

Article

The Application of Adaptive Tolerance and Serialized Facial Feature Extraction to Automatic Attendance Systems

Chun-Ling Lin *  and Yi-Huai Huang

Department of Electrical Engineering, Ming Chi University of Technology, No. 84, Taishan Dist., New Taipei City 243, Taiwan; m07128014@mail2.mcut.edu.tw

* Correspondence: ginnylin@mail.mcut.edu.tw; Tel.: +886-2-2908-9899 (ext. 4819)

Abstract: The aim of this study was to develop a real-time automatic attendance system (AAS) based on Internet of Things (IoT) technology and facial recognition. A Raspberry Pi camera built into a Raspberry Pi 3B is used to transfer facial images to a cloud server. Face detection and recognition libraries are implemented on this cloud server, which thus can handle all the processes involved with the automatic recording of student attendance. In addition, this study proposes the application of data serialization processing and adaptive tolerance vis-à-vis Euclidean distance. The facial features encountered are processed using data serialization before they are saved in the SQLite database; such serialized data can easily be written and then read back from the database. When examining the differences between the facial features already stored in the SQLite databases and any new facial features, the proposed adaptive tolerance system can improve the performance of the facial recognition method applying Euclidean distance. The results of this study show that the proposed AAS can recognize multiple faces and so record attendance automatically. The AAS proposed in this study can assist in the detection of students who attempt to skip classes without the knowledge of their teachers. The problem of students being unintentionally marked present, though absent, and the problem of proxies is also resolved.

Keywords: automatic attendance system (AAS); Internet of Things (IoT); face detection; face recognition; Euclidean distance; SQLite



Citation: Lin, C.-L.; Huang, Y.-H. The Application of Adaptive Tolerance and Serialized Facial Feature Extraction to Automatic Attendance Systems. *Electronics* **2022**, *11*, 2278. <https://doi.org/10.3390/electronics11142278>

Academic Editor: Nurul I. Sarkar

Received: 23 May 2022

Accepted: 20 July 2022

Published: 21 July 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The regular attendance of students is a prerequisite for good academic performance. Student attendance is one of the most important issues for all educational institutions. There are two common types of student attendance systems in use: manual attendance systems (MSA) and automatic attendance systems (AAS) [1]. The term MSA covers the traditional systems that require teachers to fill in attendance sheets manually. Where classes are larger, this can be difficult to administer because teachers often face enormous pressures, and it takes a lot of time to collect the necessary details about each student's name and record these without any errors. However, of late, information technology [2–5] has been widely used to provide convenience, speed up the task, and also streamline it. For the purposes of AAS, various technologies, such as facial recognition, iris detection, and radio frequency identification (RFID), have been used [6]. RFID systems utilize sensors in order to read data. The usefulness of RFID for this application lies in the fact that it can facilitate lecturers and, indeed, students with monitoring class attendance, via an AAS [7]. On the other hand, there are also some disadvantages, such as the fact that RFID is not as secure, again for this application, as biometric methods; the system is prone to manipulation [8]. That is, another person can use same RFID to do the roll call. In recent years, systems have been provided with powerful tools supporting artificial intelligence (AI) operations, including machine learning operations based on high bandwidth CPUs, GPUs, and specific AI accelerators. One such operation, facial recognition [9], has attracted a great deal of research attention and

has been subject to sustained development over the past 30 years. It has great potential for use across numerous government and commercial applications [10–12]. Security cameras with facial recognition capabilities are presently common in airports, offices, universities, ATM installations, banks, and indeed, in any location with a security system [13–15]. Parmar and Mehta described the most common methods of face recognition, such as the holistic matching method, the feature extraction method, and hybrid methods; and proposed a number of face recognition applications, such as face identification, access control, and security and identity verification [13]. Khandelwal et al. suggested various face recognition processing methods and adopted a Convolutional Neural Network (CNN) to develop a face recognition algorithm [14]. Norouzi explored face recognition based on deep neural networks and described various facial expression recognition applications [15], such as student attendance systems and building security systems.

Among all the techniques applied to AAS, facial recognition is considered to be the most efficient [16]. Although facial recognition based on deep learning techniques [17–19] has high accuracy, the model is complex and the recognition speed is slow, even when working on just a single image. In order to realize real-time facial recognition of students, two particular computer vision libraries—the open-source computer vision libraries (OpenCV) [20,21] and Dlib [22–24]—have been applied in many previous studies. Kar et al. adopted the OpenCV and Light Tool Kit (FLTK) to develop an AAS system and suggested that OpenCV can provide a simple-to-use computer vision infrastructure that can help people build fairly sophisticated vision applications quickly. The OpenCV library contains over 500 functions that span many areas of vision [20]. Joseph et al. adopted face detection, 128-dimension face encoding extraction, and support-vector machine (SVM) training to develop a system [21]. Boyko et al., explored the features and analyzed the pros and cons of OpenCV and Dlib [22]. Xu et al., adopted Dlib to detect faces, and then adopted CNN and a deep residual network (ResNet) for real-time face recognition [23]. Ambre et al. explored a real-time face recognition system using easily attainable components and libraries, such as Raspberry Pi; Dlib, a Face Recognition library; and OpenCV [24].

This has been done on the basis that the utilization of libraries allows easy application and reimplementations. The OpenCV library is a cross-platform computer library which can provide an infrastructure for computer vision applications and allow the use of machine perception easily within commercial products. However, OpenCV [25] suffers from the problems of missed detections, false detections, and in effect poor recognition overall. On the other hand, Dlib is a cutting-edge toolbox which includes various different AI tools and which allows the creation of complex software programs for solving real-world problems [26]. Dlib can deal with bad and inconsistent lighting and various facial positions (such as tilted or rotated). It also achieves high performance within real-world implementations both in terms of speed and accuracy [27]; all this makes it suitable as a basis from which to develop an AAS. The challenges of AAS include how to make the attendance registration and management systems efficient, time-saving, simple, and easy. Furthermore, an AAS needs to recognize, accurately, multiple faces, faces in inconsistent lighting conditions, and faces presented via various positions. Pandey et al. combined the on-demand resource availability of cloud computing, which can store and retrieve the captured video anytime. Additionally, its surveillance mechanism involves the Viola–Jones algorithm for face detection by analyzing captured data [28]. Shanthi and Svalakshmi developed a surveillance method by using a face recognition based unmanned aerial vehicle (UAV) [29]. The method of UAV was implemented on Raspberry Pi module with Python libraries which included OpenCV, Dlib, Face_recognition, and Numpy. Gupta and Singh used in-memory computation to develop a real-time face recognition system. Even while the number of faces is increasing, it can still keep the frame rate steady during the entire process [30]. These recent studies developed an AAS system based on Raspberry Pi, a cloud computing environment, and face detection algorithms. They can make the systems simply, easily, and more efficiently.

The aim of this study was to develop a real-time AAS system based on Internet of Things (IoT) technology and facial recognition, which can handle the registration of students. The main contributions of this study are threefold:

1. Building a cloud server to develop face detection and recognition algorithms. A Raspberry Pi Camera system built into a Raspberry Pi 3B was used to capture and transfer images to a cloud server with a high-speed GPU. The development of our face detection and recognition algorithms was based on this cloud server, which handles the processes involved in registering student attendance, namely, comparing faces using extracted features stored in a database.
2. Creating an SQLite database to save students' facial features based on data serialization, which we propose as the strategy for this process. In order to create the SQLite database, the students' facial features will be put through data serialization before they are saved using SQLite.
3. Improving the performance of face recognition using adaptive tolerance with respect to Euclidean distance. In comparing the facial features found in the database with the new facial features encountered in the input, the former needs to be subjected to data deserialization. The proposed adaptive tolerance method can improve the performance of the facial recognition processes using Euclidean distance.

With the implementation of our proposed AAS, students cannot skip classes without the knowledge of their teachers.

This study is organized as follows. In Section 2, the proposed AAS schematic and methods are described. Section 3 shows the results of this study. Section 4 discusses the proposed AAS system and indicates the advantages of the proposed methods. Section 5 provides concluding remarks.

2. Methods

Figure 1 shows the schematic of our proposed AAS. The design of our AAS system includes four main components: a Raspberry Pi camera built into a Raspberry Pi 3B; a Django background processing system, the OpenCV image processing library, and the Dlib face detection and face recognition module. For each student, the proposed AAS system must save a headshot, and a complete set of facial features derived from this, onto the cloud server. The dimensions of the headshot are reduced to a standard size and then converted to a grayscale image, using OpenCV. Once this is done, the facial landmarks are identified using the Dlib library. Each set of facial landmarks derived from the students' faces is saved in SQLite [31] and associated with an appropriate students' names. A Raspberry Pi camera built into a Raspberry Pi 3B was set up in the classroom in order to capture the images and transfer them to a cloud server through the RESTful application programming interface (API). When the cloud server receives an image, the facial landmarks from this can be extracted using the Dlib library. The newly extracted features can then be compared with those already stored in the SQLite database in order to confirm (or not) the identity of the student. If this identity verification is successful, the student is recognized and so the student's name will show on a computer screen in the classroom as having been marked as in attendance.

2.1. The Cloud Server

The Django cloud server has an automatic background task management interface and can provide technical support systems, including a user authentication mechanism; it (the server) is used to receive the facial images, store these in the database, and process them [32]. This study adopted the Django system as the basis for implementing Raspberry Pi as a web server [33], applying the RESTful API [34]. The latter is an architectural style API that uses HTTP requests to access and use data. Into the Django system, we imported the OpenCV and Dlib libraries. These support the engine in processing the images and so detecting and recognizing the faces of the students.

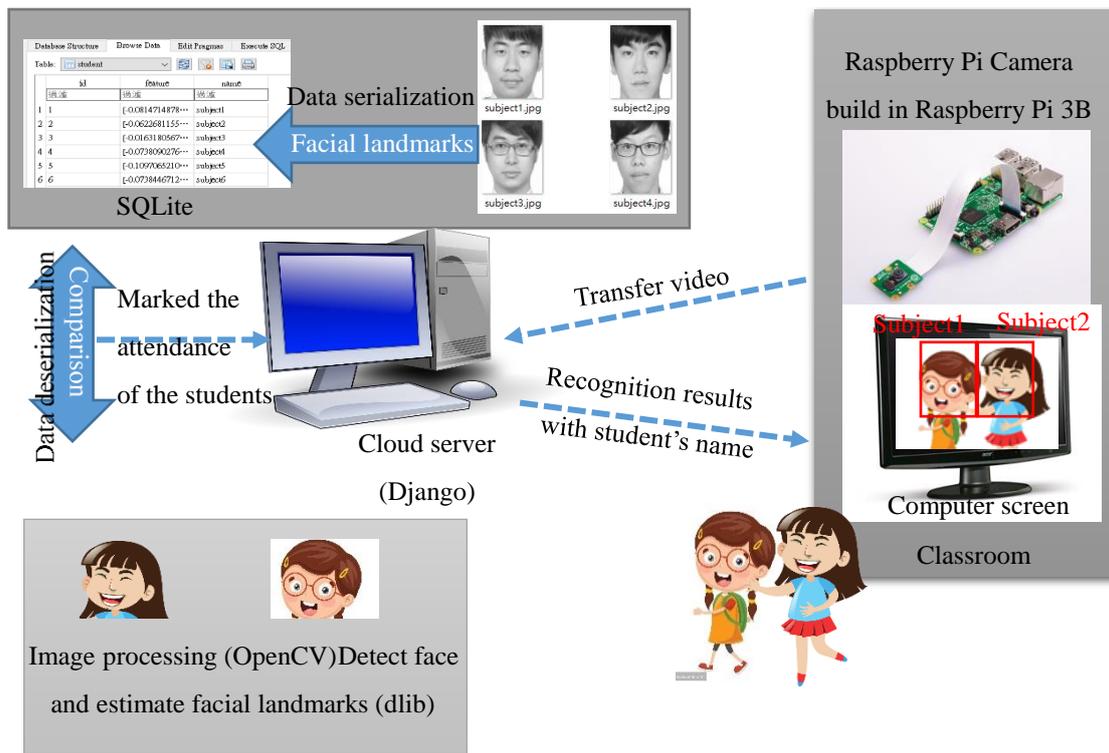


Figure 1. AAS schematic proposed in this study. There are four main components: a Raspberry Pi camera built into a Raspberry Pi 3B, a Django background processing system, the OpenCV image processing library, and the Dlib face detection and face recognition module. In the SQLite, the meaning of 過濾 is filter.

2.2. SQLite Databases

The flowchart in Figure 2 shows how the student database is created via SQLite. SQLite is a popular database management system [35]. In this case, the SQLite database will contain information regarding the students to be detected and recognized. Each student, on enrolment, must provide a headshot which presents a complete set of facial features, as shown in Figure 3a. The convolutional neural network’s (CNN) features, used within the context of the Maximum-Margin Object Detector (MMOD) (MMOD-CNN) face detector in the Dlib library (Dlib CNN), are adopted as the basis by which the faces and their positions within the object are detected, identified, and recognized [36–38]. The face within the object is extracted from the original headshot and converted to grayscale using OpenCV (Figure 3b). For the Dlib facial recognition network, the image being processed must be transformed into an output feature vector with 128 numerical facial features. These are used to characterize each face [39], as shown in Figure 4.

In order to store, in SQLite, the output feature vectors, each representing the 128 numerical features which in turn represent a particular face, the feature vector needs to be subjected to data serialization. This technique converts data objects represented as complex data structures into a byte stream for storage, transfer, and distribution purposes on physical devices, using JavaScript Object Notation (JSON library) [40]. The name of the subject and the 128 numerical facial features, after JSON serialization, will be stored in the SQLite database, as shown in Figure 5.

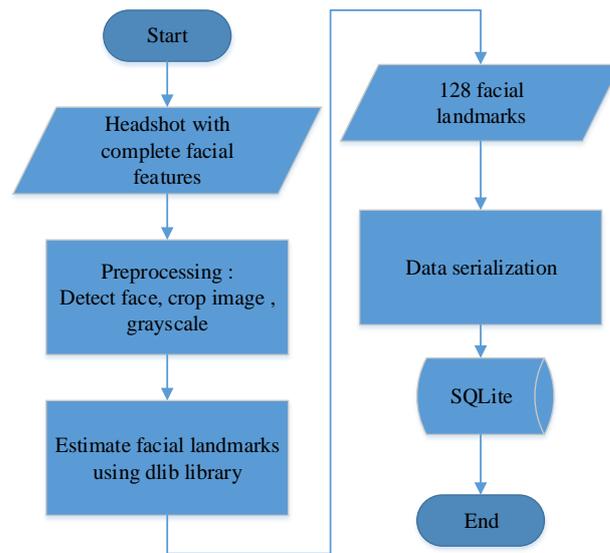


Figure 2. Flowchart showing the process of creating the student database in SQLite.

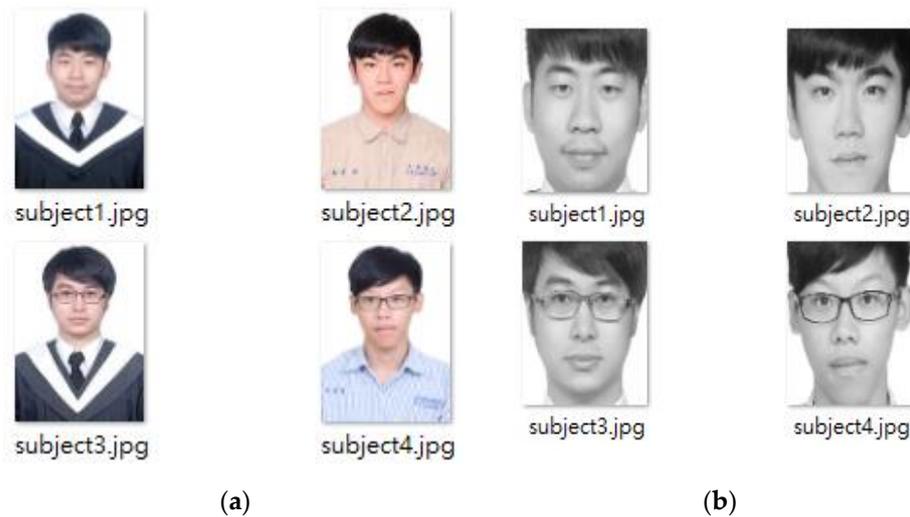


Figure 3. The headshot with complete facial feature. (a) The headshot which presents a complete set of facial features. (b) The face within the object is extracted from the original headshot and converted to grayscale using OpenCV.

[-0.073808059	0.023394182	0.016828062	-0.015616010
-0.128747314	0.015926898	-0.020160629	-0.126099691
0.169534236	-0.126991540	0.257808805	-0.032651115
-0.196335346	-0.118733460	0.005301114	0.178630547
-0.150136396	-0.138667986	-0.002825755	0.019164924
0.088197798	-0.064872883	0.044611085	0.069038384
-0.088866383	-0.374791265	-0.128574252	-0.108442500
0.036157772	-0.107900344	-0.073528722	0.055888694
-0.164956629	0.015193242	-0.016166175	0.093032621
-0.021139123	-0.104937270	0.188696166	-0.056512587
-0.290270299	0.029002756	0.003925093	0.156920642
0.190407872	0.041880634	0.008822564	-0.089200839
0.080764718	-0.155750453	0.052590743	0.143794313
0.052435525	0.012301343	-0.054230191	-0.161440661
0.005815323	0.062656872	-0.110088676	-0.014144851
0.063672394	-0.100441523	-0.011834437	-0.020623902
0.231908605	0.131724730	-0.121522091	-0.160300255
0.090293169	-0.198756690	-0.064311959	0.046206439
-0.222958550	-0.213464469	-0.295057684	0.029147003
0.410964072	0.086139560	-0.177729607	0.034496654
-0.030929461	0.050941691	0.142053038	0.203301698
-0.002924228	0.007855131	-0.128640414	0.010752909
0.237488270	-0.073403411	-0.045293916	0.243974403
-0.037613012	0.133805454	0.002736636	0.023758892
0.047262665	0.028300472	-0.080618791	0.017525349
0.067027181	-0.039010250	-0.027649137	0.123970747
-0.173203334	0.068781003	-0.015800448	0.063422494
0.053454261	-0.008215142	-0.127380639	-0.061022285
0.132721931	-0.218805462	0.229971990	0.215902939
0.087361805	0.104119539	0.149996951	0.091853194
0.013973497	-0.064014137	-0.244877905	0.050616484
0.117065243	-0.040216848	0.043420331	-0.000358786]

Figure 4. Output feature vector with 128 numerical facial features by Dlib facial recognition network which was sipped from the python results.

	id	feature	name
	過濾	過濾	過濾
1	1	[-0.0814714878...	subject1
2	2	[-0.0622681155...	subject2
3	3	[-0.0163180567...	subject3
4	4	[-0.0738090276...	subject4
5	5	[-0.1097065210...	subject5
6	6	[-0.0738446712...	subject6

Figure 5. SQLite databases after JSON serialization which was sipped from the SQLite software results The meaning of 過濾 is filter.

2.3. Face Recognition

Figure 6 shows the flowchart of the face recognition process proposed and described as follows:

1. The real-time continuous face recognition AAS, which captures the attendance and duration of attendance of students from the Raspberry Pi camera set-up, requires that Raspberry Pi Model B.
2. The raspberry Pi model extracts an image from every six frames of the live video (of the classroom) and uploads them to the cloud server.
3. The MMOD-CNN process has been adopted as the process by which the faces—and their positions within the object—are detected. The object is a list of rectangular objects. If the MMOD-CNN cannot obtain rectangle objects, we go back to Step 2.
4. OpenCV crops the image of each face with the object and converts it to grayscale form.
5. The estimated 128 facial landmarks are obtained using the Dlib facial recognition network.
6. For facial similarity calculation, the estimated facial landmarks are compared with the facial landmarks currently stored in the SQLite database. This comparison is performed after the JSON deserialization process has been applied to the existing facial features stored in the SQLite database, using Euclidean distance [41] as follows:

$$\sum_{i=1}^{128} \sqrt{(x_{1i} - x_{2i})^2} \quad (1)$$

where x_1 is one of the facial landmarks which currently exists in the SQLite database, and x_2 is a just-estimated facial landmark derived from a new image of a student.

7. If the value of the Euclidean distance is smaller than the tolerance, this means that the student whose image is being processed has been identified.
8. The name of the student can be shown on the computer screen, and the student is marked as present via the cloud server.

2.3.1. Maximum-Margin Object Detector Convolutional Neural Network (MMOD-CNN)

Dlib has two types of face detection tool (<https://pyimagesearch.com/2021/04/19/face-detection-with-Dlib-hog-and-cnn/> (accessed data:15 June 2022)): one is based on a classic histogram of oriented gradient (HOG) features, in conjunction with the linear classifier SVM, pyramid images, and a sliding window detection scheme; the other is based on the Maximum-Margin Object Detector convolutional neural network (MMOD-CNN), a face detector that uses deep learning face detection. The MMOD-CNN can be adopted using `cnn_face_detection.py`; the function returns a list of rectangle objects which extract the face within the object.

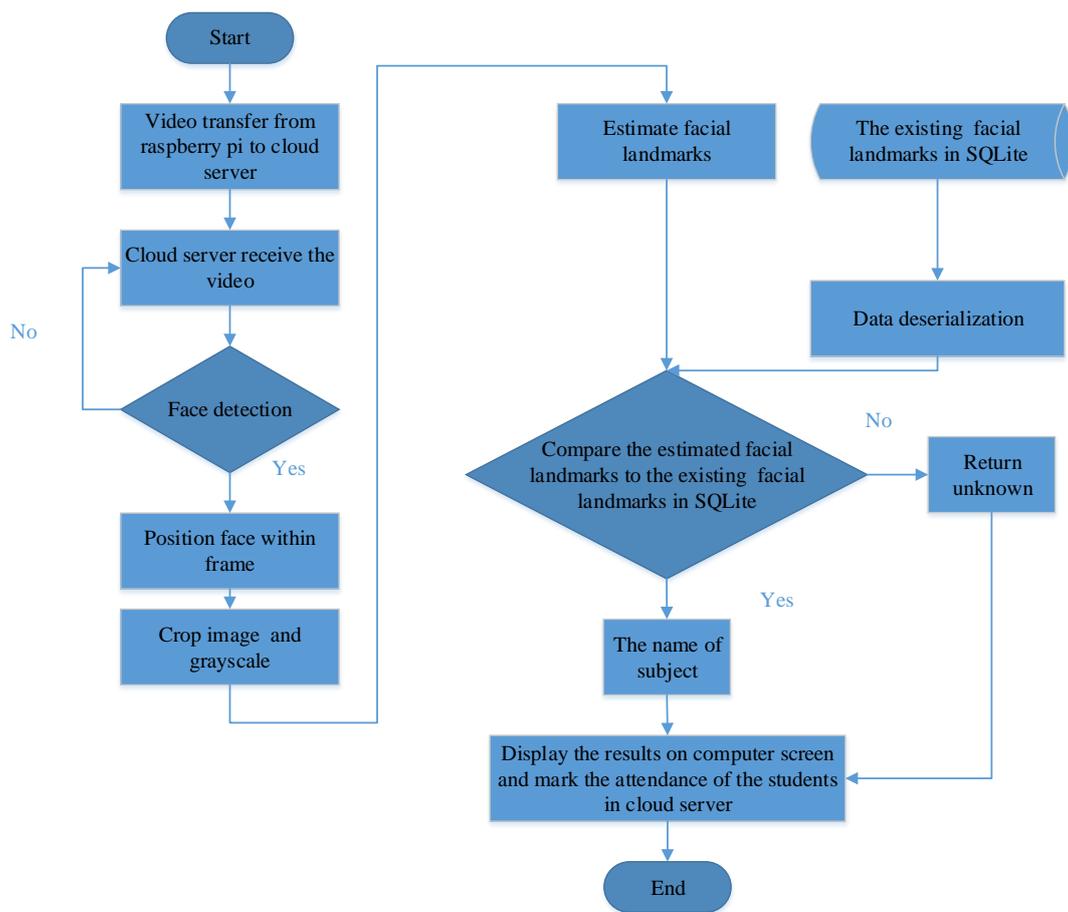


Figure 6. Flowchart of face recognition process.

2.3.2. Adaptive Tolerance

The definition of the tolerance, in terms of Euclidean distance, is a challenging problem in terms of face detection. In order to find a suitable tolerance, we set up a camera at a classroom door and asked six subjects to stand two meters from that door with their faces uncovered. With regard to these subjects, the smallest values of Euclidean distance between estimated facial landmarks and existing (in the database) facial landmarks can be observed in Table 1. Thus, in this case, the tolerance in terms of Euclidean distance can be defined as 0.44.

Table 1. Euclidean distances from six subjects.

Subject	Euclidean Distance
Subject1	0.458
Subject2	0.433
Subject3	0.427
Subject4	0.417
Subject5	0.44
Subject6	0.451
Average	0.438

However, in general, tolerance has to be defined as a fixed number, whereas the performance of facial recognition will be variable because, for instance, the distance between the student and the camera will affect this performance, as shown in Figure 7. When a

student is a significant distance from the camera, the face she or he presents will be relatively small. Although the MMOD-CNN process can, nevertheless, detect the face and find its position within the object, the number of pixels representing the face becomes very limited, which means that the detailed facial landmarks cannot be extracted via the Dlib facial recognition network. This causes the Euclidean distances involved to increase overall. As a result, students cannot necessarily be identified, as shown in Figure 7b.

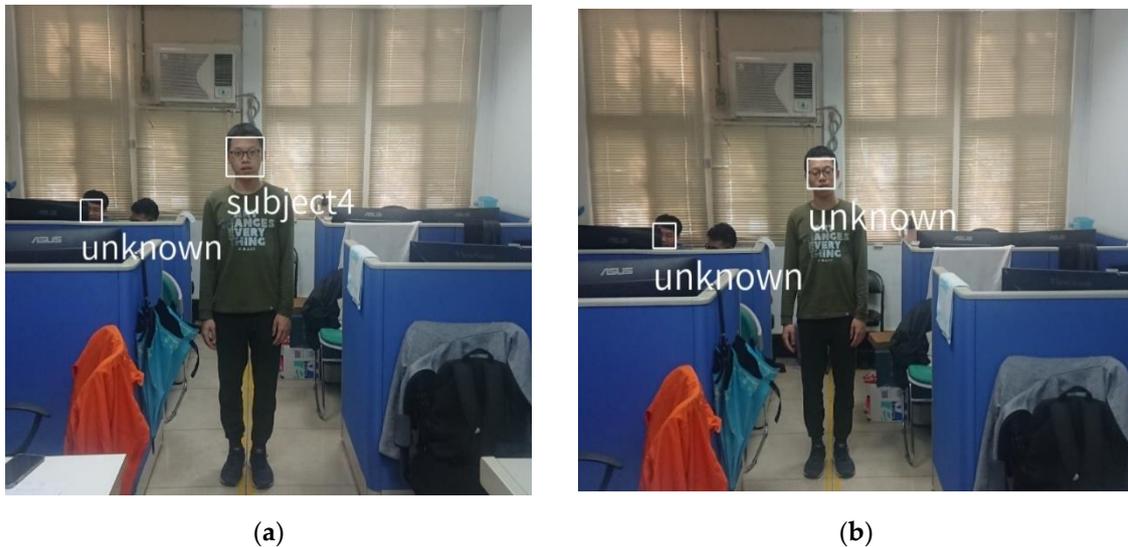


Figure 7. The performance of facial recognition, keeping the tolerances the same throughout (0.44). (a) The middle subject in the image is 0.68 cm and can be recognized. (b) The middle subject in the image is 0.47 cm and cannot be recognized.

Table 2 shows the values of the Euclidean distances between some estimated facial landmarks and some existing facial landmarks (existing, as in being present within the database), when the widths of the faces involved are different; see Figure 7. In Figure 7b, the width of the face of the middle subject in the image is 0.47 cm, and that of the left subject in the image is 0.39 cm. The smallest value of Euclidean distance, as between the middle subject and the existing dataset, is that between this subject and subject 4. If the tolerance with respect to the Euclidean distance is equal to 0.44, the facial recognition process will yield the value “unknown.” This study proposes a method whereby an adaptive tolerance can be defined, based on the width of the face presented. Table 2 shows that the width of the face represented by it is equal to 0.68, and this can be recognized accurately using the tolerance with respect to Euclidean distance of 0.44. Thus, the standard width of face is defined as 0.7 cm, and the standard tolerance with respect to Euclidean distance as 0.44. When the width of the face is larger or smaller than 0.7, we say that the definition of the tolerance will vary as follows:

$$\text{Adaptive Tolernace} = 0.44 \times \left(\frac{0.7}{10} + \text{WF} \right) / \text{WF} \quad (2)$$

where WF is the width of the face presented.

Table 2. The values of the Euclidean distance with respect to two subjects when the sizes of faces are different, as in Figure 7.

Width of Face = 0.39 cm (Left Subject)	Euclidean Distance in Figure 7a
Subject 1	0.5833
Subject 2	0.6239
Subject 3	0.5238
Subject 4	0.6260
Subject 5	0.6388
Subject 6	0.5625
Width of face = 0.68 cm (middle subject)	Euclidean distance in Figure 7a
Subject 1	0.5203
Subject 2	0.5745
Subject 3	0.3967
Subject 4	0.3535
Subject 5	0.5153
Subject 6	0.4891
Width of face = 0.39 cm (left subject)	Euclidean distance in Figure 7b
Subject 1	0.5937
Subject 2	0.5409
Subject 3	0.5835
Subject 4	0.6728
Subject 5	0.6107
Subject 6	0.6113
Width of face = 0.47 cm (middle subject)	Euclidean distance in Figure 7b
Subject 1	0.5723
Subject 2	0.6466
Subject 3	0.5223
Subject 4	0.4681
Subject 5	0.6230
Subject 6	0.6096

3. Results

In this study, we developed a real-time AAS based on Internet of Things (IoT) technology and facial recognition. A Raspberry Pi 3B setup including a Raspberry Pi camera captures and then transfers six frames from each local video of a face to the cloud server. The development of the facial detection and recognition algorithms is based on the facilities available on the cloud server. In addition, in this study, we propose two strategies in particular: data serialization for storing facial features in SQLite easily and adaptive tolerance (with respect to the width of the facial image) for improving the performance of facial recognition using Euclidean distance.

3.1. The Performance of Adaptive Tolerance

This study proposes adaptive tolerance for improving the performance of face recognition using Euclidean distance. The tolerance for face recognition is not fixed, instead of Equation (2). In order to compare the performance of face recognition using fixed tolerance (Figure 7b) with adaptive tolerance (Figure 8), two different tolerances were

set in the face recognition using Euclidean distance. The student stood at the same location (Figures 7b and 8). If adaptive tolerance was adopted to set the tolerance in the face recognition using Euclidean distance, the student could be recognized.

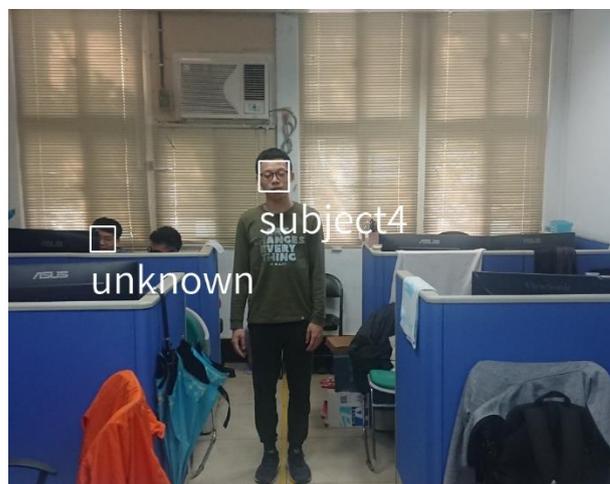


Figure 8. Adaptive tolerance for Euclidean distance in the facial recognition task.

3.2. The Performance of Face Detection Using the Dlib CNN and Dlib HOG Libraries

The Dlib library is arguably one of the most widely utilized packages for the purposes of facial recognition. This study compares the performance of face detection achieved by Dlib CNN with that achieved by Dlib HOG [42] in Figure 9.

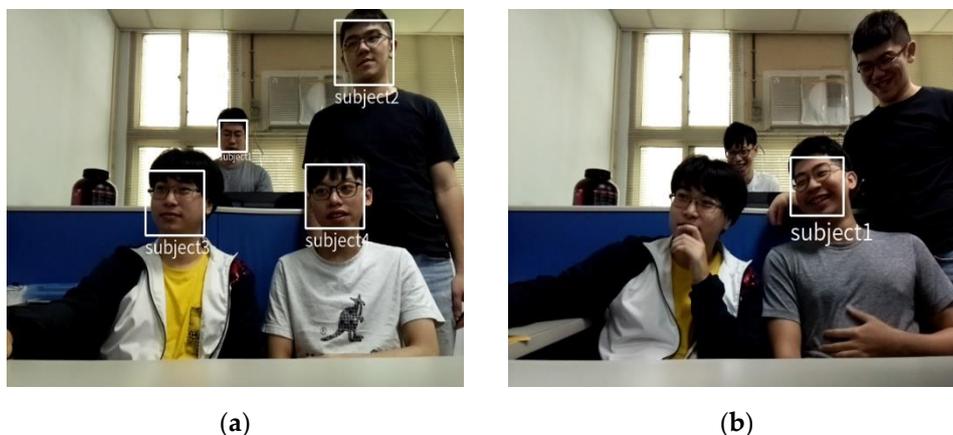


Figure 9. The performance of face detection using Dlib CNN (a) and Dlib HOG (b).

1. Dlib CNN: A Maximum-Margin Object Detector (MMOD) CNN face detector that is highly accurate, very robust, and capable of detecting faces from varying viewing angles, in various lighting conditions, and with occlusions.
2. Dlib HOG: A HOG + Linear SVM face detector that is accurate and computationally efficient.

Figure 9 shows that the Dlib CNN can detect the four faces presented within the object, but that Dlib HOG can only detect one face. Thus, Dlib CNN can be said to have better performance, in terms of detecting faces, than Dlib HOG. Hence, this study adopted the Dlib CNN library for the purpose of detecting faces and the positions of faces within frames.

3.3. Mask and Light Tests for Face Detection and Facial Recognition

The nature and competence of the extraction of facial features affect the performance of face detection and face recognition. The important facial features include: the location

of the eyes, and the relationship between the facial features and the overall contour of the faces. The ears and hairstyles are not determinants in terms of facial recognition because hairstyles are changeable, and so are often controlled or excluded, and the ears are almost always covered, which makes them less meaningful. In this study, we implemented our proposed facial detection and recognition methods, working under differing masking and lighting conditions, as shown in Figure 10a–i. Figure 10a–d indicates which conditions cause face detection and facial recognition failures, and Figure 10e–i indicates under which conditions successful face detection and recognition can be achieved.

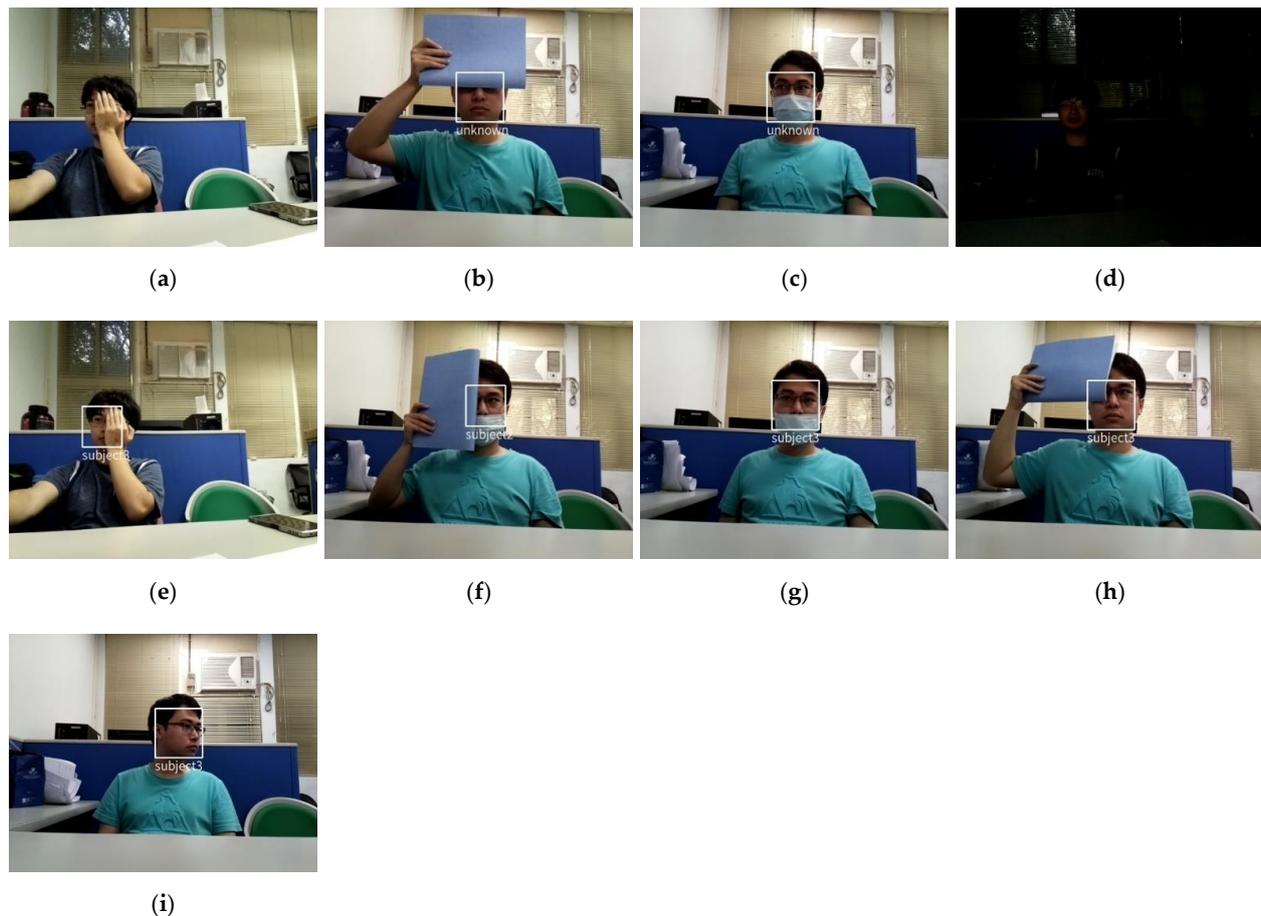


Figure 10. The performance of the proposed face detection and facial recognition methods under different masking and lighting conditions. (a) More than half of the facial features are covered. (b) Two eyes covered by a book. (c) Mouth and nose cover by a mask. (d) Environmental brightness is low. (e) Half of the facial features are covered. (f) Mouth and eye cover by a mask and book. (g) Mouth covered by mask. (h) One eye covered by a book. (i) Side profile.

According to the results shown in Figure 10, even when only half the facial features are visible, the system works; however, when the environmental brightness is low or more than half of the facial features are covered, face detection does not work. If most of the face is covered, very little information can be extracted by the Dlib library, and so the subsequent processing cannot work. Additionally, most face detection methods are significantly affected by changes in environmental illumination levels [43]. This proposed system will be set up in the classroom in order to register the attendance of students, and so it is unlikely that the environmental illumination levels will always be low enough to affect the facial recognition system significantly.

3.4. The AAS System

In order to test the performance of our AAS system (Figure 1), we set it up in the laboratory. The Raspberry Pi camera built into the Raspberry Pi 3B was thus able to capture relevant images, as shown in Figure 11. The Raspberry Pi transferred the images to the cloud server using an appropriate API. The results of the face detection and facial recognition processes are shown in Figure 9a. The face detection and facial recognition system implemented in the cloud server using the OpenCV and Dlib libraries successfully supported the automatic registration of the attendance of the students in the cloud server (Figure 12); the results were returned to the Raspberry Pi and shown on the computer screen.



Figure 11. Example of an image that can be captured by the Raspberry Pi camera.

Database Structure		Browse Data		Edit Pragmas		Execute SQL	
Table: attendance							
	name		time				
	過濾		過濾				
1	subject3		2020-7-22 14:2...				
2	subject1		2020-7-22 14:2...				
3	subject4		2020-7-22 14:2...				
4	subject2		2020-7-22 14:2...				

Figure 12. Roll call in SQLite which was sipped from the SQLite software results. The meaning of 過濾 is filter.

In addition, the AAS system was set up in the laboratory (Figure 11) and actually worked for one minute. The Raspberry Pi Model B sets 12 frames per second for the video stream, and extracts an image from every six frames of the live video. The AAS can work and record the attendance of the students every 0.5 s. Over the 120 occasions of four students' attendance the recording, the proposed system obtained 100% accuracy; i.e., four students could be detected and recognized in each image. When the AAS system with face detection using Dlib HOG and the adaptive tolerance for face recognition was involved, only one student could be recognized each time. When the AAS system with the face detection using Dlib CNN and with the fixed tolerance for face recognition was used, only three students could be recognized each time. The last student could be detected but not recognized. Thus, the results show that the proposed system is better than the AAS system at face detection using Dlib HOG, and the AAS system with face detection using Dlib CNN and a fixed tolerance for face recognition. Additionally, the performance of the proposed AAS system was compared with the previous study [44] which adopted Dlib HOG and Dlib CNN for face detection, deep residual networks (ResNets) for feature extraction, k-nearest neighbors (KNN) classifier for face recognition in the same video (Figure 11). The pervious study [44] obtained 99 % accuracy. Only a few images were not able to be used to detect and recognize all the four students.

4. Discussion

This study was a pilot study because we tested the performance of an AAS system in the laboratory rather than in the classroom. The study proposed a real-time AAS system schematic. Data serialization converted data objects (record images) present in complex data structures into a byte stream for storage, transfer, and distribution purposes on a physical device, which is a good tool for creating a dataset in SQLite. The results shown in Figure 8 reveal the adaptive tolerance for improving the performance of face recognition using Euclidean distance. In addition, Dlib CNN can be said to have a better performance in terms of detecting faces than Dlib HOG. Finally, this study tested a real-time AAS system in a laboratory setting, and the results shown in Section 3.4 indicate that the system was able to detect all of the students and could be registered in the cloud server automatically. In order to verify the performance of a real-time AAS system, we will apply to the Institutional Review Board (IRB) and test the AAS system in a classroom setting in our future work. In addition, this study adopted Euclidean distance with adaptive tolerance to face recognition. Future, the SVM and a more powerful classify method (CNN) will be adopted to compare its performance and efficacy to related face recognition methods.

5. Conclusions

In this study, we applied a combination of a number of common methods and libraries in order to develop an AAS system. In addition, we used our proposed data serialization and adaptive tolerance for Euclidean-distance methods to optimize the performance of this AAS. Data serialization was utilized in order to support the storing of facial features via SQLite. The calculation of adaptive (rather than fixed) tolerance was shown here to improve the performance of Euclidean-distance-based facial recognition. The results yielded by this study demonstrate that the proposed AAS can recognize multiple faces and so facilitate the automatic marking of attendance. Our proposed AAS would mean, in the classroom situation, that students would not be able to skip classes without the knowledge of their teachers.

Our development of this AI model and these techniques supports improvements in the design and application of facial recognition. However, the AAS implemented in this study also has some shortcomings which should be addressed—viz:

When the light levels in the classroom are low, this can mean that the proposed system manifests low accuracy in terms of face detection and recognition. In the future, we will explore methods by which the face detector system can be adjusted to work with low light levels.

Author Contributions: Conceptualization, C.-L.L. and Y.-H.H.; methodology, C.-L.L. and Y.-H.H.; software, Y.-H.H. validation, C.-L.L. and Y.-H.H.; formal analysis, Y.-H.H.; data curation, Y.-H.H.; writing—original draft preparation, C.-L.L. and Y.-H.H.; writing—review and editing, C.-L.L.; supervision, C.-L.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Shah, K.; Bhandare, D.; Bhirud, S. Face recognition-based automated attendance system. In *International Conference on Innovative Computing and Communications*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 945–952.
2. Liu, H.; Zheng, C.; Li, D.; Shen, X.; Lin, K.; Wang, J.; Zhang, Z.; Zhang, Z.; Xiong, N.N. EDMF: Efficient deep matrix factorization with review feature learning for industrial recommender system. *IEEE Trans. Ind. Inform.* **2021**, *18*, 4361–4371. [[CrossRef](#)]
3. Li, D.; Liu, H.; Zhang, Z.; Lin, K.; Fang, S.; Li, Z.; Xiong, N.N. CARM: Confidence-aware recommender model via review representation learning and historical rating behavior in the online platforms. *Neurocomputing* **2021**, *455*, 283–296. [[CrossRef](#)]
4. Liu, T.; Liu, H.; Li, Y.-F.; Chen, Z.; Zhang, Z.; Liu, S. Flexible FTIR spectral imaging enhancement for industrial robot infrared vision sensing. *IEEE Trans. Ind. Inform.* **2019**, *16*, 544–554. [[CrossRef](#)]
5. Liu, H.; Fang, S.; Zhang, Z.; Li, D.; Lin, K.; Wang, J. MFDNet: Collaborative poses perception and matrix Fisher distribution for head pose estimation. *IEEE Trans. Multimed.* **2021**, *24*, 2449–2460. [[CrossRef](#)]

6. Ali, N.S.; Alhilali, A.H.; Rjeib, H.D.; Alsharqi, H.; Al-Sadawi, B. Automated attendance management systems: Systematic literature review. *Int. J. Technol. Enhanc. Learn.* **2022**, *14*, 37–65. [[CrossRef](#)]
7. Ula, M.; Pratama, A.; Asbar, Y.; Fuadi, W.; Fajri, R.; Hardi, R. A New Model of The Student Attendance Monitoring System Using RFID Technology. *J. Phys. Conf. Ser.* **2021**, *1807*, 012026. [[CrossRef](#)]
8. Solanke, M.D.S. RFID technology in libraries. *Int. J. Res.* **2021**, *8*, 90–97.
9. Liu, T.; Wang, J.; Yang, B.; Wang, X. Facial expression recognition method with multi-label distribution learning for non-verbal behavior understanding in the classroom. *Infrared Phys. Technol.* **2021**, *112*, 103594. [[CrossRef](#)]
10. Zhang, Z.; Lai, C.; Liu, H.; Li, Y.-F. Infrared facial expression recognition via Gaussian-based label distribution learning in the dark illumination environment for human emotion detection. *Neurocomputing* **2020**, *409*, 341–350. [[CrossRef](#)]
11. Liu, H.; Nie, H.; Zhang, Z.; Li, Y.-F. Anisotropic angle distribution learning for head pose estimation and attention understanding in human-computer interaction. *Neurocomputing* **2021**, *433*, 310–322. [[CrossRef](#)]
12. Wu, H.; Liu, Y.; Liu, Y.; Liu, S. Fast facial smile detection using convolutional neural network in an intelligent working environment. *Infrared Phys. Technol.* **2020**, *104*, 103061. [[CrossRef](#)]
13. Parmar, D.N.; Mehta, B.B. Face recognition methods & applications. *Divyvarajsinh N Parmar Int.J.Comput. Technol. Appl.* **2014**, *4*, 84–86.
14. Khandelwal, V.; Verma, V.; Devi, P.R. *Face Recognition Security System*; EasyChair, 2022; pp. 2314–2516. Available online: <https://easychair.org/publications/preprint/pZBJ> (accessed on 15 June 2022).
15. Norouzi, M. A Survey on Face Recognition Based on Deep Neural Networks. *Res. Sq.* **2022**. [[CrossRef](#)]
16. Kaur, H.; Kaur, M.; Mahmood, M.R.; Badhyal, S.; Kaur, S. Automatic Attendance System Using AI and Raspberry Pi Controller. In *Innovations in Electronics and Communication Engineering*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 361–369.
17. Xiao, H.; Hu, Z. Feature-similarity network via soft-label training for infrared facial emotional classification in human-robot interaction. *Infrared Phys. Technol.* **2021**, *117*, 103823. [[CrossRef](#)]
18. Ju, J.; Zheng, H.; Li, C.; Li, X.; Liu, H.; Liu, T. AGCNNs: Attention-guided convolutional neural networks for infrared head pose estimation in assisted driving system. *Infrared Phys. Technol.* **2022**, *123*, 104146. [[CrossRef](#)]
19. Liu, H.; Liu, T.; Zhang, Z.; Sangaiah, A.; Yang, B.; Li, Y.A. Asymmetric Relation-aware Representation Learning for Head Pose Estimation in Industrial Human-machine Interaction. *IEEE Trans. Ind. Inform.* **2022**, *18*, 7107–7117. [[CrossRef](#)]
20. Kar, N.; Debbarma, M.K.; Saha, A.; Pal, D.R. Study of implementing automated attendance system using face recognition technique. *Int. J. Comput. Commun. Eng.* **2012**, *1*, 100. [[CrossRef](#)]
21. Joseph, D.; Mathew, M.; Mathew, T.; Vasappan, V.; Mony, B.S. Automatic Attendance System using Face Recognition. *Int. J. Res. Appl. Sci. Eng. Technol.* **2020**, *8*, 769–773. [[CrossRef](#)]
22. Boyko, N.; Basystiuk, O.; Shakhovska, N. Performance evaluation and comparison of software for face recognition, based on dlib and opencv library. In Proceedings of the 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP), Lviv, Ukraine, 21–25 August 2018; pp. 478–482.
23. Xu, M.; Chen, D.; Zhou, G. Real-Time Face Recognition Based on Dlib. In *Innovative Computing*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 1451–1459.
24. Ambre, S.; Masurekar, M.; Gaikwad, S. Face recognition using raspberry pi. In *Modern Approaches in Machine Learning and Cognitive Science: A Walkthrough*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 1–11.
25. Zhang, D.; Li, J.; Shan, Z. Implementation of Dlib Deep Learning Face Recognition Technology. In Proceedings of the 2020 International Conference on Robots & Intelligent System (ICRIS), Sanya, China, 7–8 November 2020; pp. 88–91.
26. Pattnaik, P.; Mohanty, K.K. AI-based techniques for real-time face recognition-based attendance system-A comparative study. In Proceedings of the 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 5–7 November 2020; pp. 1034–1039.
27. Suwarno, S.; Kevin, K. Analysis of face recognition algorithm: Dlib and opencv. *J. Inform. Telecommun. Eng.* **2020**, *4*, 173–184. [[CrossRef](#)]
28. Pandey, S.; Chouhan, V.; Mahapatra, R.P.; Chhettri, D.; Sharma, H. Real-Time Safety and Surveillance System Using Facial Recognition Mechanism. In *Intelligent Computing and Applications*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 497–506.
29. Shanthi, K.; Sivalakshmi, P. Smart drone with real time face recognition. *Mater. Today Proc.* **2021**. [[CrossRef](#)]
30. Gupta, N.K.; Singh, G. In-Memory Computation for Real-Time Face Recognition. In *Intelligent Computing and Applications*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 531–539.
31. Mustakim, N.; Hossain, N.; Rahman, M.M.; Islam, N.; Sayem, Z.H.; Mamun, M.A.Z. Face Recognition System Based on Raspberry Pi Platform. In Proceedings of the 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), Dhaka, Bangladesh, 3–5 May 2019; pp. 1–4.
32. Shrestha, A. Face Recognition Student Attendance System. 2021. Available online: <https://www.theseus.fi/handle/10024/503517> (accessed on 15 June 2022).
33. Penmatsa, R.; Raju, P.; Priyanka, M. An IoT application for environmental monitoring and control using Raspberry-Pi. *Int. J. Eng. Technol.* **2017**, *9*, 546–552.
34. Hillar, G.C. *Django RESTful Web Services: The Easiest Way to Build Python RESTful APIs and Web Services with Django*; Packt Publishing Ltd.: Birmingham, UK, 2018.

35. Trivedi, A.; Tripathi, C.M.; Perwej, Y.; Srivastava, A.K.; Kulshrestha, N. Face Recognition Based Automated Attendance Management System. *Int. J. Sci. Res. Sci. Technol.* **2022**, *9*, 261–268.
36. Mohanty, S.; Hegde, S.V.; Prasad, S.; Manikandan, J. Design of real-time drowsiness detection system using dlib. In Proceedings of the 2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), Bengaluru, India, 15–16 November 2019; pp. 1–4.
37. Khamket, T.; Surinta, O. Feature Extraction Efficient for Face Verification Based on Residual Network Architecture. In Proceedings of the International Conference on Multi-Disciplinary Trends in Artificial Intelligence, Virtual Event, 2–3 July 2021; pp. 71–80.
38. Li, D.; Cui, Z.; Cao, F.; Cui, G.; Shen, J.; Zhang, Y. Learning State Assessment in Online Education Based on Multiple Facial Features Detection. *Comput. Intell. Neurosci.* **2022**, *2022*, 3986470. [[CrossRef](#)] [[PubMed](#)]
39. Xia, H.; Li, C. Face recognition and application of film and television actors based on DLIB. In Proceedings of the 2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Suzhou, China, 19–21 October 2019; pp. 1–6.
40. Al-Fahsi, R.D.H.; Pardosi, A.P.J.; Winanta, K.A.; Kirana, T.; Suryani, O.F.; Ardiyanto, I. Laboratory attendance dashboard website based on face recognition system. In Proceedings of the 2019 International Electronics Symposium (IES), Surabaya, Indonesia, 27–28 September 2019; pp. 19–23.
41. Wu, H.; Cao, Y.; Wei, H.; Tian, Z. Face recognition based on Haar like and Euclidean distance. *J. Phys. Conf. Ser.* **2021**. [[CrossRef](#)]
42. Jadhav, A.; Lone, S.; Matey, S.; Madamwar, T.; Jakhete, S. Survey on face detection algorithms. *Int. J. Innov. Sci. Res. Technol.* **2021**, *6*, 291–297.
43. Hidai, K.-i.; Mizoguchi, H.; Hiraoka, K.; Tanaka, M.; Shigehara, T.; Mishima, T. Robust face detection against brightness fluctuation and size variation. In Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000), Cat. No. 00CH37113, Takamatsu, Japan, 31 October–5 November 2000; pp. 1379–1384.
44. Tammiseti, A.K.; Nalamalapu, K.S.; Nagella, S.; Shaik, K.; Shaik, K.A. Deep Residual Learning based Attendance Monitoring System. In Proceedings of the 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 25–26 March 2022; pp. 1089–1093.