Institute of
Space Technology

Artificial intelligence

Title:

Attendance System

Group 7

Batch:

BS CS 02 B

Submitted to :

Maam Reeda

1. Overview

This project involves building an attendance system that utilises facial recognition to mark the presence and time-in of students. It covers areas such as facial detection, alignment, and recognition, along with the development of a web application to cater to various use cases of the system such as the registration of new students, addition of photos to the training dataset, viewing attendance reports, etc. This project intends to serve as an efficient substitute for traditional manual attendance systems. It can be used in corporate offices, schools, and organisations where security is essential.

2. Purpose

The purpose of this document is to specify the software requirements of the Attendance Management System Using Face Recognition. It is intended to be a complete specification of what functionality the Attendance Management System provides. Furthermore, this project aims to automate the traditional attendance system where the attendance is marked manually. It also enables an organisation to maintain its records like in-time and attendance digitally. Digitalization of the system would also help in better visualisation of the data using graphs to display the number of students present today and total work hours of each student. Its added features serve as an efficient upgrade and replacement over the traditional attendance system.

3. Scope

Facial recognition is becoming more prominent in our society. It has made major progress in the field of security. It is a very effective tool that can help law enforcers to recognize criminals, and software companies are leveraging the technology to help users access the technology. This technology can be further developed to be used in other avenues such as ATMs, accessing confidential files, or other sensitive materials. This project serves as a foundation for future projects based on facial detection and recognition. This project also covers web development and database management with a user-friendly UI. Using this system, any corporate office, school, or organisation can replace their traditional way of maintaining attendance of the students and can also generate their availability (presence) report throughout the month.

4. Introduction:

4.1 Programming Languages and Libraries/Frameworks used

- Python is a popular choice for this application due to its extensive libraries for scientific computing and machine learning.
- Django for web development.
- SQLITE Database.
- JavaScript
- Bootstrap
- OpenCV library provides functionalities for image processing and face detection.
- Scikit-learn library offers tools for machine learning model implementation and evaluation (including SVM).
- Dlib
- Open-Source Face Recognition Library

4.2 Machine Learning Implementation

The system trains the SVM model using facial encodings extracted from students' images. During attendance marking, the system extracts facial encodings from the user's face and feeds them to the trained model. The model predicts the user's identity based on the learned patterns in the encodings.

5. Functionality

We have two types of users of the system:

1. Student
2. Admin

5.1 Following functionalities can be performed by the admin:

- Login
- Register new students to the system
- Add student photos to the training data set
- Train the model
- View attendance report of all students. Attendance can be filtered by date or student.

5.2 Following functionalities can be performed by the student:

- Login
- Mark his/her time-in by scanning their face
- View attendance report of self

6. Implementation Details

The features of the system are mainly divided into three modules:

6.1 Registration and Login Module

This module mainly deals with the functionalities related to the registration of any new student to the organisation, log into the system, and managing student profile details. Using features provided by this module, admin can register new students to the system, and both admin and student can log into the system using their credentials.

6.2 Manage Attendance Details

This module mainly deals with the features related to the student's attendance. Using this, the student can mark their presence and time-in in the system. The admin can see the availability report of each student, and the student can see his/her attendance report along with some possible filters such as filter by student and filter by date.

6.3 Manage student Details

This module mainly deals with the features related to the student's profile. Using this, the admin can add a photo of the newly registered student during registration. The admin can also command the system explicitly to train the model, and the system will make necessary calculations and generate some data which will be used internally to identify each student uniquely.

6.4 Functionalities implemented successfully:

- Registration
- Login / Logout
- Manage User Profile
- Update user profile
- View My Attendance
- View Attendance by Date
- View Attendance by student
- Manage Attendance
- Add photos
- Add new student
- Train the system
- View Attendance record by date
- View the number of students present today

- View the total number of students

7. Detailed Implementation

7.1 Face Detection

We used OpenCV's Haar Cascade classifier to detect faces in real-time video streams. This method is effective for detecting faces in various lighting conditions and angles.

```
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
```

7.2 Face Alignment

Dlib's face alignment functionality ensures that detected faces are properly aligned before processing, improving the accuracy of facial recognition.

```
face_aligner = FaceAligner(predictor, desiredFaceWidth=256) face_aligned = face_aligner.align(frame, gray_frame, rect)
```

7.3 Face Recognition

For facial recognition, we used the face_recognition library, which is based on deep learning models to encode faces into 128-dimensional embeddings.

```
face_encodings = face_recognition.face_encodings(image)[0]
```

7.4 Support Vector Machine (SVM)

We used SVM from scikit-learn to classify these face embeddings. SVM is effective for small to medium-sized datasets and provides high accuracy.

```
svc = SVC(kernel='linear', probability=True) svc.fit(X_train, y_train)
```

8. Data Handling

We created datasets by capturing images from video streams, ensuring enough samples per individual for training the SVM model. This process involved capturing, aligning, and storing images in a structured directory format.

9. Model Training and Evaluation

9.1 Model Training

We trained the SVM model using the facial embeddings generated from the captured images. The dataset was split into training and testing sets to evaluate performance.

svc.fit(X_train, y_train)

The preprocessed dataset containing labelled facial encodings from students' images is split into training and testing sets. The SVM model is trained on the training data, learning to distinguish between different students' faces.

9.2 Model Evaluation

We evaluated the model using standard metrics such as accuracy, precision, recall, and F1-score.

from sklearn.metrics import classification_report y_pred = svc.predict(X_test) print(classification_report(y_test, y_pred))

9.2.1 Evaluation Metrics:

- Accuracy: Percentage of correctly predicted identities compared to the total number of predictions.
- Precision: Ratio of true positives (correctly identified students) to all positive predictions.
- Recall: Ratio of true positives to all actual positive cases (students present).

Our Projects:

```
# Calculate evaluation metrics
accuracy = accuracy_score(y_encoded, predictions)
precision = precision_score(y_encoded, predictions, average='weighted', zero_division=1)
recall = recall_score(y_encoded, predictions, average='weighted')
f1 = f1_score(y_encoded, predictions, average='weighted')
```

9.3 Hyperparameter Tuning

Various hyperparameters were tuned to optimise the model's performance, including the choice of kernel and regularisation parameters.

```
svc = SVC(kernel='rbf', C=1, gamma='auto')
```

9.4 Future Enhancement:

In the current implementation, the model undergoes repetitive training on the entire dataset whenever new data is added. This approach, while functional, is inefficient, especially as the dataset grows larger. A more efficient strategy would involve training the model only on the newly added data, updating its knowledge incrementally.

9.5 Comparing Models:

We can experiment with different machine learning models like K-Nearest Neighbors (KNN) or deep learning models like Convolutional Neural Networks (CNNs).

10. Application or Experimentation

10.1 Real-World Application

The developed model was integrated into a Django-based web application for real-time attendance monitoring. Students can mark their attendance by scanning their faces, and the system logs the time and presence accurately.

10.2 Interface Implementation

We implemented user-friendly interfaces using Django forms to facilitate data entry and attendance marking.

10.3 Data Visualization

We visualised attendance data using matplotlib and seaborn to generate insights such as daily attendance trends and individual attendance records.

```
sns.barplot(data=df, x='date', y='hours')
plt.savefig('./recognition/static/recognition/img/attendance_graphs/hours_vs_date/1.png')
```

11. Experimental Results

We conducted several experiments to validate the system's effectiveness under different conditions (e.g., varying lighting and camera angles). The results demonstrated high robustness and accuracy, making it suitable for real-world deployment.

11.1 Expected Outcomes:

- Improved accuracy and efficiency in attendance tracking compared to manual methods.
- Reduced time spent on attendance management for both students and administrators.
- Potential for integrating with other systems for better data management.

11.2 Additional Considerations:

- Data Security: Ensure secure storage and transmission of students' facial data to comply with privacy regulations.
- Lighting Conditions: Train the model under various lighting conditions to improve robustness.
- Scalability: The system should be scalable to accommodate a growing number of students.
- User Acceptance: Address students' concerns regarding privacy and potential for bias in facial recognition technology.

12. Conclusion:

In conclusion, the attendance system utilising facial recognition technology represents a significant advancement in automating and optimising attendance management processes. Its successful implementation, coupled with robust performance and user-friendly features, underscores its potential as a valuable tool for enhancing efficiency, security, and data management in various organisational settings.