International Journal of Frontier Technology and Engineering (IJFTE)
Vol. 04 No. 01, December 2025, pp. 45-52

ESP32-CAM Based Smartdoor with Cloud Services Face Recognition

Rosa Andri Asmara[1]

[1] *Department of Information Technology, Politeknik Negeri Malang, Malang, Indonesia*

Corresponding Author: Rosa Andri Asmara, rosa.andrie@polinema.ac.id

Received Date: 10-10-2025 Revised Date: 14-11-2025 Accepted Date: 21-12-2025

Abstract

Technological advancements have made robust security systems essential, especially for sensitive facilities such as server rooms, which house critical services and data. These spaces are typically restricted to authorized individuals, but conventional locks often prove inadequate due to the risks of damage and duplication. Digital technology offers a more secure solution through biometric systems to address these challenges. Facial recognition, known for its high accuracy and resistance to manipulation, is a key to improving security. This research utilizes the DeepFace library, which integrates pre-trained models like VGG-FACE and FaceNet, significantly improving system efficiency by reducing training time. Testing showed that DeepID with the OpenCV backend achieved the fastest encoding time, while VGG-FACE with MTCNN provided the highest accuracy. The encoding times ranged from 66.8 to 114.2 seconds, with accuracy varying from 18% to 90% and optimal results were achieved at a distance of 30 cm. These findings provide valuable information for the implementation of effective facial recognition systems.

Keywords : Facial Recognition, Digital Key System, IoT, CNN, DeepFace Library, ESP32-CAM

1 Introduction

Security is one of the crucial things along with the rapid development of technology, not least in particular areas or facilities with a strict level of security so that not just anyone can access the facility. In this case, conventional key systems still need to be more efficient and less practical because they carry many keys and are often forgotten or lost. In addition, this type of lock is also easy to break into, so its security is not guaranteed [1]. To overcome these problems, facial recognition technology can be applied to improve security in door lock systems.

Face recognition is a face identification and verification process in which the system performs face detection and recognition or a face comparison with facial data in the database[2]. According to Ulla Delfana Rosiani et al., each person's face reveals various facts such as expression, gender, age, and race. Therefore, the face can be an identifier in biometric or biological data recognition technology [3]. The application of face recognition technology to digital key systems requires artificial intelligence technology with appropriate algorithms. This research uses a deep learning algorithm to perform the face recognition process. Deep learning is a branch of machine learning that uses artificial neural networks to solve problems using large amounts of data [4]. One of the deep learning algorithms that can be used in face recognition is the Convolutional Neural Network (CNN). CNN is an effective neural network for processing images, classifying and recognizing patterns because its architecture mimics how the human brain processes visuals [5]. Research conducted by Alsyayadeh shows that systems using Convolutional Neural Networks (CNN) and ESP32-CAM achieve a face recognition accuracy of 98.48% [6]. The research conducted by Damale and Pathak shows that CNN can outperform machine learning methods such as SVM and MLP in face recognition, with a precision of 98% [7]. Other research conducted by Adi Nugroho P et al. shows that the CNN method is very suitable for testing an image because of its multi-layered process, evidenced by the correct predictions of 28 out of 35 images with different expressions

[8]. However, CNN only provides optimal results on large datasets, and CNN requires a lot of data and computing resources to train [9]. Pre-trained models can be utilized to overcome this when the available dataset is limited. Based on research conducted by Wijaya on implementing face recognition technology in the server room lock system with the Triangle Face algorithm. The Triangle Face algorithm considers the distance between facial features such as the nose, eyes, and mouth. This method, from 30 image data, produces an accuracy of 93.3%Ḣowever, using the Raspberry Pi is considered less than optimal because CPU resources are excessively used, preventing the detection of facial features [10].

Then, Hanuebi conducted other research on a face recognition system implemented as a key with biometric authentication on a Raspberry Pi by applying the LBPH (Local et al.) algorithm. The results of the tests carried out showed that the average face recognition time was 2.364299631118 seconds. The ideal light conditions were 100 - 120 lux meters and the ideal distance between the facial object and the camera module was 40 - 60 cm [11]. Furthermore, based on research conducted by Dang on a smart home management system equipped with biometric authentication in the form of face recognition and hand gestures, the modeling used is ArcFace with a MobileNetV2 backbone. The results of this study show that ArcFace's accuracy is better than FaceNet's. The ArcFace model can also increase the accuracy of the previous model from 92% to 97% [12].

Then, based on research by Asmara on developing a digital attendance system on the Google Cloud Platform by utilizing the deepface library, the pre-trained model chosen is ArcFace as a face recognizer and RetinaFace as a face detector. The results of this study indicate that the highest average accuracy is 100% provided that the image data entered is natural (without a mask). The lowest accuracy is 20% with the image data used using a mask [13]. In another study, Asmara compared the Haar Cascade and CNN face detection methods, CNN proving to be more accurate, reaching 86.53% compared to Haar Cascade, which is only 81.12% [14]. DeepFace is a lightweight Python library for face recognition and facial attribute analysis. This open-source library provides pre-trained models, including VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace, DeepID, ArcFace, Dlib, and Sface [15]. In this research, a deep learning method with a CNN algorithm using a pre-trained model in the Deepface library is proposed because some research has proven that it can perform image processing tasks very well.

2 Method

The pivotal stages in this research project, which lay the groundwork for the entire process, are requirements analysis, system design, implementation, and testing. The analysis and design phases are initiated by identifying the functional requirements crucial for achieving the system's objectives. Subsequently, the system design is executed to generate the following diagrams.

2.1 System Architecture

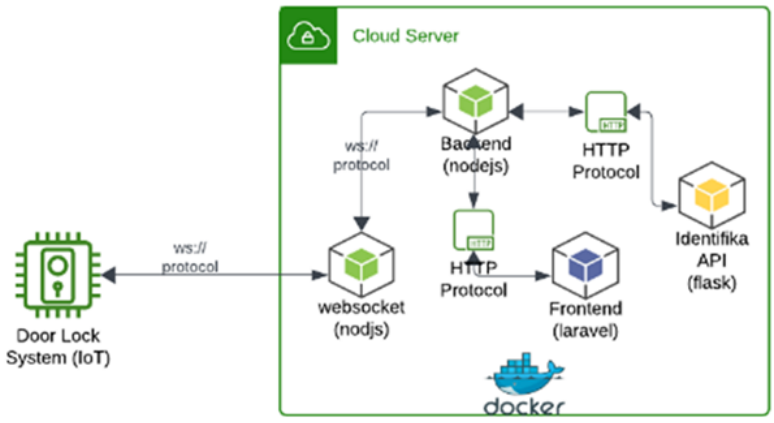

Figure 1: System Architecture Design

Based on Figure 1, the system is divided into two main parts: the door lock system, which is an IoT device, and the cloud server, which consists of several web-based applications running in Docker containers. The system work process starts when the IoT device detects a face; if a face is detected, the device will send image data in base64 format to the backend via the WebSocket server. The WebSocket server then receives the message from the IoT device and forwards it to the backend. The backend then processes the image by sending it to the Identifika API via HTTP protocol for face recognition. Once the Identifika API receives this request, the face recognition process is performed and the response is sent back to the backend. The backend compares the received response with the database to determine if the individual can access the space. Finally, the backend sends a message to the IoT device via WebSocket, which then executes the function of opening or not opening the door lock solenoid based on the results received from the backend.

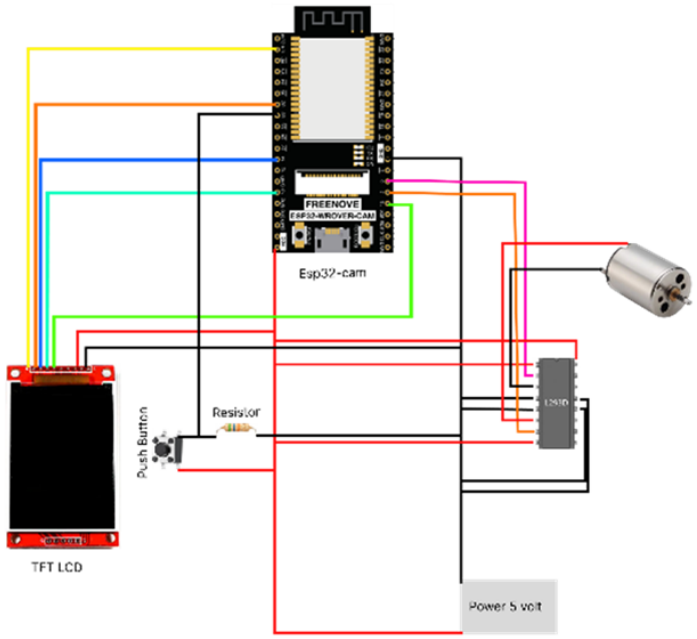

Figure 2: IoT Design

2.2 IoT Design

The series of components of the IoT device will look like Figure 2, where several components include esp32-cam, TFT LCD, push button, DC motor, and IC L293D. Note in the sketch that the red and black cables were used explicitly as power with the provision of positive (+) flow in red and negative (-) flow in black. Most of the gpio (general purpose input/output) on the esp32-cam has been used for the camera, so the determination of gpio for other components must be considered to avoid errors during the development process. Because gpio 22, 23, 21, 19, 18, 5, 4, 36, 39, 25, 26, and 27 have been used for the camera, the TFT LCD will be connected to esp32-cam on gpio 13, 14, 15, and 32. Then, to control the push button, gpio 33 is used on the esp32-cam. Finally, to regulate the rotation of the DC motor, gpio 0 and 2 will be used to trigger the door lock to open or close.

2.3 Testing

The first test, model testing on the deep face library, was carried out using photos of two different people. Each person had ten photos, so in total, there would be 20 photos of two people and two different resolutions in one folder. This test is intended to find the combination of models and backend detectors with the fastest

encoding time and highest accuracy.The second test focuses on testing the prototype of the digital lock system. This test was carried out under dim and bright lighting conditions with participants facing the camera integrated into the IoT device from three predetermined distances, namely 30 cm, 50 cm, and 70 cm. The test procedure begins with each participant alternately standing facing the camera at a distance of 30 cm, 50 cm, and 70 cm for 5 times at each distance. This test aims to ensure that the IoT device is running according to the predetermined plan.The last test, the threshold test, was conducted with three participants, each providing ten photos in bright and 5 in dim lighting. The threshold values tested were 0.3, 0.4, 0.5, and 0.6. The objective of this test was to identify the most effective threshold value that could be applied to the digital key system.

3 Result And Discussion

3.1 Model Testing Result on Deepface Library

The tests were conducted under two conditions: photos with good resolution and photos with poor resolution obtained from esp32-cam cameras. The models and backend detectors in the deep face library were combined to find the difference in encoding time in each combination, resulting in comparative data as follows.

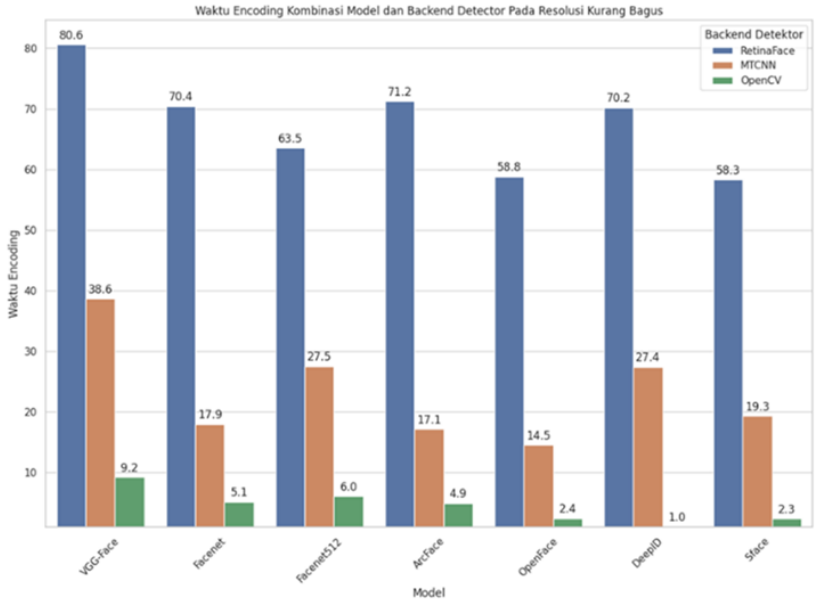

Figure 3: Comparison of Encoding Time on Photos with Poor Resolution

Based on the graph in Figure 4, it can be seen that in photo conditions with good resolution, the combination of models and backend detectors with the fastest encoding time is DeepID and OpenCV, which is 6 seconds. Then, the combination of models and backend detectors with the longest encoding time is VGG-FACE and RetinaFace, which is 187.4 seconds.

Based on the graph in Figure 4, it can be seen that in photo conditions with good resolution, the combination of models and backend detectors with the fastest encoding time is DeepID and OpenCV, which is 6 seconds. Then, the combination of models and backend detectors with the longest encoding time is VGG-FACE and RetinaFace, which is 187.4 seconds. Furthermore, testing is continued by comparing the accuracy of the combinations of models and backend detectors in the deep face library to find combinations with high accuracy and produce data as follows.

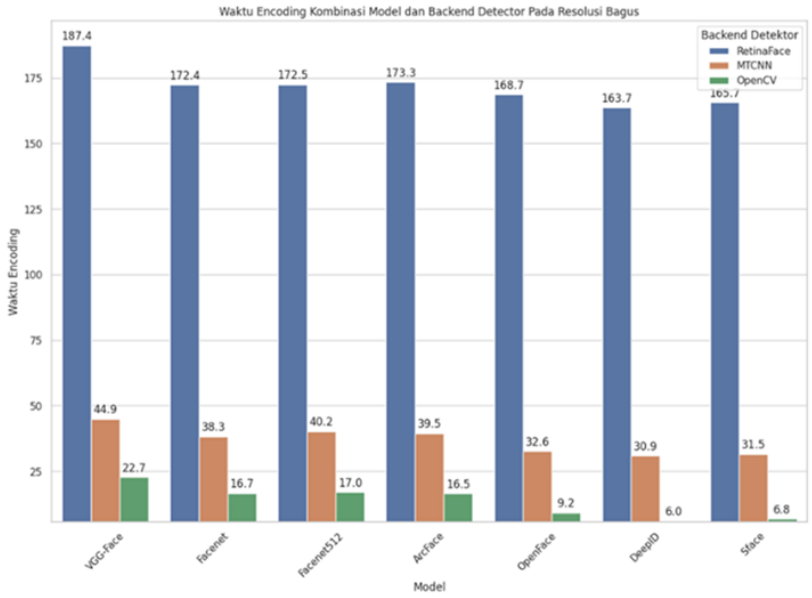

Figure 4: Comparison of Encoding Time on Photos with Good Resolution

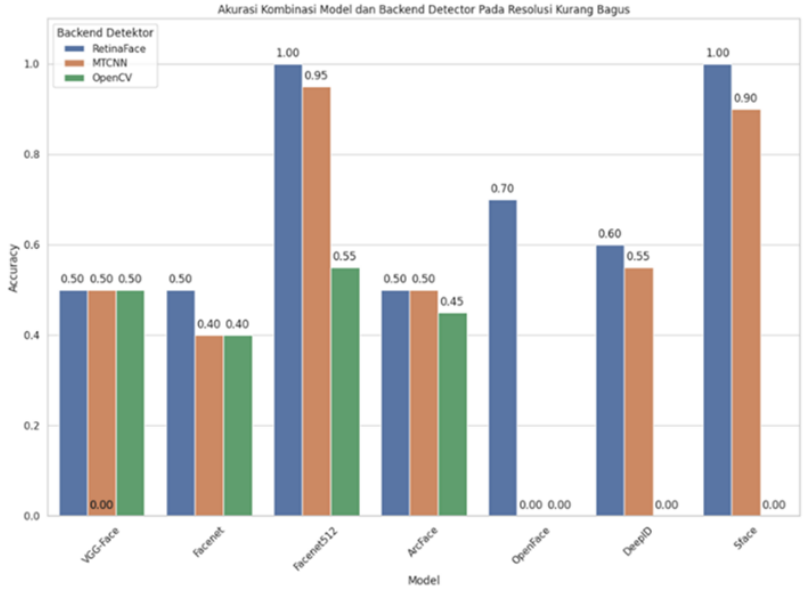

Figure 5: Comparison of Accuracy on Photos with Poor Resolution

Based on the graph in Figure 5, it can be seen that in photo conditions with poor resolution, the combination of models and backend detectors with the highest accuracy is Facenet512 and RetinaFace; Sface and RetinaFace, which is 100%.Then, the combination of the model and backend detector with the lowest accuracy is OpenFace and MTCNN; OpenFace and OpenCV; DeepId and OpenCV; and Sface and OpenCV, which is 0%

In the graph in Figure 6, it can be seen that in photos with good resolution, the combination of models and backend detectors with the highest precision is ArcFace and MTCNN, which is 95%.Then, the model and backend detector with the lowest accuracy combination are OpenFace and OpenCV, DeepId and MTCNN, DeepId and OpenCV, and Sface and RetinaFace, which is 0%

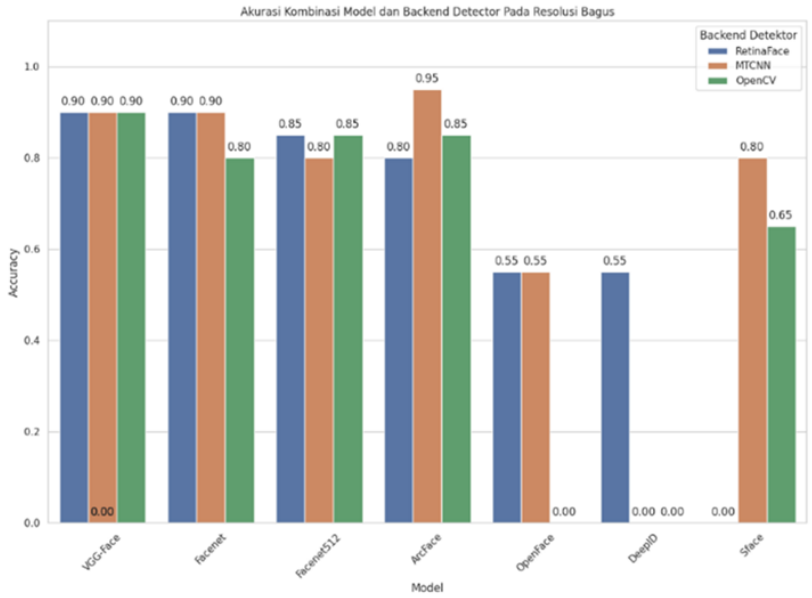

Figure 6: Comparison of Accuracy on Photos with Good Resolution

3.2 System Prototype Testing Result

The next test is to test the prototype by simulating facial recognition on an IoT device with a distance of 30 cm, 50 cm, and 70 cm 5 times in each room with light intensity and distance, each resulting in the data as follows.

Table 1: Threshold Testing Result

Lighting	**Threshold**	**Object 3 people with a distance of 30 cm**	
		True	**False**
Bright (520 lux)	30 cm	29	1
	50 cm	28	2
	70 cm	0	0
Dim (82 lux)	30 cm	16	10
	50 cm	14	11
	70 cm	0	0

Based on the data in Table 1, the distance between the camera and the best face in bright room conditions (520 lux) is 30 cm, with a total of 29 correct detections and 1 false detection. Then, in dim room conditions (82 lux), the best distance that can be used is 30 cm, with a total of 16 correct detections and 10 false detections.

Based on the data in Table 2, the number of correctly detected data at all threshold values is the same, so to ensure a better level of security, the threshold value of 0.3 is the best value that can be applied to all the conditions mentioned.

4 Conclusion

on the research and development of digital key systems, IoT devices connected to web-based applications are effective in building digital key systems. The application of the DeepFace library facilitates system research and development, supports data normalization, and ensures accurate face reading and detection. The best

Table 2: Threshold Testing Result

Lighting	**Threshold**	**Object 3 people with a distance of 30 cm**	
		True	**False**
Bright (520 lux)	0.3	15	0
	0.4	15	0
	0.5	15	0
	0.6	15	0
Dim (82 lux)	0.3	13	2
	0.4	13	2
	0.5	13	2
	0.6	13	2

model combination using high photo resolution is obtained from VGG-Face with OpenCV at 95% while for photos taken with ESP-32 Cam, the best combination is Facenet512 with RetinaFace at 100%İn addition, lighting factors and the distance between the face and the IoT device camera significantly affect face recognition accuracy in the developed system. Based on the analysis, it is known that the number of correctly detected data at all threshold values is the same. To ensure a better level of security, a threshold value of 0.3 is the best choice that can be applied in all the conditions mentioned.

References

[1] E. Saputro and H. Wibawanto, "Rancang bangun pengaman pintu otomatis menggunakan e-ktp berbasis mikrokontroler atmega328," *Jurnal Teknik Elektro*, vol. 8, no. 1, pp. 1–4, 2016.

[2] A. F. Fauzan, L. Novamizanti, and R. Y. N. Fuadah, "Perancangan sistem deteksi wajah untuk presensi kehadiran menggunakan metode lbph (local binary pattern histogram) berbasis android," *eProceedings of Engineering*, vol. 5, no. 3, 2018.

[3] U. D. Rosiani, R. A. Asmara, and N. Laeily, "Penerapan facial landmark point untuk klasifikasi jenis kelamin berdasarkan citra wajah," *Jurnal Informatika Polinema*, vol. 6, no. 1, pp. 55–60, 2019.

[4] R. Soekarta, M. Yusuf, M. F. Hasa, and N. A. Basri, "Implementasi deep learning untuk deteksi jenis obat menggunakan algoritma cnn berbasis website," *JIKA (Jurnal Informatika)*, vol. 7, no. 4, pp. 455–464, 2023.

[5] A. Anhar and R. A. Putra, "Perancangan dan implementasi self-checkout system pada toko ritel menggunakan convolutional neural network (cnn)," *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, vol. 11, no. 2, p. 466, 2023.

[6] J. A. J. Alsyayadeh, A. Aziz, K. X. Chang, A. Z. Hossain, S. G. Herawan *et al.*, "Face recognition system design and implementation using neural networks," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 6, 2022.

[7] R. C. Damale and B. V. Pathak, "Face recognition based attendance system using machine learning algorithms," in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE, 2018, pp. 414–419.

[8] P. A. Nugroho, I. Fenriana, and R. Arijanto, "Implementasi deep learning menggunakan convolutional neural network (cnn) pada ekspresi manusia," *Algor*, vol. 2, no. 1, pp. 12–20, 2020.

[9] I. D. Wijaya, U. Nurhasan, and M. A. Barata, "Implementasi raspberry pi untuk rancang bangun sistem keamanan pintu ruang server dengan pengenalan wajah menggunakan metode triangle face," *Jurnal informatika polinema*, vol. 4, no. 1, pp. 9–9, 2017.

[10] T.-V. Dang, "Smart home management system with face recognition based on arcface model in deep convolutional neural network," *Journal of Robotics and Control (JRC)*, vol. 3, no. 6, pp. 754–761, 2022.

[11] A. Hanuebi, S. R. Sompie, and F. D. Kambey, "Aplikasi pengenalan wajah untuk membuka pintu berbasis raspberry pi," *Jurnal Teknik Informatika*, vol. 14, no. 2, pp. 243–252, 2019.

[12] T.-V. Dang, "Smart home management system with face recognition based on arcface model in deep convolutional neural network," *Journal of Robotics and Control (JRC)*, vol. 3, no. 6, pp. 754–761, 2022.

[13] R. A. Asmara, B. Sayudha, M. Mentari, R. P. P. Budiman, A. N. Handayani, M. Ridwan, and P. P. Arhandi, "Face recognition using arcface and facenet in google cloud platform for attendance system mobile application," in *2022 Annual Technology, Applied Science and Engineering Conference (ATASEC 2022)*. Atlantis Press, 2022, pp. 134–144.

[14] R. A. Asmara, M. Ridwan, A. N. Handayani *et al.*, "Pengembangan sistem face recognition menggunakan cloud service, raspberry pi dan convolutional neural network (cnn)," *Jurnal Informatika Polinema*, vol. 9, no. 1, pp. 95–102, 2022.

[15] M. F. Aditama and M. Haryanti, "Sistem pengenalan dan verifikasi wajah menggunakan transfer learning berbasis raspberry pi," *Jurnal Teknologi Industri*, vol. 12, no. 1, 2023.