Real-Time Face Detection and Recognition for Secure Access Control Using Deep Learning

Jannatul Ferdous Esha[1], Lamia Habib[1], Sabrina Subah Nisa[1], and Abu Sayed Md. Mostafizur Rahaman[2*]

[1] Department of Information and Communication Technology, Bangladesh University of Professionals, Dhaka, Bangladesh
[2] Department of Computer Science and Engineering, Jahangirnagar University, Dhaka, Bangladesh
{jannatul.ferdous.esha11235, lamiahabib0123, sabrina.nisa43}@gmail.com, asmmr@juniv.edu*

Abstract. In both the real and virtual worlds, security is still a major concern. In order to improve access control in online meeting environments, this study suggests an effective virtual security solution that integrates face detection and recognition. The system uses the OpenCV module for face identification and the ArcFace model from the InsightFace library with a CNN-based deep learning method for face detection. Real-time authentication via live webcam input is made possible by an intuitive web-based interface. In addition to maintaining distinct databases for authorized and illegal users, the system has an automated email warning system that notifies authorities when an intrusion is detected. The technology is appropriate for real-world security applications since experimental findings show great accuracy and dependability in real-time facial recognition.

Keywords: Face Detection, Face Recognition, ArcFace Model, Real-time Security,Intruder Detection, Web-based Interface

1 Introduction

Security refers to a state's or organization's ability to protect itself from unauthorized access.The most common types of security system lock-key is an extremely insecure security system. the lock is easily breakable. So, some upgraded versions of the security are needed such as biometric authentication and face recognition technologies.

Nowadays face detection and recognition has become popular for user verification.When it comes to face detection and recognition based on recorded or live video, it is possible to think of it as an extension of the research and study of face recognition in still images, which has been thoroughly investigated over the years and yielded some promising results. When using a video-based face detection and recognition system, for example, the face area is confirmed by the face symmetry verification module, face skin validation module, and the eye pattern verification module, all of which are performed in a cascade. The accuracy of these

M. S. Arefin et al. (eds.), *Proceedings of the International Conference on Intelligent Data Analysis and Applications (IDAA 2025)*, Advances in Intelligent Systems Research 206,
https://doi.org/10.2991/978-94-6239-664-7_42

systems has been greatly increased by recent developments in video-based face recognition, making it possible to identify people from live video streams with ease. Applications for these systems are useful in fields including ATM security, border control, and access control in limited locations [2,17]. However, real-time face detection and recognition poses distinct difficulties in guaranteeing prompt and precise identification, particularly in dynamic contexts.The primary goal of many current facial recognition systems is to only confirm identity from pictures or videos. Nevertheless, the majority of these solutions do not actively differentiate between authorized users and possible invaders in real time; instead, they just improve general security. Our solution is unique because it integrates an automated alarm system, highly accurate ArcFace-based recognition, and real-time face detection into an intuitive web interface. This enables the system to deliver notifications via a variety of channels, including email, SMS, and alarms, as well as to quickly identify unauthorized individuals and maintain thorough logs of intruders. Our technology, which offers proactive intrusion management and increased operational transparency, is made for dynamic contexts, such as live online meetings, in contrast to conventional access control techniques.

Any image, no matter how large, small, or blurry, can be used to detect and locate a person's face using Face Detection. Early research on face detection relied on simple criteria such as motion, color, and texture to identify individuals. The statistical face detection methodology published by Viola and Jones is the most widely used of the statistical face detection methodologies available [15]. This face detection method is a variation of the AdaBoost algorithm, which was developed by Google [6]. A face detection framework based on the AdaBoost algorithm and Haar features was proposed by the researchers, which was implemented in MATLAB. Real-time face detection is achievable with high-performance computers; however, face detection tends to take up a disproportionate amount of the system's resources. Because of this, face identification in real time is hampered in its implementation [3]. Almost all real-time face detection research is either theoretical or specifies a software implementation in the some way. Only a few research have looked into the development and production of real-time facial recognition software. McCready worked on the face detection system for the Transmogrifier-2 programmable hardware system, which he designed and constructed. The FPGA boards used in this implementation totaled nine [13]. Sadri and colleagues developed a neural network based face identification system that was implemented on the Virtex-II Pro FPGA [14]. Reduced processing time is achieved through the use of edge detection and skin color filtering. Some tasks, on the other hand, are carried out on a dedicated PowerPC processor that is controlled by ingrained software. Wei et al. reported an FPGA solution for face detection that used scaled input images and fixed-point expressions to identify faces in images [16]. In addition, only a few pieces of the classifier cascade (120x120 pixels) are really applied in the final output. Hiromoto et al. used the AdaBoost algorithm to create a real-time object detection system . A hybrid design was presented that included a parallel processing mechanism for the very first two stages of the cascade and a consecutive processing mod-

ule for the final two levels. However, after processing with fixed Haar feature data, the parallelization processing module and sequential processing module are segregated; the design and implementation of the parallel processing module and sequential processing module must be redone in order to incorporate new Haar features. Along with that experimental result and analysis have not been discussed for the implemented system. In 2018, Ebenezer Owusu et al., demonstrated a face detection approach based on the extraction of haar features and classification by MFNN using the MFNN [13]. His motivation for conducting this research was to improve the accuracy of the system. Haar approaches are used to extract the most important face expressions for this investigation. In 2018, Mehta et al. [16], have presented research on face detection technology that makes use of deep learning algorithms. In that research, he expanded the working of multi-view face detection utilizing Convolution Neural Networks, which were previously employed by Farfade et al., resulting in a critical face detection system that is now available [12]. Ahmed et al. [1] investigated the major steps of brain hemorrhage, including preprocessing, feature extraction, and classification, as well as their findings and limitations. Furthermore, this in-depth examination describes the available benchmark datasets that are used to examine the detection process and compares performance in detail. Esha et al. [5] suggested a Multi-View Soft Attention-Based Convolutional Neural Network (MVSA-CNN) model for classifying lung nodules in three stages: benign, primary, and metastatic.Initially, patches from each nodule are dissected into three separate views, which are then fed into our model to determine the malignancy. Jie et al. [8] designed a more effective face access control system by utilizing deep learning and neural networks with the MXNet framework and ArcFace.This study demonstrates the viability of this approach, verifies the accuracy of face recognition in the data set using mtcnn detection technology and additive angular margin loss, and offers a theoretical foundation for universities to adopt face recognition access control. Using customized modules for face recognition systems in the market, Gülşen et al. [7] examined the impact of pre-processing techniques such face identification, image cropping, image scaling, and image normalization on the performance of face recognition algorithms. The most effective approach and module combinations that will boost success are determined by the study. According to the study, all flows performed better when the suggested strategy was used during the pre-processing phase.

Li et al. [9] used the VGG-Face model for gender, expression, and age recognition on the UCEC-Face, along with four models, including OpenFace and ArcFace, for face verification. Because it is more akin to the real world, the experimental results demonstrate that the UCEC-Face developed in this paper is more difficult to verify in face verification tasks.Manesco y Marana [11]has suggested a new method for periocular recognition based on attention mechanisms that combines a recent ViT architecture with the ArcFace loss function. Experimental results on UBIPr and FRGC, two popular datasets, revealed that the proposed method achieved lower error rates when compared to other cutting-edge periocular recognition methods, as well as the ability to visualize attention weights

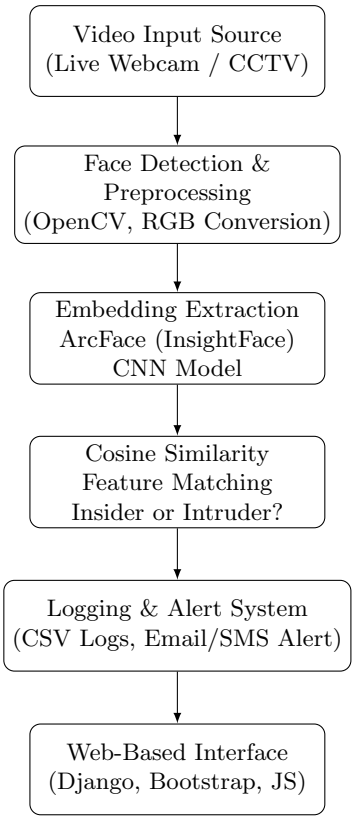


Fig. 1: System architecture of the proposed real-time face detection and authentication framework.

for a better understanding of the most important periocular regions used by the neural network for biometric recognition. Liu et al. [10] demonstrated that the upgraded YOLOv3 is a high-performance, lightweight model. The electric power worker identification approach suggested in this paper features a fast recognition procedure and produces accurate results. Improved detection performance and robustness demonstrate the approach's usefulness.

2 METHODOLOGY

Our system is designed with two key components: face detection and face recognition, which work together to identify potential intruders in real time. First, the system scans the video feed to spot any human faces. Once a face is found, it's checked against a database of trusted or authorized individuals. If the face doesn't match anyone in the system, it's treated as a possible intruder. At that moment, the system automatically records the time, snaps a photo from the

video feed, and immediately sends an alert to the administrator or someone responsible for monitoring usually by email. This quick response helps strengthen security by catching suspicious activity before it becomes a threat. For recognizing faces, we use a model called ArcFace, which is known for accurately turning facial features into data. To compare faces, we rely on cosine similarity, a method that helps ensure matches are as precise as possible.

To provide a clearer understanding of the implementation, Figure 1 illustrates the system architecture, which includes: (1) the video input layer for capturing live webcam or CCTV streams; (2) the face detection and embedding extraction module using ArcFace and CNN-based InsightFace; (3) the authentication engine, where embeddings are compared using cosine similarity; (4) the logging and alert module that generates records and sends automated notifications; and (5) a Django-based web interface for real-time monitoring, intrusion alert review, and user management. This architecture shows how all components operate together to deliver secure real-time face authentication and alerting.

2.1 Dataset

A custom dataset was created for this study, consisting of 100 images of 100 different individuals. Each image is stored in JPG format with a resolution of 224x224 pixels. The dataset includes a mix of genders, ages, and facial expressions. Images were captured under varying lighting conditions and backgrounds to simulate realistic scenarios. All images were manually annotated to ensure accurate labeling. The dataset provides a representative sample for evaluating real-time face recognition in controlled environments. As shown in Figure 2, the dataset is intentionally small, designed to simulate a controlled environment with a limited number of authorized personnel like what you'd expect in a secure office or lab setting. We're aware that the size of our dataset brings certain limitations. A smaller dataset can make it harder for the model to generalize well to broader, real-world scenarios. While the system performs well within this specific setup, it will need to be tested on larger and more diverse datasets to confirm how well it holds up outside of this controlled environment.

Fig. 2: Sample images from the custom face recognition dataset.

2.2 Proposed Model

Figure 3 illustrates the proposed design of a face recognition system, which integrates seamless face detection, embedding extraction, comparison, and alert generation to enable robust real-time identification of authorized individuals. The primary purpose of this system is to provide an automated, accurate, and efficient method for identifying authorized individuals (insiders) and unauthorized individuals (intruders) in restricted areas, i.e., live meetings in real-time. Figure 3 illustrates the complete workflow of the proposed authentication pipeline. There are several steps involved in the authentication process, including input face detection, face recognition, log creation, and an alerting system. The system accepts various types of inputs, such as live video from CCTV cameras, still images, and recorded video clips. With a wide range of applications, including surveillance, access control, and monitoring, the system is designed to function across multiple use cases. Specifically, during live streaming of real-time meetings, the system verifies each individual separately and grants access only to authorized users. The ArcFace model (InsightFace library) is used for the face detection process. The system extracts unique 512-dimensional vectors that represent the facial features of each individual. These embeddings are then compared with the stored embeddings of known insiders in the database. Two separate logs (insider log, intruder log) are maintained to manage insider and outsider information. If an outsider is detected and their facial embedding does not match any insider's, the system triggers an alert. Alerts are sent via email, SMS, or alarms to notify security personnel.

2.3 Face Detection

A technique referred to as face detection is often utilized in face analysis to determine which areas of a still image, video, or live footage should be focused on in order to assess age, gender, and emotions based on the gestures on the faces of individuals within the image or video. Face detection data is important for the algorithms that ascertain which components of an image or video are required to develop a faceprint during a face recognition system, which quantitatively maps a person's countenance and saves the data as a faceprint. Face detection data is additionally required for the algorithms that determine which parts of an image or video are needed to create a faceprint during a face recognition system, which maps a person's countenance mathematically and stores the information as a faceprint. It is then possible to match the newly created faceprint with those that have already been saved to determine whether or not there is a match. Face detection is the first critical step in the face recognition pipeline, which is based on the ArcFace model [4], a deep learning-based approach that leverages Convolutional Neural Networks (CNNs) to identify faces with high accuracy and real-time performance. In the preprocessing step, the input image or video frame needs to be compatible with the detection model. Therefore, it is converted to RGB format for compatibility with ArcFace, ensuring that it is correctly

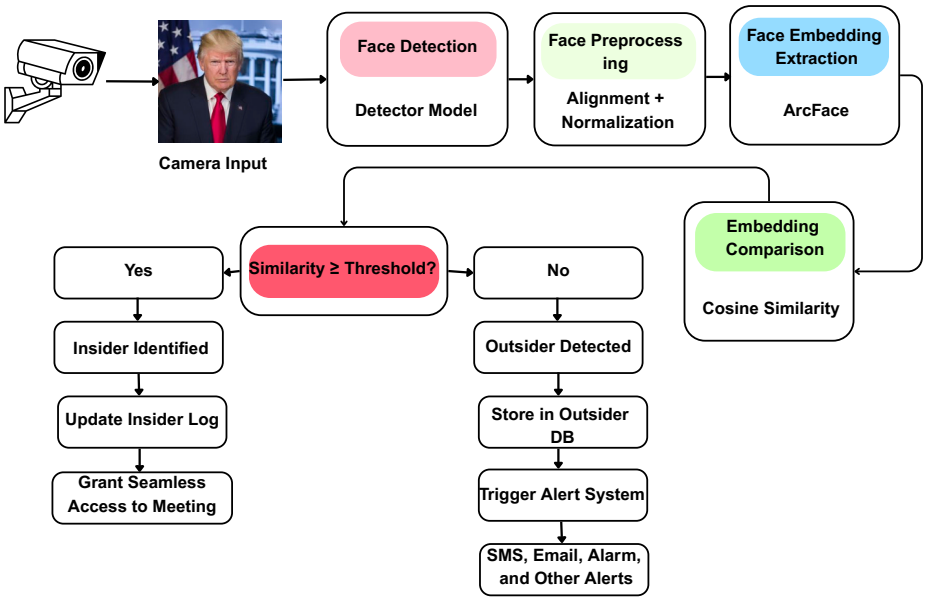


Fig. 3: Workflow of the proposed face authentication system.

formatted for the face detection model:

$$\text{Image}_{\text{rgb}} = f(\text{Image}_{\text{bgr}}) \tag{1}$$

Then, ArcFace employs CNN-based architectures to localize faces in the input image. The model outputs a set of bounding boxes for the detected faces, represented by the coordinates $[x_1, y_1, x_2, y_2]$, where (x_1, y_1) and (x_2, y_2) are the top-left and bottom-right corners of the bounding box:

$$\text{Bounding Box} = \{[x_1, y_1, x_2, y_2]\} \tag{2}$$

After detection of the face, the image is cropped based on the coordinates of the bounding box. The cropped area corresponds to the face region ready for further extraction of embeddings or features:

$$\text{Face Crop} = \text{Image}[y_1 : y_2, x_1 : x_2] \tag{3}$$

The purpose of cropping the images using the appropriate bounding box coordinates is to facilitate the next steps in the face recognition pipeline. One of the most up-to-date facial detection and identification systems is ArcFace. By making the features more discriminative, it has enhanced current techniques. The unique traits of a person's face are preserved since each identified face is recorded as a 512-dimensional embedding. After passing through a series of convolutional layers, the ArcFace model generates the embeddings—the distinct representations of a face. The embedding is computed using a deep convolutional neural

network (DCNN), and the output is a feature vector E_{face} for each detected face:

$$E_{\text{face}} = f(I_{\text{face}}) \tag{4}$$

where I_{face} is the cropped face image. Normalization is performed by means of L2-normalization. Both the embeddings and the weight vectors are normalized to unit length to ensure that the distances between them reflect angular differences rather than Euclidean distance:

$$\hat{E}_{\text{face}} = \frac{E_{\text{face}}}{\|E_{\text{face}}\|} \quad \text{and} \quad \hat{W}_j = \frac{W_j}{\|W_j\|} \tag{5}$$

where W_j represents the weight vector for class j.

2.4 Face Recognition

Facial recognition is the process of identifying or validating an individual based on their facial traits. In order to detect specific, differentiating aspects of an individual's face, face recognition systems employ computer algorithmic techniques. These characteristics, such as the distance between the eyes, the length and width of the nose, or the shape of the chin, are translated into a mathematical representation. This representation is then compared against data from other faces stored in a face recognition database to determine whether the face is recognized. The ArcFace model excels in this task because it can match faces to a database of known identities using high-quality embeddings. Firstly, the ArcFace model detects faces in the input image. The model generates a 512-dimensional embedding E_{test} for the detected face:

$$E_{\text{test}} = f(I_{\text{test}}) \tag{6}$$

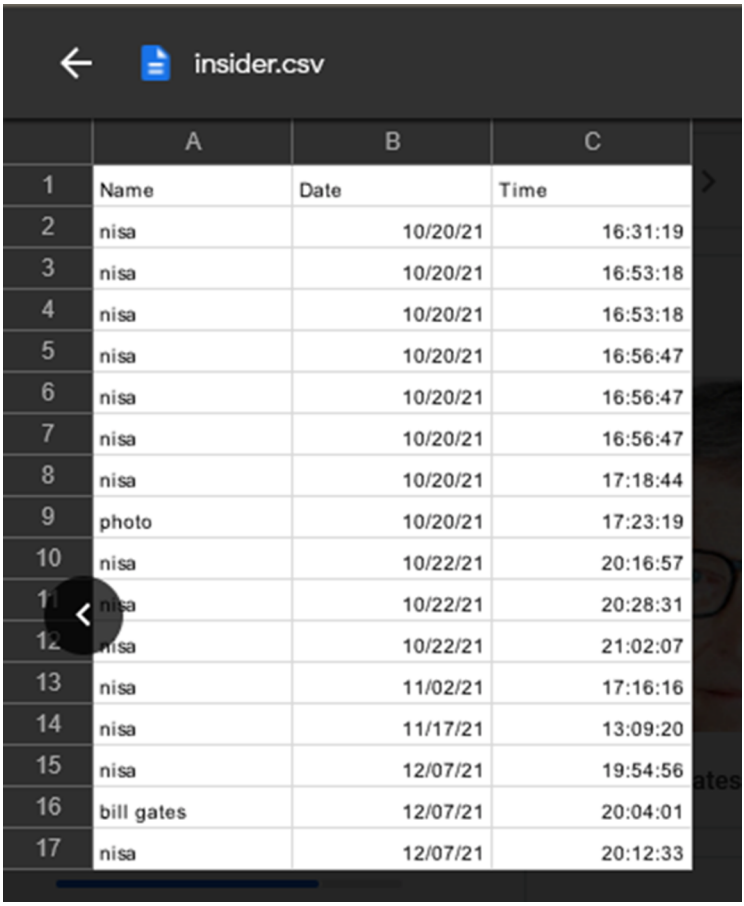
where I_{test} is the input face image. The system compares the extracted embedding E_{test} with the stored embeddings E_{known} using Cosine Similarity:

$$\text{Cosine Similarity} = \frac{E_{\text{test}} \cdot E_{\text{known}}}{\|E_{\text{test}}\| \|E_{\text{known}}\|} \tag{7}$$

If the Cosine Similarity score exceeds a predefined threshold (e.g., 0.5), the individual is recognized as an insider. Otherwise, the person is classified as an outsider. The system employs a similarity threshold $T_{\text{threshold}}$ to classify individuals as follows:

$$\text{If Cosine Similarity} > T_{\text{threshold}} \Rightarrow \text{Insider (Access Granted)} \tag{8}$$

$$\text{If Cosine Similarity} \leq T_{\text{threshold}} \Rightarrow \text{Outsider (Access Denied)} \tag{9}$$



insider.csv

	A	B	C
1	Name	Date	Time
2	nisa	10/20/21	16:31:19
3	nisa	10/20/21	16:53:18
4	nisa	10/20/21	16:53:18
5	nisa	10/20/21	16:56:47
6	nisa	10/20/21	16:56:47
7	nisa	10/20/21	16:56:47
8	nisa	10/20/21	17:18:44
9	photo	10/20/21	17:23:19
10	nisa	10/22/21	20:16:57
11	nisa	10/22/21	20:28:31
12	nisa	10/22/21	21:02:07
13	nisa	11/02/21	17:16:16
14	nisa	11/17/21	13:09:20
15	nisa	12/07/21	19:54:56
16	bill gates	12/07/21	20:04:01
17	nisa	12/07/21	20:12:33

Fig. 4: Log file showing entries of authorized individuals.

2.5 Face Feature Extraction and Matching

A face was detected and its embedding extracted, and the system matched the newly obtained embedding to those in its database of known insiders. The **Cosine Similarity** method compares the matching of the new face to those of insiders. The system generates a facial embedding for each detected face E_{test}. The system calculates the Cosine Similarity between the test face's embedding and the stored embeddings. If the similarity score exceeds a certain threshold, the individual is recognized.

$$\text{Cosine Similarity} = \frac{E_{\text{test}} \cdot E_{\text{known}}}{\|E_{\text{test}}\| \|E_{\text{known}}\|} \tag{10}$$

The best match is the one with the highest similarity score. If no match is found, the person is classified as an outsider. The system maintains two separate logs:

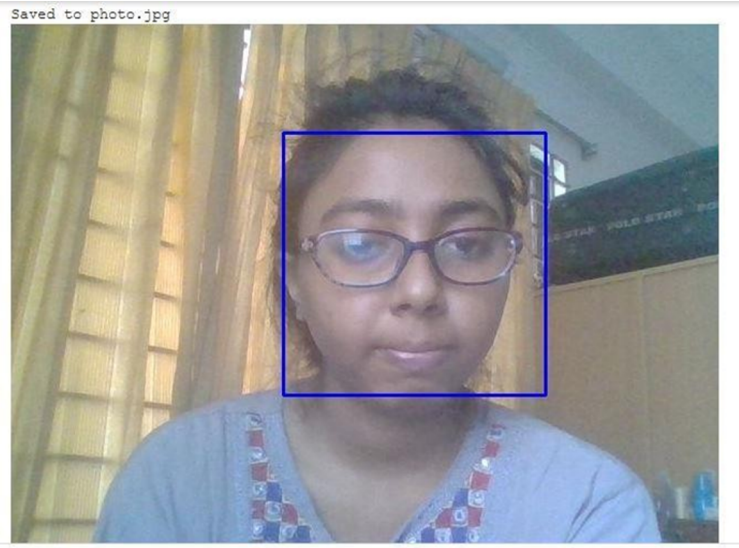


Fig. 5: System detects and identifies a person.

- **Insiders Log:** Records information about identified individuals, including their embeddings and any pertinent personal information.
- **Outsiders Log:** Records information about unidentified individuals, including images, embeddings, and timestamps.

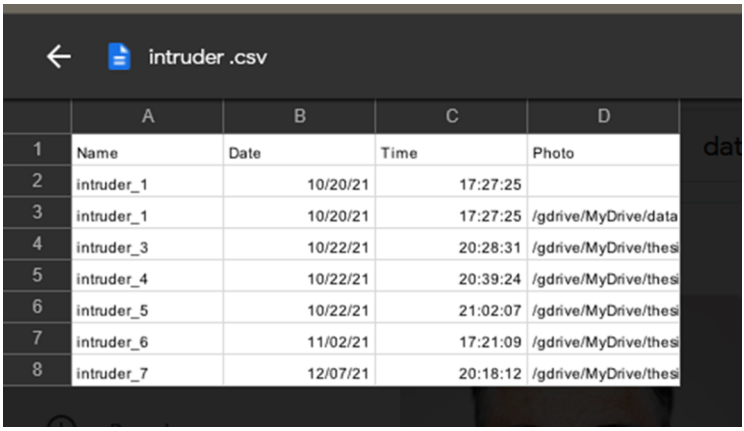
When the system detects an outsider whose embedding does not match the list of recognized insiders, it issues an alarm. It updates the Outsiders Log with outsider information, including photos, timestamps, and embeddings. Alerts are sent to security staff for appropriate action. Notification methods can include email alerts and on-site alarms.

3 EXPERIMENT AND RESULT

The proposed system was developed and tested on a high-performance workstation featuring an NVIDIA RTX 4090 GPU, an AMD Ryzen 9 7950X CPU, and 64 GB of DDR5 RAM, which easily handled both the Django backend and real-time face detection. While this setup ensured smooth performance, we recognize that not all deployments will have such powerful hardware. To support lower-power devices like embedded or edge systems, we explored optimizations like model pruning, quantization, and edge computing. These techniques improve efficiency and make the system more scalable across different hardware setups.

3.1 Face Recognition Performance

We evaluated the proposed system using a custom dataset consisting of 100 images from 100 different individuals, split into 80% for training and 20% for



intruder .csv

	A	B	C	D
1	Name	Date	Time	Photo
2	intruder_1	10/20/21	17:27:25	
3	intruder_1	10/20/21	17:27:25	/gdrive/MyDrive/data
4	intruder_3	10/22/21	20:28:31	/gdrive/MyDrive/thesi
5	intruder_4	10/22/21	20:39:24	/gdrive/MyDrive/thesi
6	intruder_5	10/22/21	21:02:07	/gdrive/MyDrive/thesi
7	intruder_6	11/02/21	17:21:09	/gdrive/MyDrive/thesi
8	intruder_7	12/07/21	20:18:12	/gdrive/MyDrive/thesi

Fig. 6: Log file of intruder who have not the permission to enter the restricted area

testing. The ArcFace model was compared with several widely used face recognition methods, including VGG-Face, Eigenfaces, and Haar Cascade with LBPH. Evaluation metrics included accuracy, precision, recall, F1-score, and average latency per recognition.

As shown in Table 1, ArcFace achieved perfect face verification with an accuracy of 100%, outperforming VGG-Face (98.9%), Eigenfaces (96.7%), and Haar Cascade with LBPH (76.9%). It also achieved the highest precision, recall, and F1-score, demonstrating its ability to correctly identify authorized individuals while rejecting intruders. The average recognition latency was 1.2 seconds per face, indicating the system is suitable for real-time applications. The superior performance of ArcFace is attributed to its additive angular margin loss, which enhances feature discrimination and improves robustness against variations in lighting, pose, and occlusion.

Table 1: Performance Comparison of Face Recognition Models on the Custom Dataset

Model	Accuracy (%)	Precision	Recall	F1-score	Latency (s)
ArcFace	100.0	1.00	1.00	1.00	1.2
VGG-Face	98.9	0.99	0.99	0.99	1.5
Eigenfaces	96.7	0.97	0.97	0.97	1.0
Haar Cascade + LBPH	76.9	0.77	0.77	0.77	0.8

Fig. 7: User interface for Secure User Authentication and Meeting Management: Featuring Sign-Up, Login, Identity Verification, Intruder Alerts, and Seamless Meeting Control.

3.2 Detection Log

When the system recognizes a known person, it logs their name, date, and time in a file called insider.csv. This is shown in Figure 4. When a face is detected (Figure 5), a blue square appears around it. If the face matches someone in the authorized list, their information is saved in the insider log. The system matches the detected face to the known face database and if it does not match, the person is claimed as an outsider . After an unknown face is detected the date, time and the image of that unknown face is saved in the intruder CSV file 6.

It is updated every time the system catches an intruder. It also updates a web interface, which allows administrators to track intruders and manage users. Figure 7 shows the user interface of the web application.

At the same time, the information is sent to the alerting system, which checks the unknown face against a database of recently added authorized people. This verification can be done automatically or reviewed manually by security staff to reduce false alarms. Once an intruder is confirmed, alerts are sent through multiple channels to ensure a quick response. Security staff receive email notifications that include the intruder's image, detection time, and location, while on-site alarms are also triggered to draw immediate attention. We tested the alerting system extensively in simulated real-world situations. The results showed it responds quickly, in less than 1.2 seconds detects intruders accurately, and works reliably under different lighting and environmental conditions. These results prove the system is effective at keeping security strong during live use and show how reliable it is in everyday operations.

3.3 Interface Design and Technologies Used

As shown in Figure 7, the user interface of the project was built using Django, CSS, Bootstrap, and JavaScript to create a dynamic and responsive web application. Django took care of the backend to provide efficient management of dynamic content and user authentication. CSS provided tailored styling to give the interface a visually appealing and consistent layout, and Bootstrap ensured that the interface became responsive across devices. JavaScript was used to incorporate interactivity to support dynamic features such as form validation and refreshing content without reloading pages. The UI remained minimal and simple, with the facility to browse and access principal features with ease. Bootstrap grid system ensured the design automatically responded to different screen sizes, and custom JavaScript extended user experience through enabling real-time interaction. Dynamic content challenges and layout adjustments were addressed through combining Bootstrap's utilities and custom JavaScript solutions.

Overall, the application is well-balanced in its functionality and aesthetics, providing a seamless user experience. The employment of these technologies, especially with JavaScript, offered an interactive and responsive setting. Future improvements may involve performance improvement and incorporating additional interactive features.

4 Conclusion

Security is a top priority for organizations and individuals alike, as it protects valuable assets and sensitive information from harmful intruders. Unauthorized access can cause serious damage not only to the affected organization but also to the wider community. Our research presents a comprehensive security system that combines deep learning-based face detection and recognition with a real-time web interface to enhance traditional surveillance methods. Using the ArcFace model from the InsightFace library, our system can accurately identify unauthorized people in restricted areas, allowing for quick response. The web interface, built with Django, CSS, Bootstrap, and JavaScript, supports live webcam feeds for real-time monitoring, easy management of authorized and unauthorized user records, and an automated alert system that notifies security staff as soon as an intruder is detected. This integration improves control, user experience, and the ability to scale the system. However, deploying this system comes with challenges. Privacy is a major concern since continuous face recognition raises questions about data protection and user consent, which must be handled carefully to meet legal and ethical standards. Additionally, the system's accuracy may vary depending on lighting and environmental conditions, which means more testing and adjustments are necessary to ensure it works well in different real-world settings. Looking ahead, there is great potential to enhance this system by adding hardware components to create a more compact and automated security solution. Despite these challenges, our proposed system offers an advanced and proactive approach to security, designed to prevent incidents before they happen and improve safety in sensitive environments.

References

1. Ahmed, S., Esha, J.F., Rahman, M.S., Kaiser, M.S., Hosen, A.S., Ghimire, D., Park, M.J.: Exploring deep learning and machine learning approaches for brain hemorrhage detection. IEEE Access (2024)
2. Australian Border Force: Smartgates arrivals (2024), https://www.abf.gov.au/entering-and-leaving-australia/smartgates/arrivals, accessed: 2025-04-28
3. Cho, J., Mirzaei, S., Oberg, J., Kastner, R.: Fpga-based face detection system using haar classifiers. In: Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays. pp. 103–112 (2009)
4. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: Arcface: Additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4690–4699 (2019)
5. Esha, J.F., Islam, T., Pranto, M.A.M., Borno, A.S., Faruqui, N., Yousuf, M.A., Azad, A., Al-Moisheer, A.S., Alotaibi, N., Alyami, S.A., et al.: Multi-view soft attention-based model for the classification of lung cancer-associated disabilities. Diagnostics **14**(20), 2282 (2024)
6. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences **55**(1), 119–139 (1997)

7. Gülşen, M.F., Taşkıran, M., Taşçı, S.E., Kahraman, N.: An optimal flow for face recognition: Analysis/effects of face detection, alignment and cropping techniques. In: 2024 26th International Conference on Digital Signal Processing and its Applications (DSPA). pp. 1–6. IEEE (2024)
8. Jie, Y., Jiong, M., Baiyi, S., Mingxin, L., Shu, W., Zhiyou, W.: Face recognition of students based on arcface. In: Proceedings of the 2020 International Conference on Aviation Safety and Information Technology. pp. 678–681 (2020)
9. Li, N., Shen, X., Sun, L., Xiao, Z., Ding, T., Li, T., Li, X.: Chinese face dataset for face recognition in an uncontrolled classroom environment. IEEE Access **11**, 86963–86976 (2023)
10. Liu, Q., Hao, F., Zhou, Q., Dai, X., Chen, Z., Wang, Z.: An effective electricity worker identification approach based on yolov3-arcface. Heliyon **10**(4) (2024)
11. Manesco, J.R.R., Marana, A.N.: Combining arcface and visual transformer mechanisms for biometric periocular recognition. IEEE Latin America Transactions **21**(7), 814–820 (2023)
12. Mehta, J., Ramnani, E., Singh, S.: Face detection and tagging using deep learning. In: 2018 International Conference on Computer, Communication, and Signal Processing (ICCCSP). pp. 1–6. IEEE (2018)
13. Owusu, E., Abdulai, J.D., Zhan, Y.: Face detection based on multilayer feedforward neural network and haar features. Software: Practice and Experience **49**(1), 120–129 (2019)
14. Sadri, M.S., Shams, N., Rahmaty, M., Hosseini, I., Changiz, R., Mortazavian, S., Kheradmand, S., Jafari, R.: An fpga based fast face detector. In: Global Signal Processing Expo and Conference (2004)
15. Viola, P., Jones, M.J.: Robust real-time face detection. International journal of computer vision **57**, 137–154 (2004)
16. Wei, Y., Bing, X., Chareonsak, C.: Fpga implementation of adaboost algorithm for detection of face biometrics. In: IEEE International Workshop on Biomedical Circuits and Systems, 2004. pp. S1–6. IEEE (2004)
17. Zhang, P.: A video-based face detection and recognition system using cascade face verification modules. In: 2008 37th IEEE Applied Imagery Pattern Recognition Workshop. pp. 1–8. IEEE (2008)