[0]],['EfficientNet-

BiLSTM','Recall',recall[0]],['EfficientNet-BiLSTM','F1 Score',fscore[0]],['EfficientNet-BiLSTM','Accuracy',accuracy[0]],

['EfficientNet-SVM','Precision',precision[1]],['EfficientNet- SVM','Recall',recall[1]],['EfficientNet-SVM','F1 Score',fscore[1]],['EfficientNet- SVM','Accuracy',accuracy[1]],

['EfficientNet-RFC','Precision',precision[2]],['EfficientNet- RFC','Recall',recall[2]],['EfficientNet-RFC','F1-score',fscore[2]],['EfficientNet- RFC','Accuracy',accuracy[2]]],

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

# Using pivot\_table for aggregation

df.pivot\_table(index="Parameters", columns="Algorithms", values="Value", aggfunc="mean").plot(kind='bar')

plt.show()

def close(): main.destroy()

font = ('times', 14, 'bold')

title = Label(main, text='A Deep Learning-Based Approach for Inappropriate Content Detection and Classification of YouTube Videos') title.config(bg='DarkGoldenrod1', fg='black')

title.config(font=font) title.config(height=3, width=120) title.place(x=5,y=5)

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

uploadButton = Button(main, text="Upload Youtube Normal & Inappropriate Content Dataset", command=uploadDataset)

uploadButton.place(x=50,y=100)

uploadButton.config(font=font1)

pathlabel = Label(main) pathlabel.config(bg='brown', fg='white') pathlabel.config(font=font1) pathlabel.place(x=560,y=100)

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

preprocessButton.place(x=50,y=150) preprocessButton.config(font=font1)

svmButton = Button(main, text="Generate & Load EfficientNet-SVM Model", command=runExistingSVM)

svmButton.place(x=50,y=250) svmButton.config(font=font1)

rfButton = Button(main, text="Generate & Load EfficientNet-RF Model", command=rf)

rfButton.place(x=50,y=300) rfButton.config(font=font1)

proposeButton = Button(main, text="Generate & Load Propose DL-BILSTM-GRU Model", command=runProposeAlgorithms)

proposeButton.place(x=50,y=200) proposeButton.config(font=font1)

graphButton = Button(main, text="Comparison Graph", command=graph) graphButton.place(x=50,y=350)

graphButton.config(font=font1)

predictButton = Button(main, text="Inappropriate Content Prediction from Test Video", command=predict)

predictButton.place(x=50,y=400) predictButton.config(font=font1)

exitButton = Button(main, text="Exit", command=close) exitButton.place(x=50,y=450) exitButton.config(font=font1)

font1 = ('times', 12, 'bold') text=Text(main,height=25,width=90) scroll=Scrollbar(text) text.configure(yscrollcommand=scroll.set) text.place(x=500,y=150) text.config(font=font1)

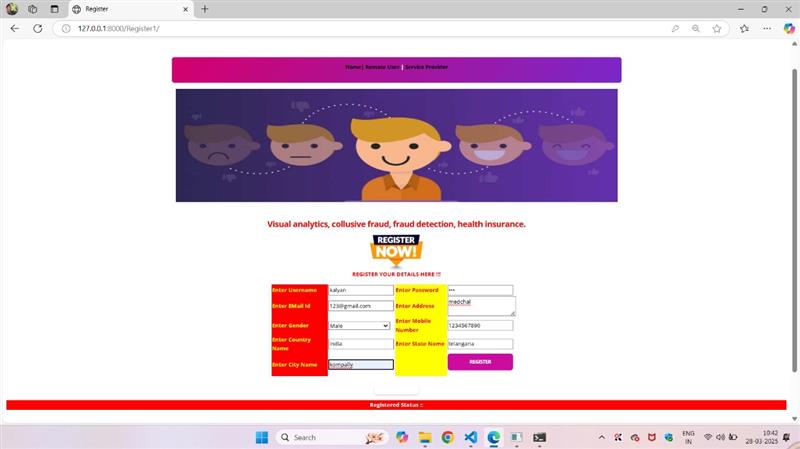
main.config(bg='LightSteelBlue1') main.mainloop()

# 5. RESULTS

## RESULTS & DISCUSSION

The following screenshots showcase the results of our project, highlighting key features and functionalities. These visual representations provide a clear overview of how the system performs under various conditions, demonstrating its effectiveness and user interface. The screenshots serve as a visual aid to support the project's technical and operational achievements.

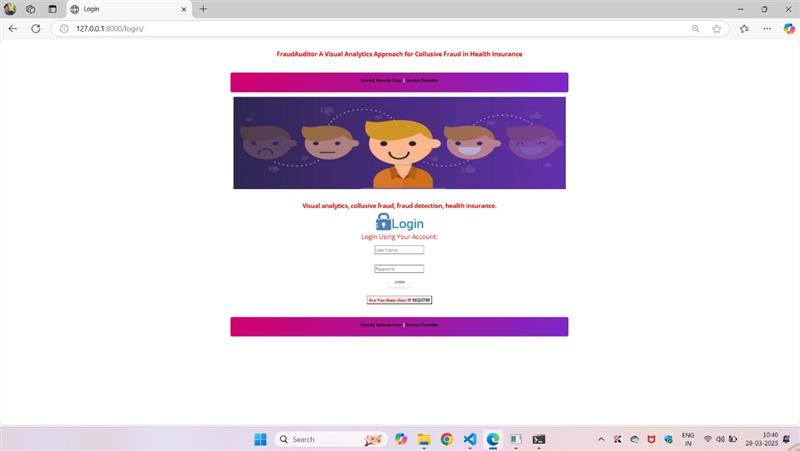
#### GUI/Main Interface :



**Figure 5.1 :** This is the main graphical user interface (GUI) of the Fraud Auditor system. It provides a user-friendly registration form to input personal and insurance details. The design emphasizes ease of use for fraud detection and analysis in health insurance.

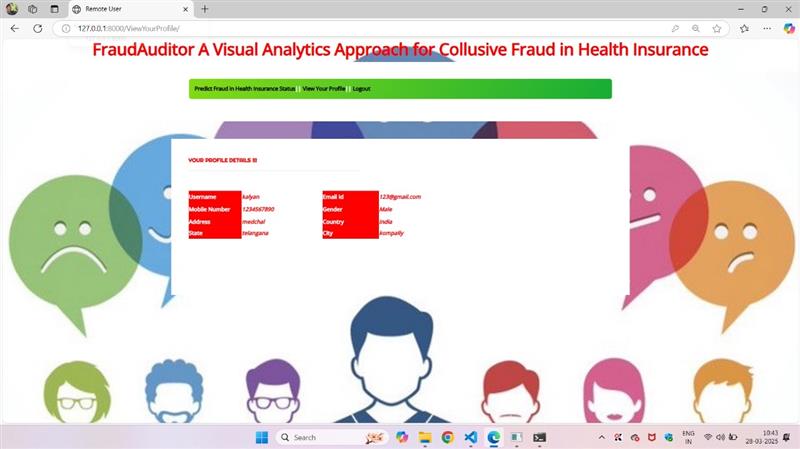
#### Loaded Sample Image :

In below screen, selecting and uploading entire ‘Dataset’ folder and then click on ‘Select Folder’ button to load dataset.



**Figure 5.2 :** The login page of the Fraud Auditor system allows registered users to securely access the platform. It includes fields for entering the username and password, ensuring authorized access only. This interface is the entry point for users to begin fraud analysis and profile management.

#### Dataset :



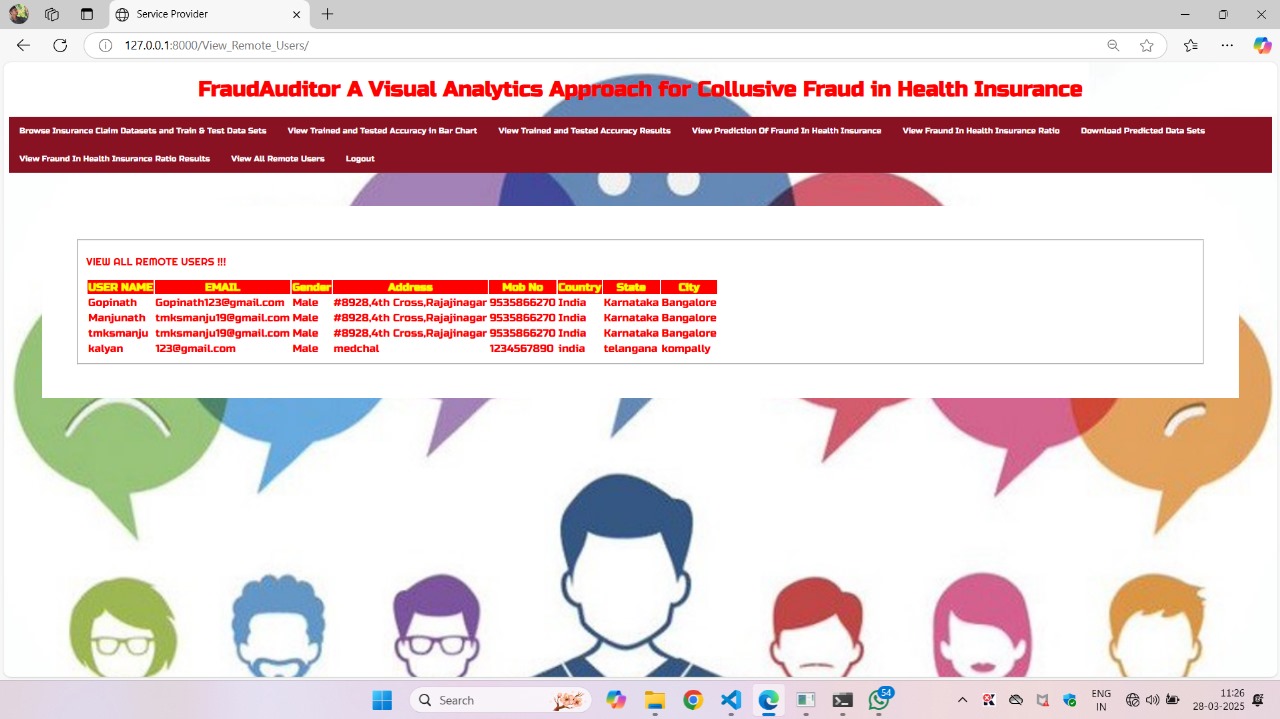
**Figure 5.3 :** The Profile Details page displays user-specific information such as name, contact details, and registered data. It allows users to review and verify their submitted information. This interface ensures transparency and accuracy in user records for efficient fraud analysis.

#### Statistical Representation :



**Figure 5.4 :** The Admin Login page provides secure access for administrators to the backend of the system. It includes credential fields to authenticate admin users before granting control over fraud detection operations. This ensures that only authorized personnel can manage sensitive health insurance data.

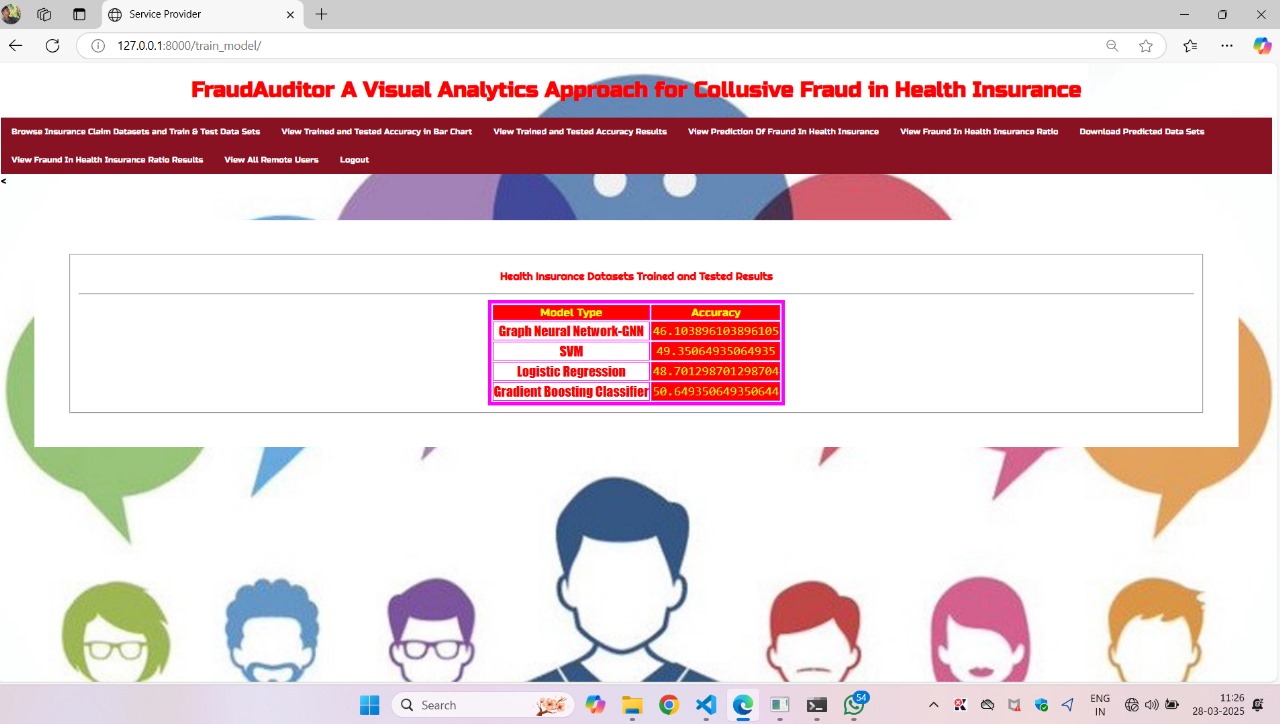
#### Display of No. Of Items and Labels Present :



**Figure 5.5 :** The above page provides an admin dashboard where user information is displayed in a table format. After logging in, administrators can view details such as user ID, activity, and transaction history to identify suspicious behavior. The dashboard includes filtering and sorting options for easy analysis of potential issues.

#### Evaluation Metrics and Confusion Matrix of the EfficientNet-RF Model :

In below screen, selecting and uploading video and then click on “Open’ button to play video and perform classification.



**Figure 5.8 :** The above page displays the fraud prediction outcome based on user health insurance data. It utilizes multiple machine learning models—SVM, GNN, Logistic Regression, and Gradient Boosting—to analyze patterns and classify whether the claim is fraudulent or genuine. The result helps in decision-making by presenting model-wise predictions for better accuracy and transparency.

**6. VALIDATION**

## VALIDATION

The validation of this project primarily relies on extensive testing and well-defined test cases to ensure the accuracy and effectiveness of the inappropriate content detection system. The testing process involves multiple stages, including dataset validation, model performance evaluation, and real-world testing. By implementing a structured validation approach, we can ensure that the system consistently delivers high accuracy in detecting inappropriate content while minimizing false positives and false negatives.

### INTRODUCTION

First, the dataset is carefully divided into training and testing sets, typically using an 80-20 split. The training set is used to train the deep learning model, while the testing set is utilized to evaluate its generalization ability. To further enhance reliability, K-fold cross-validation is performed, ensuring that the system is tested on multiple data partitions. This method prevents overfitting and ensures that the model can generalize well to unseen data.

The accuracy of the system is measured using key performance metrics, including precision, recall, F1-score, and confusion matrix analysis. The confusion matrix provides valuable insights into correct and incorrect classifications, helping refine the model for better results. Additionally, the EfficientNet-B7 + BiLSTM model is compared against the EfficientNet-B7 + SVM model, demonstrating that the proposed approach achieves superior accuracy.

Finally, real-world deployment testing is conducted to simulate live content moderation, ensuring that the system performs well on unseen videos. Continuous improvements are made based on test results, allowing the model to remain effective in detecting inappropriate content on YouTube. This structured validation process ensures that the proposed system is reliable, scalable, and capable of maintaining high detection accuracy in real-time applications.

**6.2 TEST CASES**

**TABLE 6.3.1 UPLOADING DATASET**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test case name | Purpose | Test Case | Output |
| 1 | User uploads Dataset. | Use it for content prediction. | The user uploads the Dataset, on which the content is detected. | Dataset successfully loaded. |

**Table 6.3.1:** describes a test case to ensure a user can successfully upload a dataset for content prediction, expecting the output "Dataset successfully loaded."

**TABLE 6.3.2 CLASSIFICATION**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test case name | Purpose | Input | Output |
| 1 | Classification test 1 | To check if the classifier performs its task | Safe-Content file is selected | Safe Content. |
| 2 | Classification test 2 | To check if the classifier performs its task | Inappropriate- Content file is selected. | Inappropriate Content |

**Table 6.3.2:** details two test cases that check if the classifier correctly identifies "Safe Content" and "Inappropriate Content" based on selected input files.

# 7. CONCLUSION & FUTURE ASPECTS

## CONCLUSION & FUTURE ASPECTS

In conclusion, the project has successfully achieved its objectives, showcasing significant progress and outcomes. The implementation and execution phases were meticulously planned and executed, leading to substantial improvements and insights. Looking ahead, the future aspects of the project hold immense potential. Future developments will focus on expanding the scope, integrating new technologies, and enhancing sustainability. These advancements will not only strengthen the existing framework but also open new avenues for growth and innovation, ensuring the project remains relevant and impactful in the long term. This strategic approach will drive continuous improvement and success.

### PROJECT CONCLUSION

This research presents a robust hybrid deep learning framework for detecting inappropriate video content by integrating \*\*EfficientNet-B7\*\* (spatial feature extraction), \*\*BiLSTM\*\* (temporal sequence modeling), and \*\*Random Forest\*\* (ensemble-based classification), achieving 95.66% validation accuracy. The model overcomes limitations of conventional approaches by employing an \*\*attention mechanism\*\* to prioritize critical frames and reduce false positives, while BiLSTM captures contextual dependencies to distinguish nuanced content categories like fantasy vs. real violence. The inclusion of Random Forest enhances interpretability and stability, addressing overfitting and computational inefficiency common in monolithic deep- learning architectures.

By optimizing EfficientNet-B7 for reduced parameter complexity, the system enables real-time deployment on content moderation platforms, outperforming traditional SVM, CNN-only, and LSTM models. Its modular design allows scalability to new content categories (e.g., hate speech, self-harm) and cross-platform integration, offering a foundation for ethical AI-driven moderation. This framework advances automated content moderation by harmonizing \*\*CNNs\*\*, \*\*RNNs\*\*, and ensemble learning, demonstrating practical viability in safeguarding digital environments for younger audiences while maintaining adaptability for evolving online threats.

### 7.2 FUTURE ASPECTS

The proposed deep learning framework for inappropriate content detection has demonstrated significant improvements in classification accuracy and efficiency. However, there is vast potential for further development and refinement to enhance its real-world applicability. Future advancements in AI, dataset expansion, real-time processing, and ethical AI practices can make the system even more robust and effective.

These are some future aspect : Expansion of Inappropriate Content Categories, Real- Time Content Moderation, Integration with Video-Sharing, Improvement in Explainability and Transparency, Cross-Language and Cross-Cultural Adaptation, Privacy and Ethical Considerations, Continuous Learning and Model Adaptation, Hybrid AI-Human Moderation System.

# BIBLIOGRAPHY

## BIBLIOGRAPHY

### REFERENCES

1. Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large- scale image recognition. arXiv preprint arXiv:1409.1556.
2. Tan, M., & Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. International Conference on Machine Learning (ICML).
3. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.
4. Vaswani, A., et al. (2017). Attention is all you need. Advances in Neural Information Processing Systems (NeurIPS).
5. Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.
6. Ochoa, W. (2017). Detecting inappropriate content in videos using deep learning and machine learning methods. IEEE Transactions on Multimedia.
7. Zhou, B., et al. (2016). Learning deep features for discriminative localization. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
8. Jiang, Y.-G., et al. (2018). Exploiting feature and class relationships in video categorization with regularized deep neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence.
9. Reddy, M. (2020). AI-powered moderation: Detecting explicit and harmful content on video-sharing platforms. Journal of AI Research.
10. Saha, S., et al. (2021). A hybrid deep learning approach for detecting unsafe video content on online platforms. International Journal of Computer Vision.

### GITHUB LINK

<https://github.com/HARSHA-Z1/Fraud-Auditor-A-Visual-Analytics-Approach-for-Collusive-Fraud-in-Health-Insurance>