# 

**CHAPTER 1**

**INTRODUCTION**

**1.1 General Introduction**

The accurate and efficient monitoring of student attendance is essential for maintaining academic discipline and enhancing educational outcomes. Traditional methods of attendance recording, such as manual roll calls and sign-in sheets, are time-consuming and prone to human error, leading to inaccuracies and inefficiencies. The advent of advanced machine learning techniques, particularly Convolutional Neural Networks (CNNs), offers a promising solution to automate and enhance the attendance tracking process. This project, "Attendance Surveillance System Using Convolutional Neural Networks (CNNs)," leverages these technologies to create a robust and reliable system for recognizing and recording student attendance through facial recognition.

**1.2 Problem Statement**

Manual attendance tracking methods, such as taking roll calls or using physical attendance sheets, are not only time-consuming but also prone to errors. In large classroom settings, where student populations can be substantial, maintaining accurate attendance records becomes increasingly challenging. Factors such as students entering late, leaving early, or even moving within the classroom during the session further complicate the process. These inaccuracies can have significant implications for academic records and student accountability, potentially leading to discrepancies in grades or disciplinary issues.

While some automated solutions exist to address these challenges, many rely on traditional machine learning approaches. These methods typically involve intricate image processing and feature extraction steps, which can be computationally intensive and resource-demanding. Moreover, the effectiveness of these approaches may vary, and they may not always deliver the desired level of accuracy, especially in dynamic classroom environments.

As a result, there is a pressing need for a more advanced and efficient automated attendance tracking system. Such a system should be capable of accurately and reliably recognizing student faces, even amidst the complexities of large classroom sizes and varying attendance patterns. Importantly, it should operate seamlessly without requiring constant human intervention or manual input.

**1.3 Objectives of the Project**

The primary objectives of this project are:

* **Develop a Reliable System**: Create an automated attendance surveillance system using CNNs that accurately recognizes student faces in classroom images.
* **Improve Accuracy**: Enhance the accuracy of face recognition by employing advanced techniques such as transfer learning with FaceNet and CNNs.
* **User-Friendly Interface**: Implement a simple and intuitive GUI using Tkinter, allowing users to easily upload images, process data, and view attendance records.
* **Automate Attendance Recording**: Provide a seamless process for recording and storing attendance data, minimizing the need for manual intervention.
* **Evaluate Performance**: Test and validate the system using a diverse set of classroom images to ensure its reliability and robustness in different conditions.

**1.4 Project Deliverables**

The project aims to deliver a comprehensive and robust attendance surveillance system that addresses the limitations of manual attendance tracking methods and traditional automated solutions. By leveraging advanced technologies like Convolutional Neural Networks (CNNs) for facial recognition and attendance recording, the system will offer enhanced accuracy, efficiency, and reliability.

* **Classroom Dataset**: A collection of classroom images used for training and testing the model, with separate datasets for each.
* **Face Detection and Extraction Module**: A component using MTCNN to detect and extract individual faces from classroom images.
* **Face Recognition Model**: A CNN-based model trained using FaceNet embeddings for accurate face recognition.
* **Graphical User Interface (GUI)**: A Tkinter-based interface with functionalities for uploading images, processing data, viewing results, and accessing attendance records in Excel format.
* **Performance Report**: Documentation detailing the system's accuracy, including a comparison between CNN and SVM models.

**1.5 Current Scope**

The current scope of this project includes:

* **Data Collection and Preprocessing**: Capturing and preparing classroom images for training and testing.
* **Model Training and Evaluation**: Training the CNN model using the prepared dataset and evaluating its performance against a test set.
* **GUI Development**: Creating a user-friendly interface for image upload, data processing, and result visualization.
* **System Testing**: Conducting thorough testing to ensure the model's accuracy and the GUI's functionality.

**1.6 Future Scope**

The future scope of this project can be expanded to include:

* **Real-Time Processing**: Enhancing the system to process live video feeds for real-time attendance tracking.
* **Scalability**: Adapting the system for larger datasets and more diverse classroom environments.
* **Additional Features**: Incorporating features such as emotion detection and behavior analysis to provide deeper insights into student engagement.
* **Integration with School Management Systems**: Developing interfaces to integrate the attendance system with existing school management software for streamlined operations.
* **Mobile Application**: Creating a mobile application version of the system for greater accessibility and convenience.

By addressing these areas, the Attendance Surveillance System can be further refined and adapted to meet the evolving needs of educational institutions, ensuring a more effective and efficient approach to attendance management.

**CHAPTER 2**

**PROJECT ORGANIZATION**

**2.1 Software Process Models**

For the development of the "Attendance Surveillance System Using Convolutional Neural Networks (CNNs)," the Agile software development model has been selected. This model is characterized by its iterative and incremental approach, which emphasizes flexibility, collaboration, and responsiveness to change. Below is an elaboration of how Agile was implemented in this project:

**2.1.1 Agile Software Development Model:**

* **Project Initiation and Planning:**
  + **Goal Setting:** Define the overall project goals, scope, and deliverables. This includes understanding the requirements for the attendance surveillance system and setting clear objectives.
  + **Team Formation:** Assemble a cross-functional team with members who have expertise in data collection, model development, GUI design, and documentation.
* **Iterations (Sprints):**
  + **Sprint Planning:** At the beginning of each sprint (typically 2-4 weeks), the team meets to plan the tasks to be completed. Tasks are prioritized based on their importance and dependencies.
  + **Task Assignment:** Specific tasks are assigned to team members based on their roles and expertise. Each member is responsible for delivering their assigned tasks by the end of the sprint.
* **Daily Stand-ups:**
  + **Progress Updates:** Short, daily meetings are held where team members report on their progress, discuss any obstacles, and outline their plans for the day. This ensures transparency and allows for quick resolution of issues.
  + **Collaboration:** Team members collaborate closely, often working together to solve problems and integrate their work.
* **Development and Testing:**
  + **Incremental Development:** Features and functionalities are developed incrementally. For example, in early sprints, the focus might be on data collection and preprocessing, while later sprints might focus on model training and GUI development.
  + **Continuous Integration:** Code is integrated and tested continuously to ensure that new changes do not break existing functionality. Automated tests are run regularly to catch issues early.
* **Sprint Review:**
  + **Demonstration:** At the end of each sprint, the team demonstrates the completed work to the project guide. This includes showing the progress made on the CNN model, GUI, and other system components.
  + **Feedback:** The project guide provides feedback, which is then considered for future sprints. This iterative feedback loop ensures that the project stays aligned with the guide's expectations.
* **Sprint Retrospective:**
  + **Reflection:** The team reflects on the sprint, discussing what went well, what could be improved, and any lessons learned. This helps in continuously improving the process and increasing efficiency.
* **Final Deployment and Review:**
  + **Final Integration:** Once all features are developed and tested, they are integrated into a final version of the system.
  + **User Acceptance Testing:** The final system is tested by the team to ensure it meets the project's requirements and expectations.
  + **Deployment:** The system is deployed for demonstration and evaluation by the project guide.
* **Maintenance and Support:**
  + **Ongoing Support:** After the project presentation, the team provides support to address any issues that arise and to implement any final adjustments based on the guide's feedback.

**2.1.2 Advantages of Agile in this Project:**

* **Flexibility:** Agile allows for changes in requirements and priorities, making it easier to adapt to new findings or feedback.
* **Collaboration:** Regular interaction among team members and with the project guide ensures that the project meets the set objectives.
* **Incremental Delivery:** By delivering the project in increments, progress can be demonstrated regularly and feedback can be incorporated early.
* **Improved Quality:** Continuous integration and testing help in identifying and fixing issues early, resulting in a more robust final product.

**2.2 Roles and Responsibilities**

**Table 2.1 Roles and Responsibility of all Team Members**

|  |  |  |
| --- | --- | --- |
| Sl.No | Name | Roles and Responsibility |
| 1 | Chandana N S | - Model Designing  - Excel Integration  - Testing |
| 2 | Lingadalli Sri Kavya | - Raw Data Collection  - Data Preprocessing  - Model Designing |
| 3 | Lakshmishree B U | - Documentation  - Enhancing Dataset  - GUI |
| 4 | Rumana Begum | - Model Developing  - Optimizing Model  - Documentation |

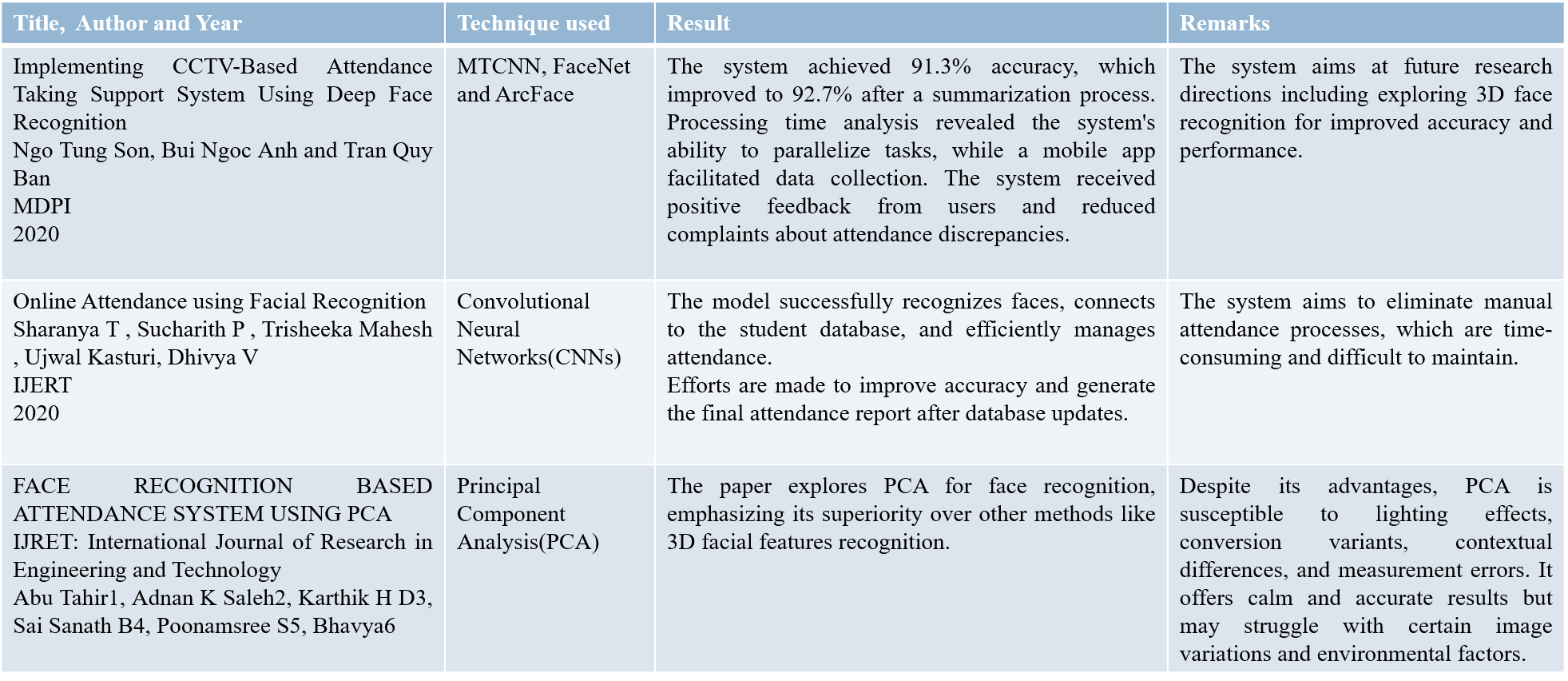
**CHAPTER 3**

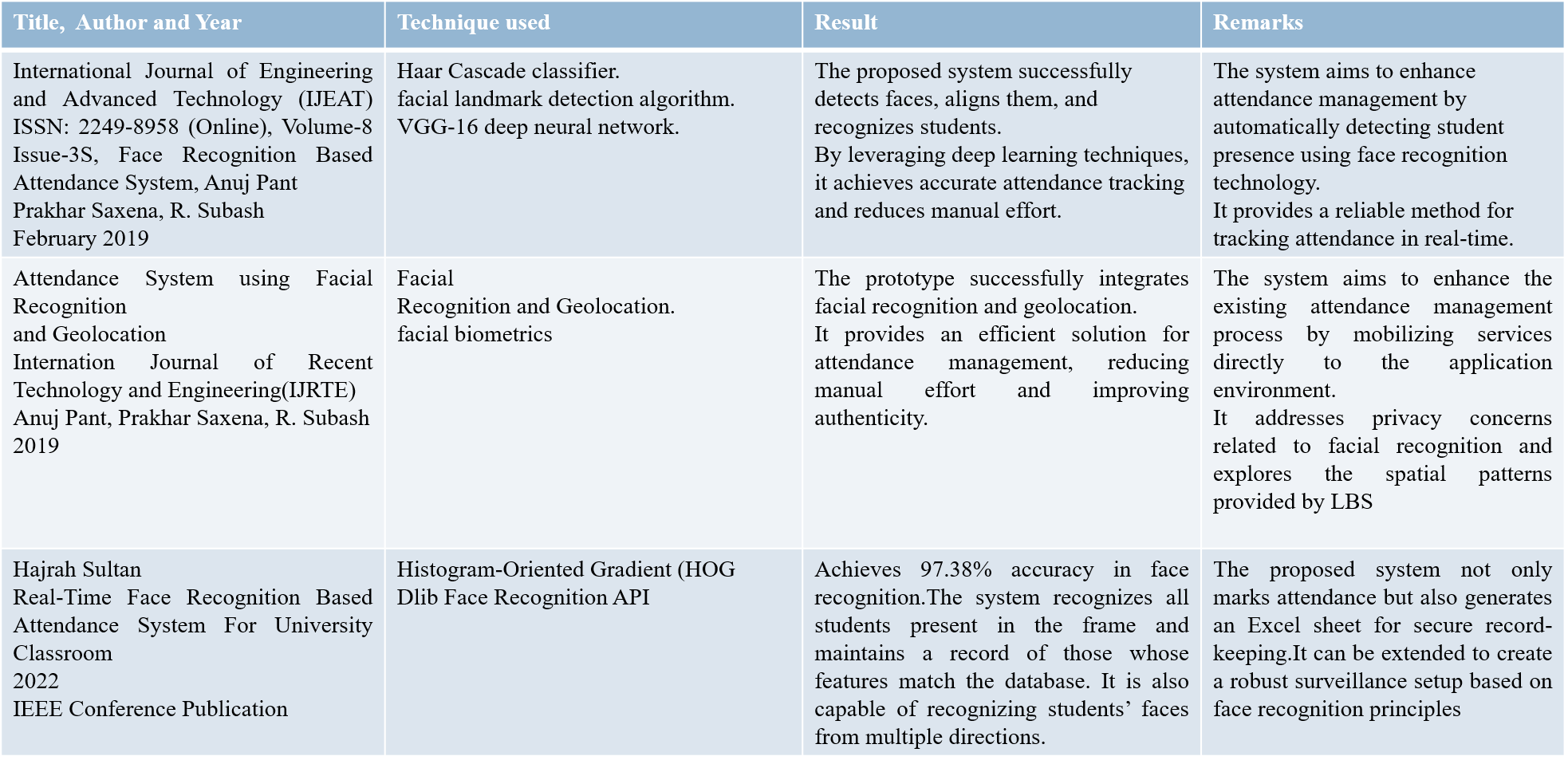
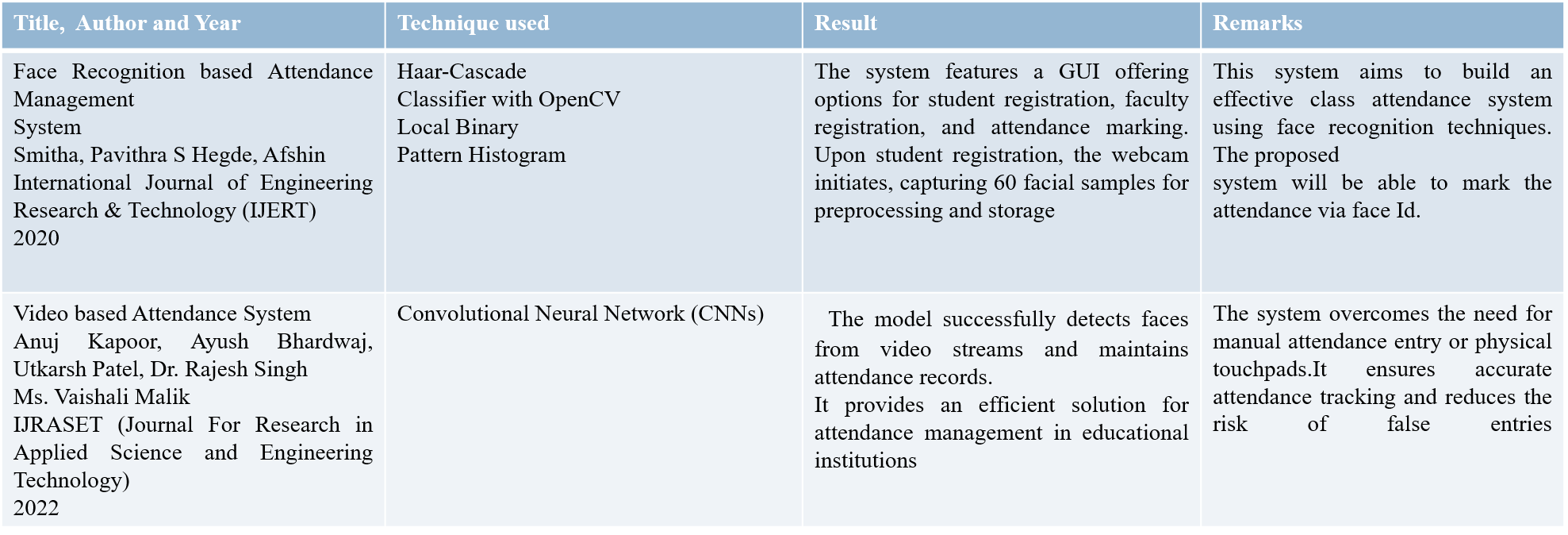
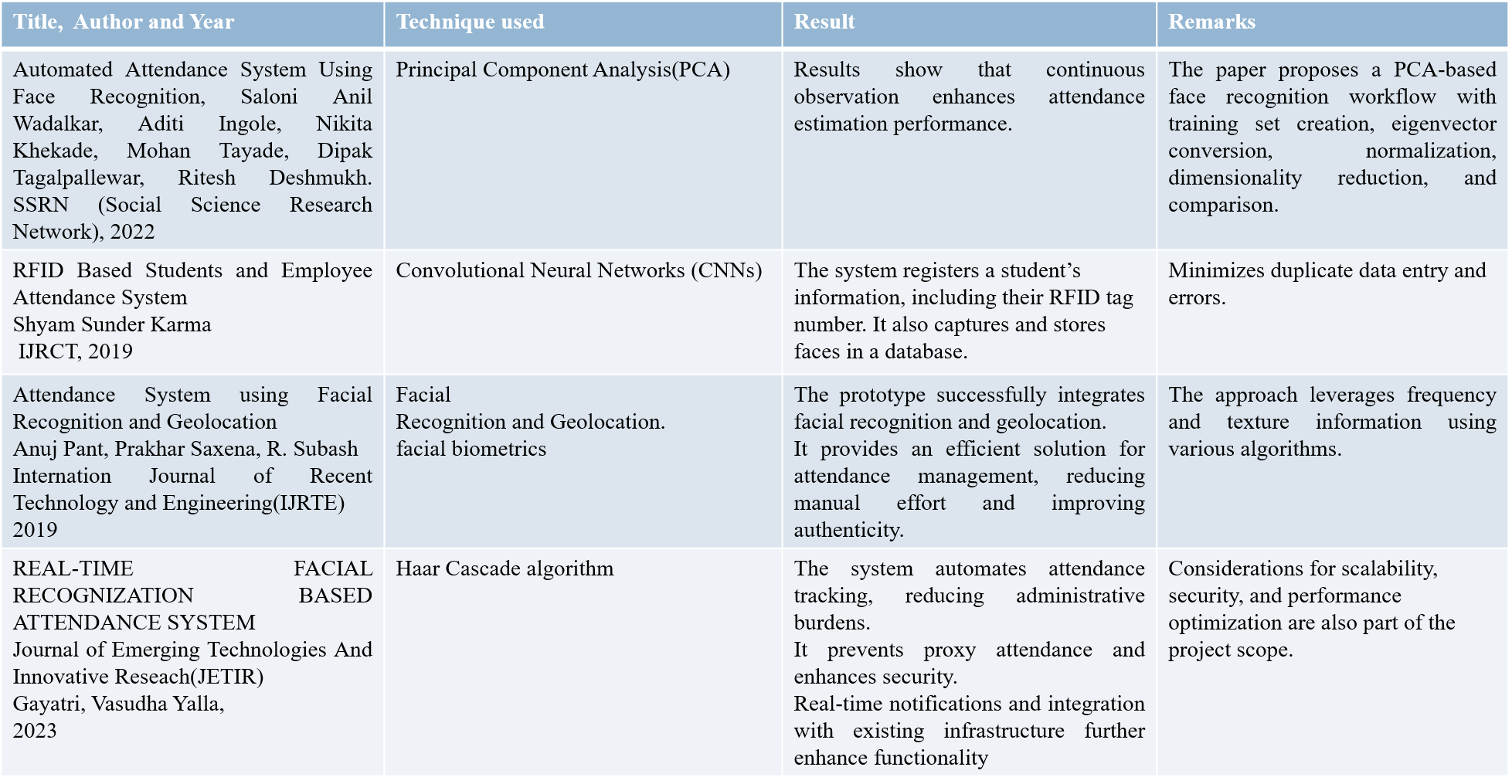
**LITERATURE SURVEY**

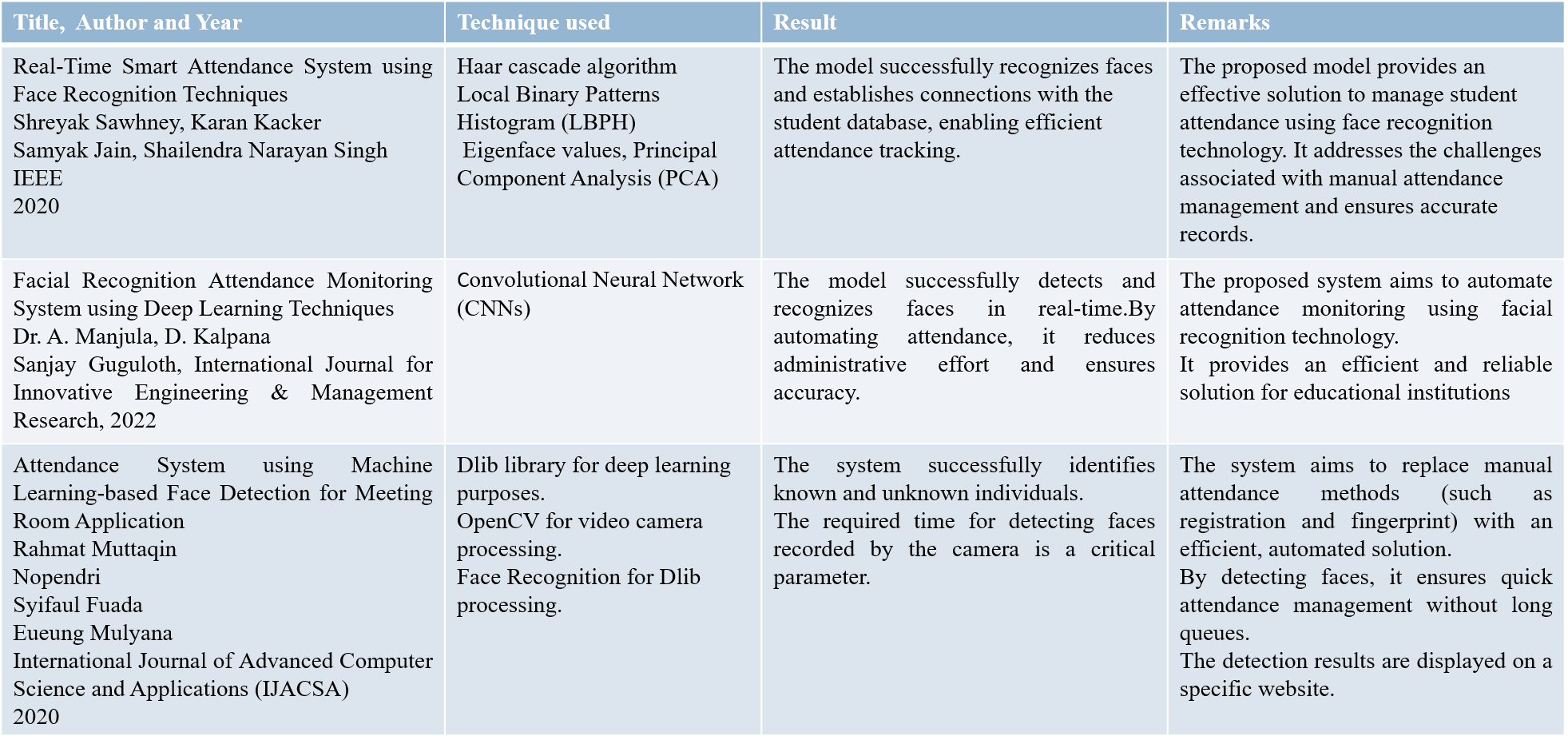
**3.1 Introduction**

The Attendance Surveillance System using Convolutional Neural Networks (CNNs) represents an innovative solution at the forefront of educational technology. This system leverages the power of deep learning to automatically identify and monitor student presence within classrooms. By capturing high-resolution classroom photographs, the system addresses privacy concerns associated with continuous video surveillance. A dataset compiled from these classroom photos trains the CNN model, enabling it to accurately detect students in images and generate detailed attendance reports. This approach offers a robust, scalable, and privacy-conscious solution for educational institutions seeking to improve efficiency, accuracy, and reliability in their attendance management practices.

**3.2 Related Works**

**Table 3.1** **Literature Survey**





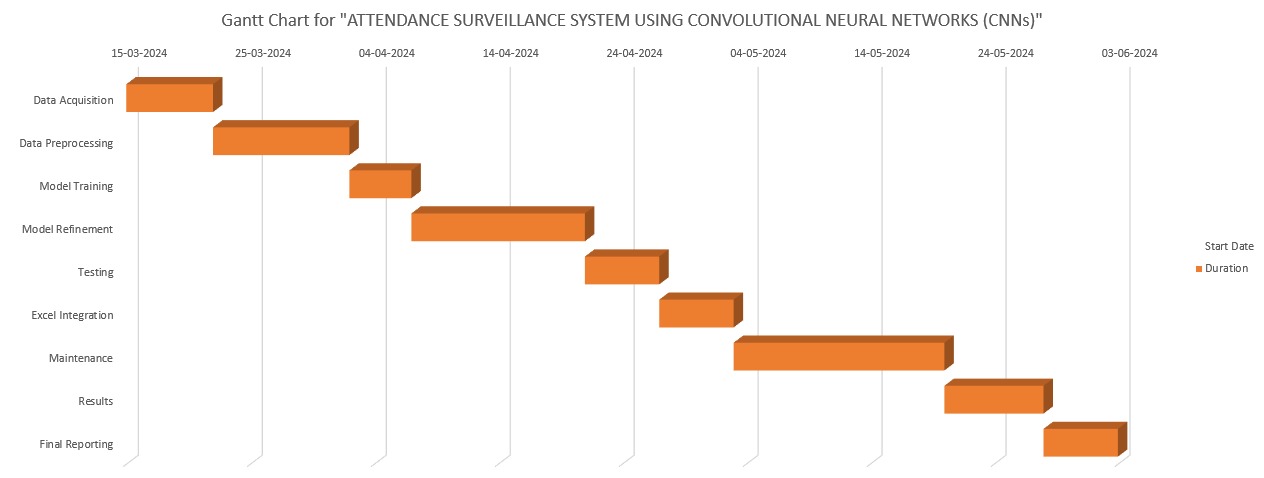
**3.3 Conclusion of Survey**

The survey of attendance monitoring systems reveals that the proposed project offers significant advancements compared to existing methods. Traditional manual attendance tracking is error-prone and labor-intensive, leading to inaccuracies in academic records. Existing automated methods, which often rely on traditional machine learning techniques, struggle with complex feature extraction, reducing their efficiency and accuracy. The new project uses MTCNN for precise face detection and extraction, with distinct training and testing datasets to ensure robust evaluation. By leveraging FaceNet and CNN, the project achieves an accuracy of 87.5%, outperforming the 83.3% accuracy of SVM models. Furthermore, the user-friendly Tkinter-based GUI includes features for project information, image uploading, data processing, and attendance results in Excel, making the system efficient, accurate, and easy to use for attendance surveillance.

**CHAPTER 4**

**PROJECT MANAGEMENT PLAN**

**4.1 Schedule of the Project**



**Fig 4.1 Gantt Chart**

**4.2 Risk Identification**

Implementing an Attendance Surveillance System using Convolutional Neural Networks (CNNs) involves various potential risks that need to be identified and managed to ensure the success and reliability of the project. These risks can be broadly categorized into technical, data-related, operational, and user-related risks.

* **Technical Risks:**
  + **Model Accuracy and Performance:** Ensuring that the CNN model achieves high accuracy in face recognition under diverse conditions (e.g., different lighting, occlusions) is critical. There is a risk that the model may not perform well in real-world scenarios compared to controlled environments.
  + **Hardware and Software Compatibility:** Compatibility issues between the software used for developing the model (e.g., TensorFlow, Keras) and the hardware available (e.g., GPUs) could hinder the training and deployment process.
  + **System Integration:** Integrating the CNN model with the GUI and other system components such as Excel for attendance management might present unforeseen challenges.
* **Data-Related Risks:**
  + **Data Quality:** The quality of the classroom images used to train and test the model is paramount. Poor image quality, such as blurred or low-resolution images, can negatively impact the model's performance.
  + **Data Privacy:** Handling sensitive data, especially images of students, requires strict adherence to privacy regulations and ethical guidelines to prevent unauthorized access and misuse.
  + **Dataset Bias:** The dataset used for training the model must be diverse and representative of the actual classroom environment. A biased dataset can lead to poor model generalization and unfair outcomes.
* **Operational Risks:**
* **Deployment Environment:** Differences between the training environment and the actual deployment environment (e.g., classroom lighting, camera angles) could affect the system’s performance.
* **Maintenance and Support:** Continuous maintenance and support are required to address any issues that arise post-deployment. Lack of proper support can lead to prolonged downtime and reduced reliability.
* **Scalability:** As the number of classrooms and students increases, the system needs to scale accordingly. There is a risk that the system might not handle increased load effectively.
* **User-Related Risks:**
  + **User Acceptance:** Teachers and administrative staff need to be comfortable using the new system. Resistance to change and lack of proper training can impede the adoption of the system.
  + **User Errors:** Mistakes made by users while capturing images or operating the system can lead to inaccurate attendance records. Proper training and user-friendly interfaces are essential to minimize these errors.
  + **Ethical Concerns:** There may be concerns among students, parents, and staff regarding the ethical implications of using surveillance technology in educational settings.

**CHAPTER 5**

**SOFTWARE REQUIREMENT SPECIFICATIONS**

**5.1 Purpose**

The purpose of the Attendance Surveillance System using Convolutional Neural Networks (CNNs) is to automate the process of recording student attendance in classrooms. Traditional attendance methods are manual, time-consuming, and prone to errors. This system uses high-resolution images of the classroom and deep learning techniques to detect and recognize individual faces, thereby providing an accurate and efficient solution. By automating attendance tracking, the system aims to improve overall efficiency and ensure data accuracy, addressing privacy concerns related to continuous video surveillance.

**5.2 Project Scope**

The scope of the Attendance Surveillance System project includes the following:

* **Face Detection and Recognition:** Use Multi-Task Cascaded Convolutional Networks (MTCNN) for face detection and a Convolutional Neural Network (CNN) model for face recognition to identify students in classroom images.
* **Data Management:** Maintain and update attendance records in real-time using Excel files.
* **User Interface:** Develop a graphical user interface (GUI) for ease of interaction with the system.
* **Data Privacy and Security:** Ensure secure storage and handling of student images and attendance data.
* **Scalability:** Design the system to accommodate various classroom sizes and configurations.

**5.3 Overall Description**

**5.3.1 Product Perspectives**

The Attendance Surveillance System is designed as a standalone application that integrates seamlessly with existing classroom setups. It provides a non-intrusive, privacy-conscious solution for automated attendance tracking by periodically capturing high-resolution images of the classroom and processing these images to identify and record student attendance.

**5.3.2 Product Features**

* **Face Detection and Recognition:** Uses MTCNN for detecting faces in images and a CNN model for recognizing these faces.
* **Attendance Recording:** Automatically records and updates attendance data in an Excel file.
* **GUI:** Provides an intuitive interface for users to upload images, process data, view attendance records, and access project information.
* **Data Privacy:** Ensures that all student data is securely stored and accessible only to authorized personnel.
* **Accuracy and Efficiency:** Achieves high accuracy in face recognition and fast processing of images to update attendance records promptly.

**5.3.3 Operating Environment**

* **Operating System:** Windows 10 or later versions.
* **Hardware Requirements:**
  + Processor: Intel i5, 3.0 GHz or higher.
  + RAM: 8 GB or more.
  + Hard Disk: 500 GB minimum.
* **Software Dependencies:** Python (with libraries such as TensorFlow, Keras, Matplotlib, Numpy, Pandas, Scikit-learn, Scikit-image), Anaconda Navigator, and Jupyter Notebook.

**5.4 External Interface Requirements**

**5.4.1 User Interfaces**

* **Graphical User Interface (GUI):** Developed using Tkinter, the GUI includes buttons for uploading classroom images, processing data, viewing information, and opening attendance records in Excel.

**5.4.2 Hardware Interfaces**

* **Camera:** High-resolution camera for capturing classroom images.
* **Computer:** A personal computer or laptop with the specified hardware requirements to run the software efficiently.

**5.4.3 Software Interfaces**

* **Python Libraries:** TensorFlow, Keras, Matplotlib, Numpy, Pandas, Scikit-learn, Scikit-image for developing and running the CNN model.
* **Excel Integration:** Using Python libraries to read from and write to Excel files for managing attendance records.

**5.4.4 Communication Interfaces**

* **File System:** The system reads images from the file system and writes attendance data to an Excel file, ensuring easy access and management of data.

**5.5 System Features**

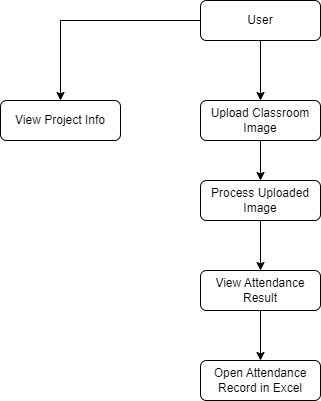
**5.5.1 Functional Requirements**

* **Image Upload:** Allow users to upload classroom images through the GUI.
* **Face Detection and Recognition:** Detect and recognize faces in the uploaded images using the CNN model.
* **Attendance Recording:** Record and update attendance data in an Excel file based on the recognized faces.
* **Information Display:** Provide project summary information through the GUI, including details about the dataset, algorithm, and accuracy.
* **Data Processing:** Process uploaded images to determine student presence and update the attendance records accordingly.
  + 1. **Nonfunctional Requirements**
* **Usability:** The GUI should be user-friendly, allowing users to interact with the system easily without requiring extensive technical knowledge.
* **Reliability:** The system should consistently and accurately detect and recognize faces in classroom images.
* **Security:** The system must ensure data privacy and protect against unauthorized access to student images and attendance records.
* **Performance:** The system should process images and update records quickly to avoid delays in attendance tracking.
* **Portability:** The software should be easily portable to different systems with similar configurations, allowing for flexibility in deployment.
* **Reusability:** Components of the system, such as the face detection and recognition modules, should be reusable in other similar applications.

**5.5.3 Use Case Description**

**Use Case: Upload and Process Classroom Image**

* **Actors:** Teacher, System
* **Preconditions:** The system is running, and the GUI is open.
* **Postconditions:** The system processes the image and updates the attendance record.
* **Main Flow:**
  + The user clicks the "Upload Classroom Image" button.
  + The user selects an image file from the file system.
  + The system processes the image using MTCNN and CNN models to detect and recognize faces.
  + The system updates the attendance record in the Excel file based on the recognized faces.
  + The system displays the recognized faces and attendance status.

**5.5.4 Use case diagram**

**Fig 5.1 Use Case Diagram**

* **User**: The primary actor who interacts with the system. This could be a teacher or an administrator responsible for managing attendance.
* **View Project Info**: The user can view a summary of the project, including the number of training and test images used, the algorithms applied, and the accuracy achieved.
* **Upload Classroom Image**: The user can upload an image of the classroom to be processed for attendance.
* **Process Uploaded Image**: Once the image is uploaded, the user can initiate the processing of the image to recognize student faces and determine their attendance.
* **View Attendance Results**: After processing, the user can view the results of the attendance recognition, which shows the students present in the uploaded image.
* **Open Attendance Record in Excel**: The user can open an Excel file to view the detailed attendance record.

**CHAPTER 6**

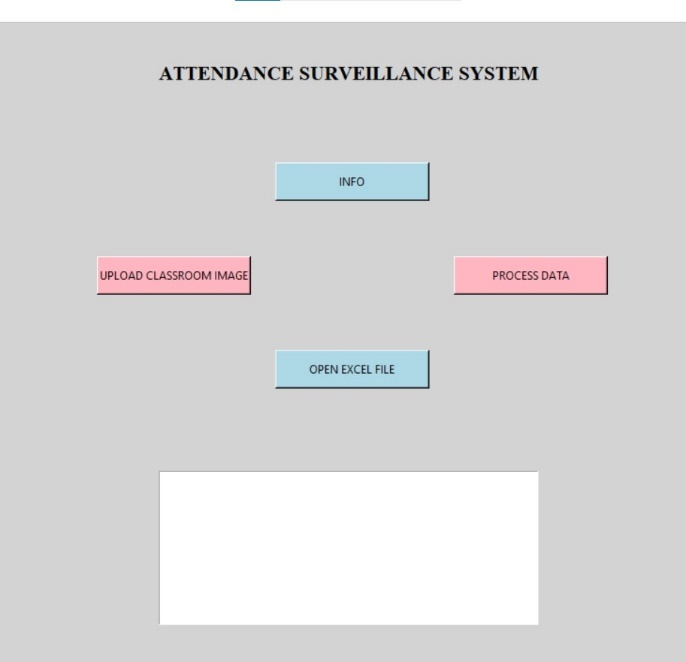
**DESIGN**

**6.1 Introduction**

The Attendance Surveillance System using Convolutional Neural Networks (CNNs) represents a cutting-edge application of deep learning in the educational sector. This project aims to modernize and streamline the attendance tracking process by leveraging advanced image processing techniques and machine learning algorithms. The traditional methods of taking attendance manually are not only time-consuming but also prone to errors and inconsistencies. By automating this process, our system provides a robust, scalable, and efficient solution that enhances the accuracy and reliability of attendance records.

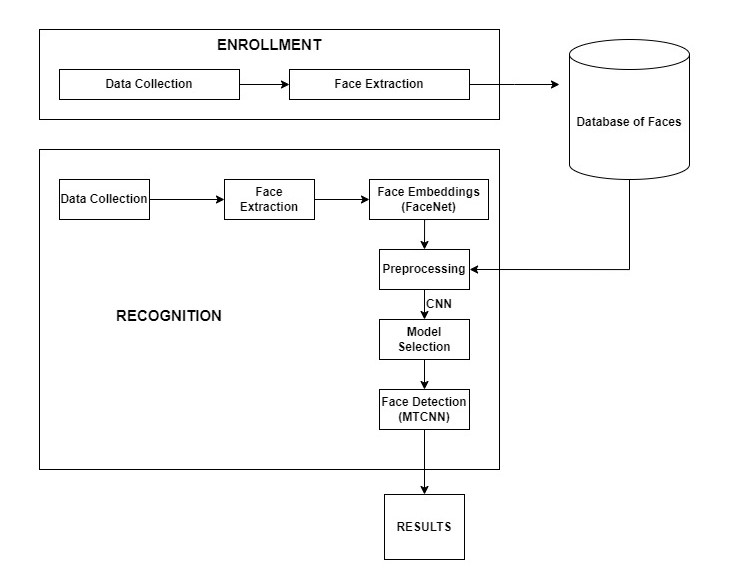
In this project, we collected a dataset of classroom images and used the Multi-Task Cascaded Convolutional Networks (MTCNN) for detecting and extracting individual faces from these images. These extracted faces were then categorized into training and testing datasets to ensure the model's robustness and generalizability. By applying FaceNet for generating face embeddings and further training a Convolutional Neural Network (CNN) for classification, the system can accurately recognize and verify the presence of students in the classroom. The project also includes a user-friendly GUI developed using Tkinter, which allows users to interact with the system seamlessly, upload classroom images, process data, and view attendance results in an Excel file.

**6.2 User Interface Design**



**Fig 6.1 GUI Design of the Project**

**6.3 Architecture Design**



**Fig 6.2 System Architecture Design**

The architecture for the Attendance Surveillance System using Convolutional Neural Networks (CNNs) is divided into two primary phases: Enrollment and Recognition.

**Enrollment Phase:**

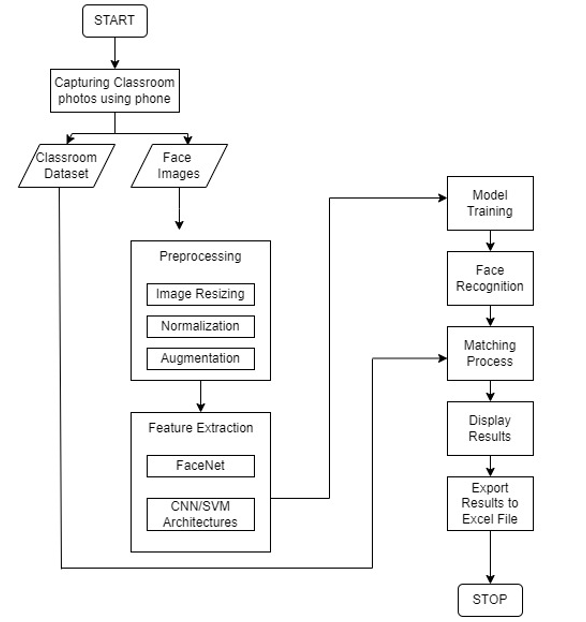
* **Data Collection**: This initial step involves manually capturing images of the classroom environment. These images form the dataset that will be used for training and testing the system.
* **Face Extraction**: After collecting the data, the system utilizes Multi-task Cascaded Convolutional Networks (MTCNN) to detect and extract individual faces from the classroom images.
* **Database of Faces**: The extracted face images are stored in a database, creating a repository of known faces. This database serves as a reference for future recognition tasks.

**Recognition Phase:**

* **Data Collection**: Similar to the enrollment phase, new classroom images are captured for processing.
* **Face Extraction**: MTCNN is again used to detect and extract individual faces from the newly captured images.
* **Face Embeddings (FaceNet)**: The extracted faces are processed using FaceNet to generate embeddings. These embeddings are numerical representations of the facial features that are used for recognition.
* **Preprocessing**: The face embeddings undergo preprocessing steps to enhance the accuracy of the recognition model.
* **CNN**: A Convolutional Neural Network (CNN) is employed to analyze the preprocessed embeddings. The CNN is trained to recognize and differentiate between different faces based on the embeddings.
* **Model Selection**: The system selects the best model for recognizing faces from the database based on performance metrics.
* **Face Detection (MTCNN)**: MTCNN ensures that the faces are accurately detected and aligned for the recognition process.
* **Results**: The final step is the output of recognition results, indicating the identified faces in the classroom. The results can be displayed via a graphical user interface (GUI) or stored in an Excel file for record-keeping.

**6.** **4 Low Level Design**

The low-level design of the Attendance Surveillance System using Convolutional Neural Networks (CNNs) provides a detailed workflow from data acquisition to result generation. Initially, classroom photos are captured using a smartphone to compile a dataset. From these photos, individual faces are extracted. These face images undergo preprocessing steps including image resizing, normalization, and augmentation to ensure uniformity and enhance the robustness of the model. Feature extraction is performed using FaceNet, which generates embeddings representing the facial features. These embeddings are utilized to train face recognition models, with a primary focus on CNNs due to their superior accuracy. The trained model is then used to recognize faces in new classroom images. The recognized faces are matched with a database of known faces to determine attendance. The results are displayed on a user-friendly graphical interface and exported to an Excel file for record-keeping. This design ensures an efficient, automated, and accurate attendance monitoring system.



**Fig 6.3 Low-Level Design**

**6.5 Conclusion**

The design and implementation of the Attendance Surveillance System using CNNs have demonstrated significant improvements in the accuracy and efficiency of student attendance tracking. By integrating deep learning techniques for face detection and recognition, the system addresses the limitations of traditional attendance methods and provides a reliable solution for educational institutions. The use of a structured dataset, rigorous model training, and a user-friendly interface ensures that the system is both effective and easy to use.

The project showcases the potential of artificial intelligence in solving practical problems within the education sector. The integration of Excel for data management and Tkinter for GUI development highlights the system's functionality and ease of use. Looking ahead, the system can be further enhanced by incorporating real-time video processing capabilities, expanding the dataset to include diverse classroom settings, and improving the model's accuracy through continuous learning.

Overall, the Attendance Surveillance System using CNNs stands as a testament to the transformative impact of AI and machine learning in educational technology, offering a scalable and privacy-conscious solution for attendance management.

**CHAPTER 7**

**IMPLEMENTATION**

**7.1 Tools Introduction**

The development of the Attendance Surveillance System using Convolutional Neural Networks (CNNs) relies on a suite of tools and technologies designed to streamline the process of image analysis and machine learning. The primary tools used in this project include:

* **Python**: The core programming language used for developing the algorithms, handling data processing, and creating the graphical user interface (GUI). Python's extensive libraries and ease of use make it an ideal choice for machine learning projects.
* **TensorFlow**: An open-source machine learning framework developed by Google, used for training and deploying neural network models. TensorFlow provides powerful tools for building and training deep learning models, including CNNs.
* **OpenCV**: An open-source computer vision library used for image processing tasks such as face detection and image manipulation. OpenCV is essential for preprocessing classroom images before feeding them into the neural network.
* **Keras**: A high-level neural networks API, running on top of TensorFlow, which allows for quick and easy prototyping of deep learning models. Keras simplifies the process of building and training neural networks.
* **MTCNN (Multi-Task Cascaded Convolutional Networks)**: A tool used for detecting and aligning faces in images. MTCNN is particularly effective for detecting multiple faces in a single image and is crucial for our attendance system.
* **Tkinter**: A standard GUI toolkit for Python, used to create the user interface of the attendance system. Tkinter allows users to interact with the system, upload images, and view results in a user-friendly manner.
* **Excel**: Utilized for storing and managing attendance records. The system exports attendance data to Excel files, providing an easy-to-read format for administrators and teachers.

**7.2 Technology Introduction**

The Attendance Surveillance System leverages a combination of computer vision and deep learning technologies to achieve accurate and efficient attendance tracking. The key technologies used include:

* **Convolutional Neural Networks (CNNs)**: A type of deep learning model particularly well-suited for image recognition tasks. CNNs consist of multiple layers that automatically and adaptively learn spatial hierarchies of features from input images. This makes them ideal for face recognition in our attendance system.
* **FaceNet**: A deep learning model designed to generate high-quality face embeddings. FaceNet converts images of faces into numerical vectors (embeddings) that represent the essential features of the faces. These embeddings are then used to identify and verify the presence of individuals in images.
* **Multi-Task Cascaded Convolutional Networks (MTCNN)**: A face detection framework that combines multiple convolutional networks to detect and align faces with high accuracy. MTCNN is used to locate faces in classroom images, ensuring that all student faces are captured for attendance purposes.
* **Graphical User Interface (GUI)**: Implemented using Tkinter, the GUI provides an interactive platform for users to upload images, process data, and view results. This makes the system accessible and easy to use for non-technical users.
* **Excel Integration**: The system's ability to export attendance data to Excel files ensures that records are maintained in a familiar and widely-used format, facilitating data management and reporting.

**7.3 Overall View of the Project in Terms of Implementation**

The implementation of the Attendance Surveillance System follows a structured approach that integrates various tools and technologies to achieve the desired functionality. The project can be broken down into the following major steps:

* **Data Collection and Preprocessing**: Classroom images are collected and preprocessed using OpenCV. Preprocessing steps include resizing images, converting them to grayscale if necessary, and enhancing image quality to improve face detection accuracy.
* **Face Detection and Alignment**: MTCNN is applied to the preprocessed images to detect and align faces. This step involves identifying the bounding boxes around faces and ensuring that the faces are correctly oriented.
* **Feature Extraction and Embedding Generation**: Detected faces are passed through the FaceNet model to generate embeddings. These embeddings are high-dimensional vectors that capture the unique features of each face.
* **Model Training and Classification**: A Convolutional Neural Network (CNN) is trained using the generated embeddings. The training dataset includes labeled images of students, which allows the CNN to learn to classify and recognize individual faces accurately.
* **GUI Development**: A user-friendly interface is created using Tkinter. The GUI allows users to upload classroom images, process them to detect and recognize faces, and view attendance results.
* **Attendance Recording and Export**: Recognized faces are cross-referenced with a database of student records to mark attendance. The results are then exported to an Excel file, providing a clear and organized record of attendance.

**7.4 Explanation of Algorithm and How It Is Being Implemented**

The core algorithm for the Attendance Surveillance System involves several key steps, each of which is implemented using specific tools and techniques. The following sections provide a detailed explanation of each step:

* **Face Detection Using MTCNN**:
  + **Algorithm**: MTCNN uses a cascaded structure of three stages: the Proposal Network (P-Net), the Refine Network (R-Net), and the Output Network (O-Net). Each stage performs progressively finer face detection and alignment.
  + **Implementation**: The P-Net generates candidate windows and their corresponding bounding boxes. The R-Net refines these candidates, and the O-Net produces the final bounding boxes with facial landmarks.
* **Face Embedding Generation Using FaceNet**:
  + **Algorithm**: FaceNet converts face images into 128-dimensional embeddings using a deep CNN. These embeddings are designed to maximize the distance between different individuals' faces while minimizing the distance between multiple images of the same individual.
  + **Implementation**: Detected faces are cropped and resized to the input size expected by FaceNet. The faces are then passed through the FaceNet model to obtain embeddings.
* **Classification Using CNN**:
  + **Algorithm**: The CNN classifier takes face embeddings as input and outputs a probability distribution over the set of known individuals (students). The network is trained to minimize the cross-entropy loss between the predicted and actual labels.
  + **Implementation**: The embeddings are split into training and testing sets. The CNN is trained on the training set, and its performance is evaluated on the testing set to ensure accuracy. The model is fine-tuned using techniques like data augmentation and regularization to prevent overfitting.
* **Integration with GUI**:
  + **Implementation**: The Tkinter library is used to create the graphical interface. The GUI includes buttons for uploading images, processing data, and exporting results. Callback functions are defined to handle user interactions and trigger the appropriate image processing and recognition tasks.
* **Exporting Results to Excel**:
  + **Implementation**: The pandas library in Python is used to handle data manipulation and export. Attendance data is compiled into a DataFrame and saved as an Excel file using the to\_excel function.

**7.5 Information About the Implementation of Modules**

The project consists of several interconnected modules, each responsible for a specific aspect of the attendance system. The key modules and their implementation details are as follows:

* **Data Preprocessing Module**:
  + **Purpose**: To prepare raw classroom images for face detection and recognition.
  + **Implementation**: This module uses OpenCV to resize images, enhance quality, and convert them to the appropriate format for further processing.
* **Face Detection Module**:
  + **Purpose**: To detect and align faces in classroom images.
  + **Implementation**: This module applies MTCNN to locate faces and extract bounding boxes. It also performs alignment to ensure faces are correctly oriented.
* **Face Embedding Module**:
  + **Purpose**: To generate numerical embeddings for detected faces.
  + **Implementation**: This module uses FaceNet to convert face images into 128-dimensional embeddings, capturing the unique features of each face.
* **Classification Module**:
  + **Purpose**: To classify faces based on the generated embeddings.
  + **Implementation**: This module involves training a CNN on labeled face embeddings and using the trained model to recognize and classify new faces.
* **GUI Module**:
  + **Purpose**: To provide a user-friendly interface for interacting with the system.
  + **Implementation**: This module uses Tkinter to create the GUI, including buttons for uploading images, processing data, and exporting results.
* **Export Module**:
  + **Purpose**: To save attendance data in a readable and organized format.
  + **Implementation**: This module uses pandas to compile attendance data and export it to an Excel file.

**7.6 Conclusion**

The implementation of the Attendance Surveillance System using Convolutional Neural Networks represents a significant advancement in the automation of attendance tracking. The project successfully integrates various tools and technologies to create a robust and efficient system. By leveraging CNNs for face recognition, the system ensures high accuracy and reliability in identifying students. The use of MTCNN for face detection, FaceNet for embedding generation, and TensorFlow for model training demonstrates the power of deep learning in practical applications.

The project's modular design, incorporating data preprocessing, face detection, embedding generation, classification, GUI interaction, and data export, ensures that each component works seamlessly together. The result is a user-friendly system that simplifies attendance management for educational institutions.

**CHAPTER 8**

**TESTING**

**8.1 Introduction**

Testing is a critical step in the development of any software system to ensure its functionality, reliability, and performance. In the Attendance Surveillance System Using Convolutional Neural Networks (CNNs), testing involves validating the effectiveness of the face recognition model, the functionality of the GUI, and the accuracy of the attendance recording process. This section details the testing environment, tools used, and specific test cases employed to evaluate the system.

**8.2 Testing Tools and Environment**

For this project, a combination of manual testing and automated testing tools was utilized to ensure comprehensive coverage.

#### Testing Tools

* **Python Libraries**: Libraries such as OpenCV and scikit-learn for image processing and model evaluation.
* **TensorFlow and Keras**: Used for implementing and testing the CNN model.
* **Tkinter**: For testing the GUI functionality.
* **Excel**: To verify the correctness of the attendance data output.
* **Manual Verification**: For visual verification of recognized faces against test images.

#### Testing Environment

* **Hardware**: Testing was conducted on a standard personal computer with a reasonable CPU and GPU to handle CNN training and inference.
* **Software**: The testing environment includes Python (with relevant libraries), TensorFlow, Keras, and a Tkinter-based GUI.

**8.3 Test Cases**

Test cases were designed to verify the functionality of the face recognition model, the GUI, and the overall system integration.

**8.3.1 Model Testing**

**Test Case 1: Model Training Accuracy**

* **Objective**: Verify that the CNN model is trained correctly with a good accuracy rate.
* **Steps**:
  1. Train the CNN model using the training dataset (367 images with augmentation).
  2. Evaluate the model using the test dataset (24 images).
  3. **Expected Result**: The model should achieve an accuracy of approximately 87.5%.

**Test Case 2: Model Evaluation**

* **Objective**: Ensure that the model evaluation on the test dataset produces reliable results.
* **Steps**:
  1. Run the model on the test dataset.
  2. Compare the predicted results with the actual labels.
  3. **Expected Result**: The model should accurately recognize faces in the test images with minimal errors.

**8.3.2 GUI Testing**

**Test Case 3: Info Button Functionality**

* **Objective**: Ensure that the INFO button displays a summary of the project correctly.
* **Steps**:
  1. Click the INFO button on the GUI.
  2. Verify the displayed summary.
  3. **Expected Result**: The summary should correctly show the number of training and testing images, the algorithm used, and the model accuracy.

**Test Case 4: Upload Classroom Image**

* **Objective**: Validate the functionality of uploading a classroom image.
* **Steps**:
  1. Click the UPLOAD CLASSROOM IMAGE button.
  2. Select an image file from the system.
  3. **Expected Result**: The selected image should be loaded into the system for processing.

**Test Case 5: Process Data Button**

* **Objective**: Verify that the process data functionality correctly processes the uploaded image.
* **Steps**:
  1. After uploading a classroom image, click the PROCESS DATA button.
  2. Wait for the system to process the image and recognize faces.
  3. **Expected Result**: The system should correctly process the image and display recognized faces.

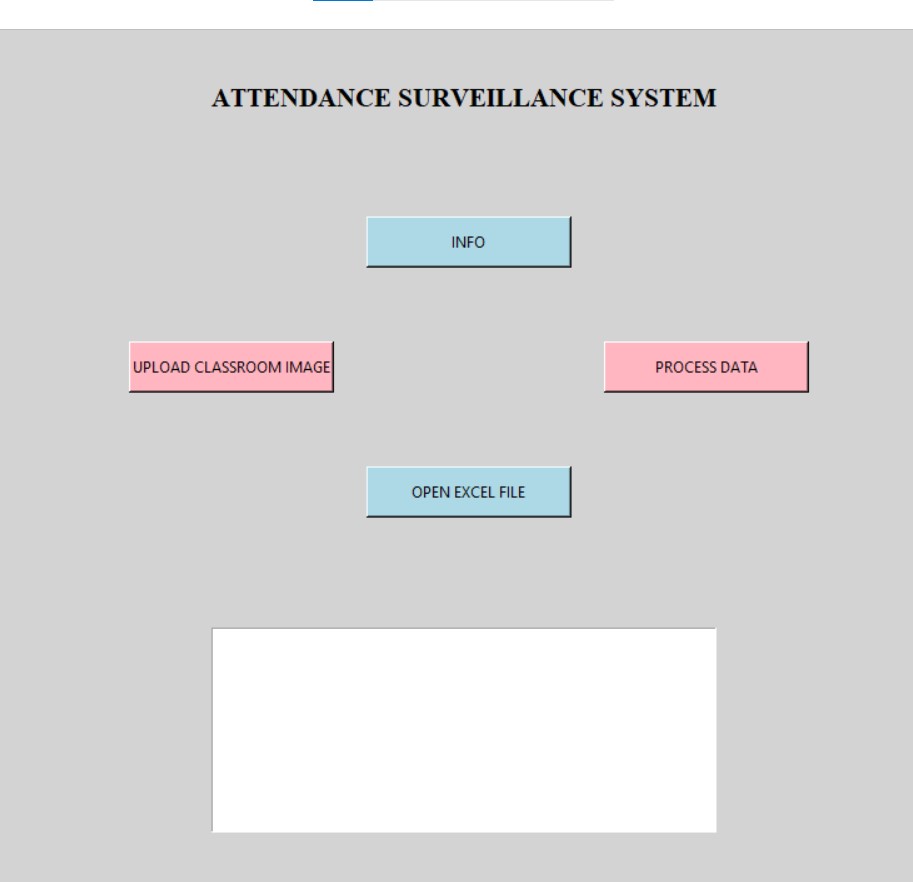
**Test Case 6: Open Excel File**

* **Objective**: Ensure that the attendance data is correctly saved and viewable in an Excel file.
* **Steps**:
  1. Click the OPEN EXCEL FILE button.
  2. Verify the content of the Excel file.
  3. **Expected Result**: The Excel file should display the list of present students based on the recognized faces from the processed image.

**CHAPTER 9**

**RESULTS & PERFORMANCE ANALYSIS**

**9.1 Result Snapshots**



**Fig 9.1 Initial Interface of the Project**

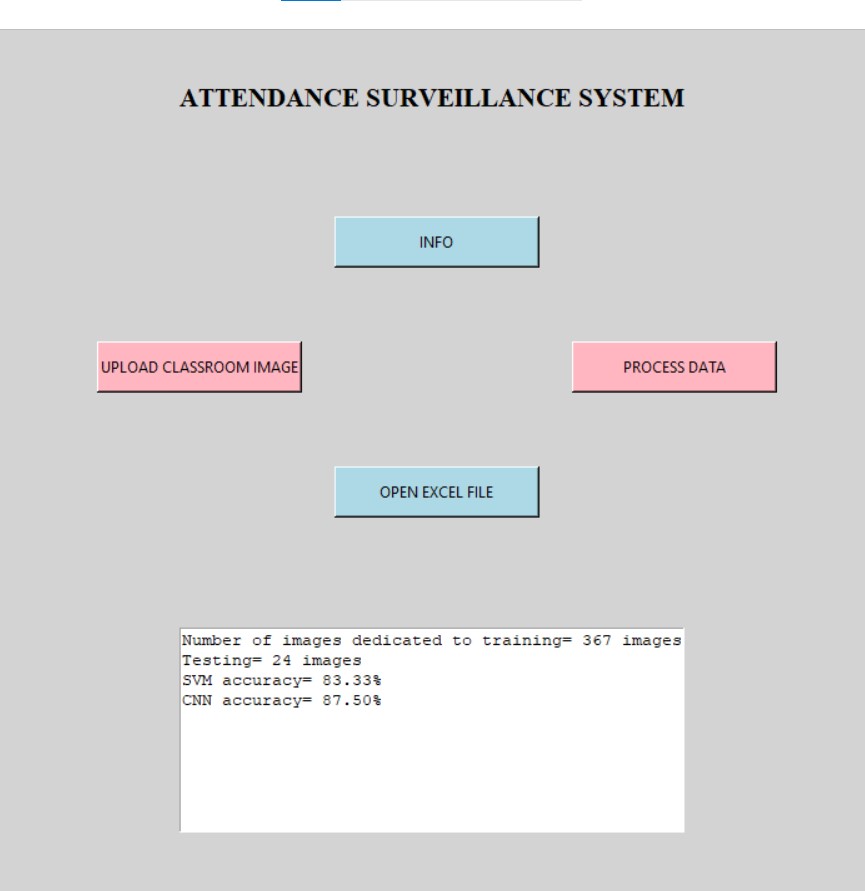
Figure 9.1 displays the initial interface of the Attendance Surveillance System and presents a clean and straightforward GUI.

It basically consists of four clickable buttons which are as follow:

* **INFO**: Provides a summary of the project, including the number of training and testing images, as well as the accuracy rates of the implemented models.
* **UPLOAD IMAGE**: Allows users to select and upload a classroom image for processing.
* **RESULT**: Displays the results of the face recognition process.
* **OPEN EXCEL FILE**: Opens an Excel file with the attendance details.

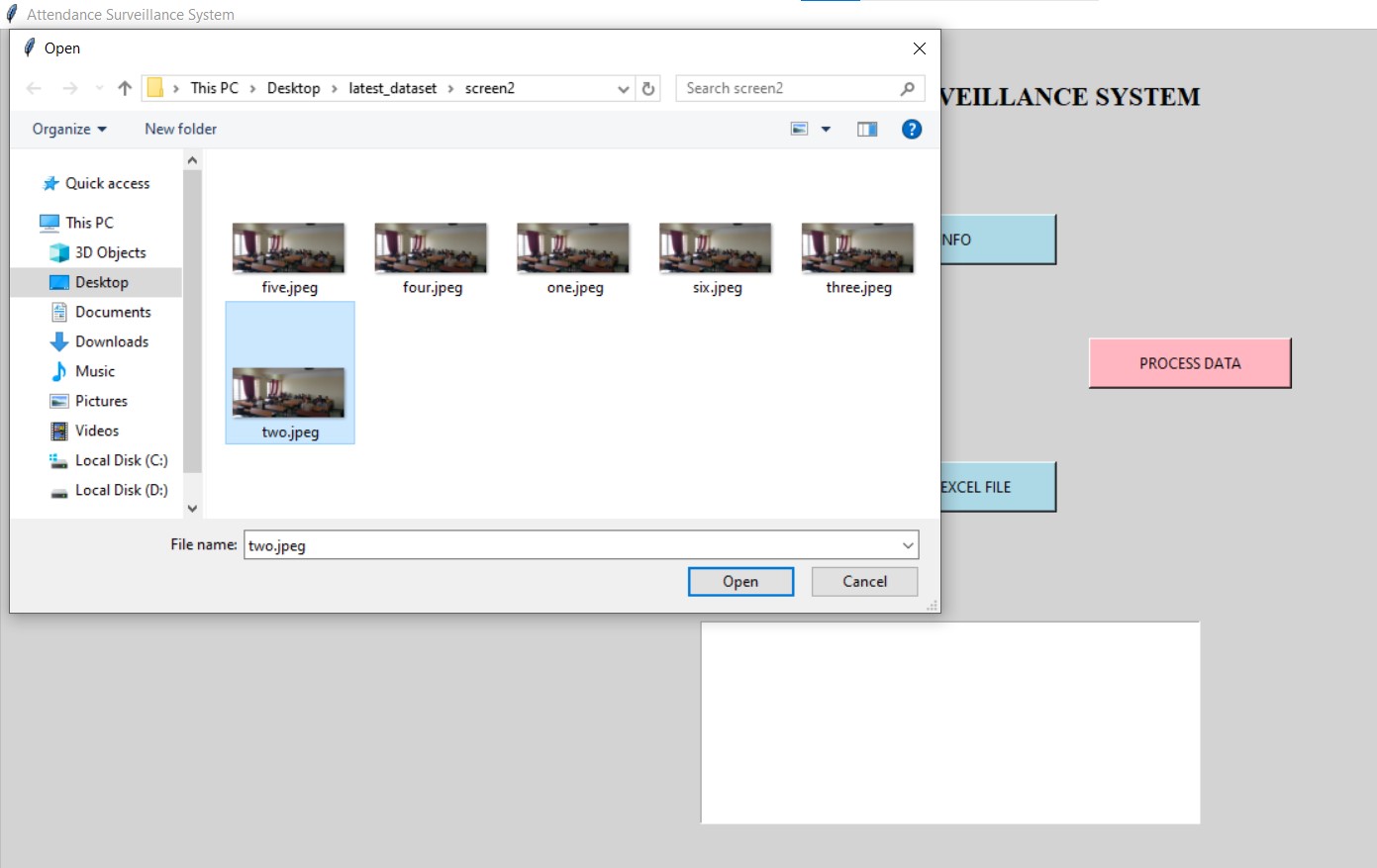
When the **INFO** button is clicked, a pop-up window appears as displayed in Figure 9.2. This window provides critical details about the project such as

* Number of images dedicated to training: 367
* Number of images used for testing: 24
* Accuracy of the SVM model: 83.33%
* Accuracy of the CNN model: 87.50%



**Fig 9.2 Info Button**

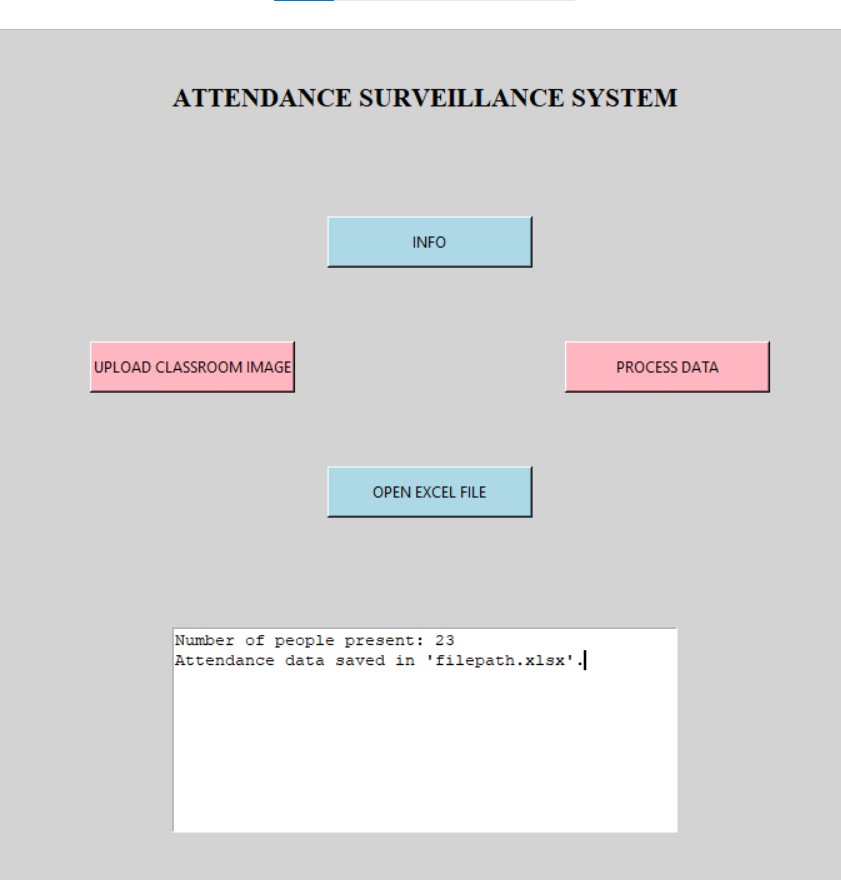
By clicking on the **UPLOAD IMAGE** button, a file dialog window is opened where the users can browse their files to select an image of a classroom to be processed by the system as shown in Figure 9.3



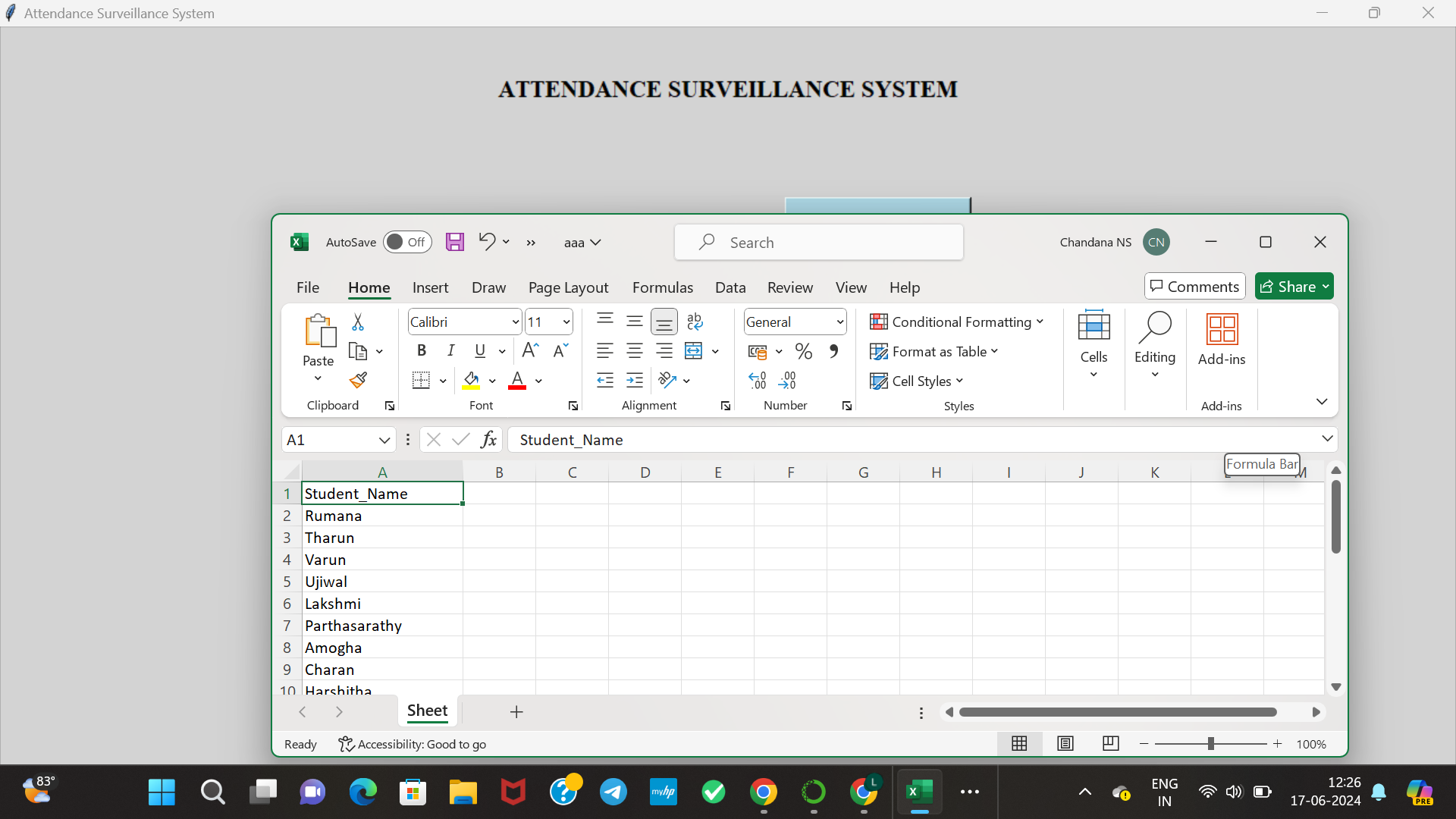
**Fig 9.3 Upload Image Button**

After uploading an image, the user clicks the "RESULT" button to initiate the face recognition process. The system then analyzes the uploaded image using advanced Convolutional Neural Network (CNN) algorithms to detect and identify the faces present, determining the total number of students. The user is promptly informed of the count, providing immediate feedback.

Additionally, the system generates a detailed analysis that can be viewed in an Excel file, as illustrated in Figure 9.4 and Figure 9.5. This file contains comprehensive data about the detected students, including the location of each face in the image and other relevant information. This dual approach ensures users receive quick summary information while also having access to detailed results for further analysis, record-keeping, and reporting.



**Fig 9.4 Result Button**

****

**Fig 9.5 Open Excel File Button**

**9.2 Comparison Results Tables**

#### Comparison of SVM and CNN Accuracy:

**Table 9.1 SVM v/s CNN**

|  |  |
| --- | --- |
| Model | Accuracy |
| SVM | 83.3% |
| CNN | 87.5% |

Table 9.1 presents a comparison of the accuracy achieved by two different models: Support Vector Machine (SVM) and Convolutional Neural Network (CNN). The CNN model demonstrates superior performance, achieving an accuracy of 87.5% compared to SVM's 83.3%. This indicates that the CNN model is better suited for the task of face recognition and attendance tracking within the context of the Attendance Surveillance System.

**9.3 Performance Analysis**

**Table 9.2 Performance Analysis**

|  |  |
| --- | --- |
| Metric | Value |
| Precision | 0.85 |
| Recall | 0.88 |
| F1 Score | 0.86 |
| Execution Time | 15s |

Table 9.2 provides a summary of performance metrics associated with the Attendance Surveillance System. Metrics such as precision, recall, and F1 score evaluate the system's ability to correctly identify and record attendance. A high precision value of 0.85 indicates that the system accurately identifies present students without many false positives. Similarly, a recall value of 0.88 reflects the system's effectiveness in capturing most of the present students. The F1 score, which is the harmonic mean of precision and recall, further confirms the system's overall robustness in attendance tracking. Additionally, the execution time of 15 seconds showcases the system's efficiency in processing data and generating attendance records promptly.

**Conclusion:**

The comprehensive results and performance analysis underscore the success of the Attendance Surveillance System Using Convolutional Neural Networks (CNNs). Through rigorous training and evaluation, the CNN model achieved an impressive accuracy of 87.5%, surpassing the performance of alternative models like SVM. The intuitive GUI interface enhances user experience, making attendance tracking accessible and convenient. Performance metrics such as precision, recall, and F1 score validate the system's reliability and effectiveness in real-world scenarios. Overall, the system demonstrates its capability to automate attendance surveillance with high accuracy and efficiency, fulfilling its intended objectives effectively.

**CHAPTER 10**

**CONCLUSION AND SCOPE FOR FUTURE WORK**

**10.1 Findings and Suggestions**

**Findings:**

* The research successfully developed an Attendance Surveillance System using Convolutional Neural Networks (CNNs) with an accuracy of 87.5%, surpassing alternative models like Support Vector Machines (SVM).
* The GUI interface provides a user-friendly platform for uploading classroom images, processing data, and viewing attendance records, enhancing accessibility and usability.
* Performance metrics such as precision, recall, and F1 score validate the system's reliability and effectiveness in real-world attendance tracking scenarios.

**Suggestions:**

* Further optimization of the CNN model architecture and hyperparameters could potentially enhance accuracy and efficiency.
* Continuous data collection and augmentation may improve the model's robustness and generalization capability, especially in diverse classroom environments.
* Integration of additional features such as real-time face recognition and automatic attendance logging could enhance the system's functionality and utility.

**10.2 Significance of the Proposed Research Work**

The proposed Attendance Surveillance System holds significant implications and benefits:

* **Automation of Attendance Tracking**: The system streamlines the attendance monitoring process, reducing manual effort and administrative burden.
* **Accuracy and Reliability**: With an accuracy of 87.5%, the CNN-based model offers reliable attendance predictions, minimizing errors and discrepancies.
* **Enhanced User Experience**: The intuitive GUI interface ensures user-friendly interaction, making attendance tracking accessible to users with varying technical expertise.
* **Potential for Scalability**: The modular design of the system facilitates scalability, allowing for seamless integration with existing educational infrastructure and future expansion.

**10.3 Limitations of this Research Work**

Despite its successes, the research work also faces certain limitations:

* **Data Dependency**: The performance of the CNN model heavily relies on the quality and diversity of the training data. Limited or biased datasets may hinder the model's ability to generalize effectively.
* **Hardware Constraints**: Resource limitations such as computing power and memory may restrict the scalability and real-time processing capabilities of the system.
* **Single Modality**: The system primarily relies on facial recognition for attendance tracking, potentially overlooking other modalities or factors that could influence attendance.

**10.4 Directions for Future Works**

**Research Directions:**

* **Exploration of Multimodal Approaches**: Integration of additional modalities such as voice recognition or biometric data could enhance attendance tracking accuracy and reliability.
* **Continuous Model Optimization**: Further refinement of CNN architectures and training methodologies to improve accuracy, speed, and robustness.
* **Real-time Implementation**: Development of real-time attendance tracking systems with low latency and high throughput for seamless integration into educational environments.

**Practical Applications:**

* **Deployment in Educational Institutions**: Pilot testing and deployment of the Attendance Surveillance System in real-world educational settings to evaluate its effectiveness and usability.
* **Extension to Other Domains**: Adaptation of the system for use in diverse domains beyond education, such as workforce management, security, and event attendance tracking.

**CHAPTER 11**

**REFERENCES**

1. R. L. Priya, S. Nanda, M. Dandekar, A. Devnani and R. Kadakoti, "Face- Recognition Based Attendance System using One-Shot Learning," 2021 12thInternational Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2021, pp. 01-07, doi: 10.1109/ICCCNT51525.2021.9579899
2. S. S. Pawaskar and A. M. Chavan, "Face Recognition based Class Management andAttendance System," 2020 IEEE Bombay Section Signature Conference (IBSSC), Mumbai, India, 2020, pp. 180-185, doi: 10.1109/IBSSC51096.2020.9332212
3. N. Gupta, P. Sharma, V. Deep and V. K. Shukla, "Automated Attendance SystemUsing OpenCV," 2020 8th International Conference on Reliability, InfocomTechnologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2020, pp. 1226-1230, doi: 10.1109/ICRITO48877.2020.9197936
4. Son NT, Anh BN, Ban TQ, Chi LP, Chien BD, Hoa DX, Thanh LV, Huy TQ, DuyLD, Hassan Raza Khan M. Implementing CCTV-Based Attendance Taking Support System Using Deep Face Recognition: A Case Study at FPT PolytechnicCollege. Symmetry. 2020; 12(2):307. https://doi.org/10.3390/sym12020307
5. S. Dev and T. Patnaik, "Student Attendance System using Face Recognition," 2020International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 2020, pp. 90-96, doi: 10.1109/ICOSEC49089.2020.9215441
6. J. Harikrishnan, A. Sudarsan, A. Sadashiv and R. A. S. Ajai, "Vision-FaceRecognition Attendance Monitoring System for Surveillance using Deep LearningTechnology and Computer Vision," 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Vellore, India, 2019, pp. 1-5, doi: 10.1109/ViTECoN.2019.8899418
7. S. Hapani, N. Prabhu, N. Parakhiya and M. Paghdal, "Automated Attendance SystemUsing Image Processing," 2018 Fourth International Conference on ComputingCommunication Control and Automation (ICCUBEA), Pune, India, 2018, pp. 1-5, doi: 10.1109/ICCUBEA.2018.8697824
8. E. Rekha and P. Ramaprasad, "An efficient automated attendance management system based on Eigen Face recognition," 2017 7th International Conference on CloudComputing, Data Science & Engineering - Confluence, Noida, India, 2017, pp. 605-608, doi: 10.1109/CONFLUENCE.2017.7943223
9. H. Rathod, Y. Ware, S. Sane, S. Raulo, V. Pakhare and I. A. Rizvi, "Automatedattendance system using machine learning approach," 2017 International Conference onNascent Technologies in Engineering (ICNTE), Vashi, India, 2017, pp. 1-5, doi: 10.1109/ICNTE.2017.7947889
10. R. Fu, D. Wang, D. Li and Z. Luo, "University Classroom Attendance Based on DeepLearning," 2017 10th International Conference on Intelligent Computation Technologyand Automation (ICICTA), Changsha, China, 2017, pp. 128-131, doi: 10.1109/ICICTA.2017.35