OPTIMIZING CNNS FOR FACE RECOGNITION: COMPARING WELL-KNOWN ARCHITECTURES WITH CUSTOM MODELS

[1]Theodorus Adrian Setiawan, [2]Y.B. Dwi Setianto.
[1,2]Program Studi Teknik Informatika Fakultas Ilmu Komputer, Universitas Katolik Soegijapranata
[1]20k10014@student.unika.ac.id, [2]setianto@unika.ac.id

ABSTRACT

This project aims to find which CNN architecture is the best. From the research results, well-known CNN architectures give better accuracy than the custom made one. Face recognition is useful for distinguishing someone's facial data that is already in the database. With this, the system can save someone's facial data so that it can be used later for other purposes. In this research's photo data will be used and there are three types of datasets, 20% of the whole dataset are used as testing. Then, one type of photo is removed from training and used in testing. Thirdly, a new 100 photos, facing forward while using glasses, is used for testing. This will test the system to show the possible accuracy rates as the outcome. CNN Algorithm shows great results in terms of researches on face recognition. Well-known CNN algorithms, VGG16 and AlexNet, tend to give a high accuracy result according to many studies. Thus, this research uses two well-known CNNs as its architecture.

Keywords: CNN Algorithm, face recognition, VGG16, AlexNet

1. INTRODUCTION

1.1. Background

In this increasingly modern era, face recognition is becoming more and more popular in society. Along with technology advancements, many machines around us are now equipped with face recognition. From simple phone unlockers to face recognition in the police department. This implementation makes those works more convenient to use. Even though face recognition has been in many places, its technology tends to be limited by its huge computational needs.

The algorithm used in this research is Convolutional Neural Network (CNN). This algorithm was chosen due to its good record for being used in face recognition. Dubey and Jain [1] conducted a research about automatic face recognition for emotion. In their research, they manage to get 94.8% accuracy on CK+ and 93.7% on the JAFFE dataset. They also mention that they found it superior to other existing methods. On the other side, Irjanto and Surantha [2] conducted research using the CNN Alexnet Method algorithm to create a door locking system. The research obtained an accuracy result of 97.5%. Irjanto and Surantha also mentioned that these results were better than other results.

In this research, researchers want to find whether it is better to use well-known CNN algorithms architectures or make a custom one. Well-known CNN architectures used in this research are VGG16 and AlexNet. VGG16 is chosen after viewing Dubey and Jain [1] research

while AlexNet was chosen based on Irjanto and Surantha [3] research. In their research, they were able to get good results using VGG16 and AlexNet architecture respectively. Custom architecture used in this research is a simple, 9 layer CNN architecture.

1.2. Problem Formulation

In this research, there are several problems that must be resolved so that this research can proceed successfully, namely:

1. Can the system recognize the testing data in the form of photos?
2. Which CNN's architecture is the best between the three?
3. How high can each architecture be in terms of accuracy with 50 epoch?

1.3. Scope

This research aims to find the best CNN architecture from the three proposed architectures. Face recognition will be used to detect and recognize the facial data that has been entered in the system database. The data that is going to be used is in the form of photos. The result sought at the end is the accuracy of each architecture.

1.4. Objective

This research focuses on implementing CNN, both well known architectures and custom design, so that researchers can find the best architecture between the three. CNN algorithm is used due to its history of having high accuracy in face recognition. The desired result is of course the highest possible accuracy so that when a different type of facial data is used in the testing, it can still produce better accuracy.

2. LITERATURE STUDY

In 2020, Irjanto and Surantha [2] conducted research where they used CNN Algorithm Alexnet for home security systems. They mention that the door house locking system is very important and very easy to do. Also with the rise of IoT these days, the IoT based security system has started to become more popular among people. They also mainly mention that facial recognition has started to be implemented in door lock systems with CNN being the easiest yet producing the best accuracy among other algorithms. After conducting research and being tested 20 times, an accuracy of 97.83% was obtained using CNN. This result holds true about CNN giving a high accuracy result as they mentioned before. One of the other ways tried by researchers is OpenCV. In their research, OpenCV produces an accuracy of 95%.

Research conducted several years prior in 2017, focuses on improving CNN algorithm. Researchers mention that face recognition has become a great importance in real world application. This makes them want to improve the existing CNN algorithm. They did this by adding two normalization operations to the first and final convolutional layers. Other than that, they also use the SoftMax classifier. In this face recognition research, researchers changed two kinds of variables, size and batch, and used it in the proposed modified CNN. Their proposed modified CNN algorithm got an accuracy of 94.8% with data size 64x64x3 in top-1 error. On the other hand, for top-5 error, all sizes with channel 3 as its third dimension got the same accuracy of 98.8%[4].

According to Georgia Tech Database in their research, their proposed method has increased the effectiveness of face recognition performance with better results.

Liu also did research about CNN algorithms in 2022[10]. Researchers stated that in this modern era, the potential expansion of deep learning applications has greatly increased. In their research, they were using the traditional nine layer of CNN algorithm. As for the datasets, they used standard Labeled Faces in the Wild (LFW) facial datasets. Their research resulted in an accuracy around 80% after 300 steps of learning. After that even though it is fluctuating, the accuracy rate is stable at more than 90% and gradually increasing while continuously learning.

On the other hand, Htwe Pa Pa Win et al[8] did a research focusing on expanding deep learning to detect faces, extract features and recognize. They also use standard facial datasets which is FEI. They mentioned that the use of standard FEI is so that they can easily compare their effectiveness and results with other researches. Their results show an increased accuracy and reduced time complexity while using CNN algorithm. Here the Deep CNN algorithm that they use got an accuracy of 99.10% when using 9 images taken from the FEI dataset, 4 are randomly chosen for training and the other 5 are used for tests. Their result time also reduced to 583.51 seconds when compared to other studies.

Real time face recognition study was also done in 2019. Here researchers were using CNN algorithm for their real time face recognition. They chose the CNN algorithm so that they can maintain its high accuracy while the system works in real time. Researchers also stated that face recognition itself shows quite a challenge because face is a multidimensional, complex, and meaningful part of the body to visualize someone. In this research, they propose a 3-layer CNN method to solve those problems. As for datasets, they use a dataset from Essex University which consists of 115 subjects in it. They also divide their research into two parts where they use normal CNN and they use pre-trained CNN. In doing so two results were obtained. In terms of accuracy both got very good results where normal CNN got 98.4% and pre-trained CNN got 98.6%. The difference between both methods are not significant but when we see the time used for the research then we can see a huge difference. When using normal CNN training time of 132s was achieved while using pre-trained CNN only needed 66s to finish its training. That was a 200% improvement in training time. Researchers mentioned that this is due to the number of epochs in pre-trained is less[11].

CNN Algorithm was also used in other research in 2020. This research focuses on face recognition to help reduce the CoViD-19 cases during pandemic. With the rising of CoViD-19 in 2020, the usage of medical masks has become a must. Unfortunately, with this sudden change many people still prefer to not use medical masks even though the government has urged people to wear masks. From this phenomenon, researchers want to help identify whether people use masks or not. In this research people's faces will be scanned and those who use or do not use masks will be classified to their corresponding group. Researchers used 800 face data where 200 data were faces without masks and 600 were faces with masks. From this research, an accuracy result of 98% was obtained with LFW datasets[3].

Almabdy and Elrefaei [5] in 2019, conducted a research with various algorithms. Here researchers mention how face recognition has been applied broadly in many daily life systems. Because of those importances, they want to test whether they can improve the performance using pre-trained CNN. The results obtained are also close to the results of other studies. CNN and its various variations got an accuracy of at least 93.3%. Even though in this study CNN is not the best result overall, other results that get higher accuracy used a combination of two or more algorithms. One CNN variation, the Deep CNN, is still able to get an accuracy of 98.95%, which is quite similar to other studies.

Still in 2019, another research was conducted using CNN-PCA method for face recognition attendance purposes. The researcher conducted the research in two separate ways. First the researchers only used the PCA method while in the second one they were using the CNN-PCA method. The researchers also used an increasing number of objects, starting from ten to fifty with an interval of ten. The PCA only experiment resulted in high accuracy but still below the accuracy of the experiments with the CNN algorithm mentioned above. The PCA only with ten objects resulted in an accuracy of 90% and the one using fifty objects resulted in an accuracy of 96%. On the other hand, although CNN-PCA only produced 90% accuracy in the test with ten objects in the test with fifty objects the resulting accuracy increased by 2% to 98% [6].

Other than that, a new challenge appeared concerning the implementation of CNN algorithm. With CNN algorithm size can be pretty big which makes it not compatible with all kinds of devices. This problem makes Sheng Chen et al[7] present a class of CNN algorithms that is called MobileFaceNets. With MobileFaceNets, researchers can cut down the size of the algorithm so that it can be used in more devices. In this research, they train MobileFaceNets with ArcFace in refined MS-Celeb-1M. With this trained MobileFaceNets, a 4MB MobileFaceNets can produce an output with 99.55% accuracy on LFW datasets, which is comparable to other big CNN algorithm methods.

Still with the same problem, Wenting Liu et al[9] found that CNN based face recognition required high computing resources and storage which makes it hard to be implemented in some devices. This is unfortunate because CNN has been very good and precise when used in facial recognition. So in their research, they want to improve facial recognition methods based on lightweight CNN. In doing so they make the Squeeze-and-Excitation (SE) block and propose three improved structures which are the depthwise SE module, the depthwise separable SE module, and the linear SE module. Then they use those methods in several datasets like LFW, AgeDV-30, VGG2-FP and many more. On LFW datasets, their linear SE module got an accuracy of 99.57% while the baseline got 99.47%. The increase of accuracy in their results is not significant but still very high. All methods also resulted in a very high accuracy with depthwise SE + linear SE did the worst with 99.4%.

Judging from the results of the literature above, it can be seen that the CNN algorithm has a very high accuracy for face recognition. From some research, it can be seen that CNN gets pretty big in terms of size. Fortunately, there are some methods to customize CNN and make it more

compatible with more devices[7,9]. On the other hand, face recognition with CNN gets accuracy above 93% [4-5] and often gets accuracy above 97%[2,4,6,9]. This shows that CNN is the best algorithm for face recognition. Therefore, this research will be conducted with CNN algorithm to process the testing photo data.

3. RESEARCH METHODOLOGY

3.1. Research Methodology

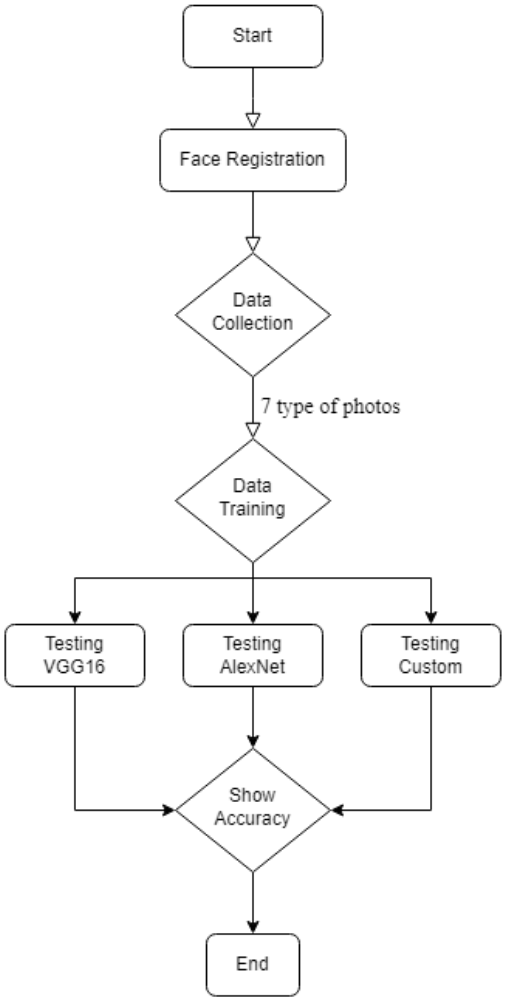

Figure 3.1. Research Methodology

3.2. Face Registration

The homeowner's face data will be entered into the database and will be adjusted to the size of the data to be used. The data entered is data in the form of photos. The photos entered are 700 photos of the owner's face with each 100 photos rotated by approximately 30 degrees in seven kinds of orientations. Those orientations are facing forward, look left, look right, look up, look down, tilt left, and tilt right.

3.3. Dataset Collection

In this research, data will be collected using videos taken with a webcam. A code will be used to turn the camera on and take 700 photos of the homeowner while using face detection to ensure that the photos have face data in them. The photos will contain the homeowner's face with a clearly visible face. The testing data that will be used for the first test is 20% of the whole photo dataset taken evenly and used as testing. For the second test, one type of photo taken from the original data that has been collected for testing while the other six other types used as training data. For the third test, all 700 original data are used as training data while 100 new photos, facing forward with glasses, are used as testing data. The data will be processed with three different types of CNN Algorithm architectures to capture people's facial data.

3.4. Data Training

Training is done using three different CNN Algorithm architectures to recognize faces in photos. It starts with recognizing the photo. The photo fed into the system will be resized to an appropriate size. After that, the system will analyze the photo which involves passing the input image through several layers. These layers consist of convolution, activation, pooling, and fully connected layers that allow the layers to extract features from the input data. The system will then be trained and optimized while performing several training sessions through forward and backward propagation while adjusting several parameters. Next, when the system is good enough in recognizing the training photos, it will be used to start recognizing faces in testing. Finally, testing accuracy can be achieved as the final goal. These steps will be done several times while adjusting epochs and while using the other architectures.

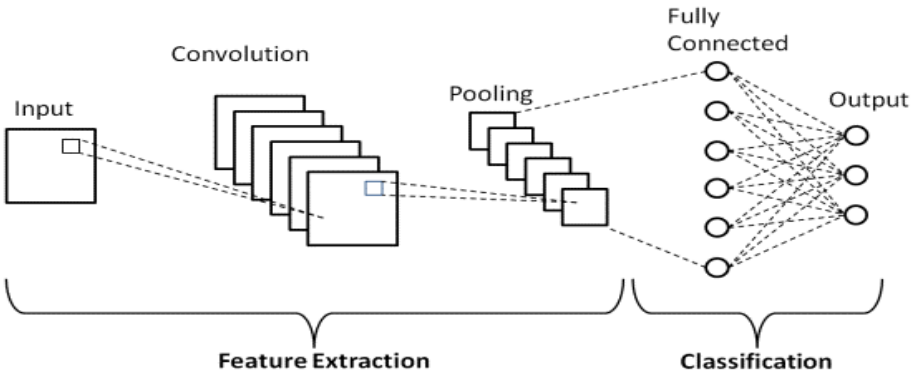

Figure 3.4.1 *CNN Algorithm Layers*

3.5. Testing

This process is designed to check the accuracy rate of a neural network at recognizing images in a test dataset. It works by feeding the test images into the model, seeing what the model predicts, and then comparing those predictions to the correct answers. The goal is to see how often the model gets it right. By calculating the percentage of correct predictions out of the total number of images tested, we can understand how well the model performs on new, unseen data. This helps us determine if the model is reliable for making accurate predictions in real-world situations.

4. IMPLEMENTATION AND RESULTS

4.1. Experiment Setup

In this research, the environment used is Google Colab version 2024-07-22 with the runtime type: python3 T4 GPU. The data used is 3500 image data spread evenly in 5 folders with image conditions as mentioned in section 3.2. On the first test, 20% of the data will be used as testing data and the rest will be used for training data. On the second test, 1 type of the photos will be used as testing photos and will be removed entirely from the training photos. In this case, the right-facing face data. On the third test, all original data will be used as training data. Testing data is a new, 100 facing forward face while using glasses photos.

4.2. Implementation

In this research, there are two versions of the code used. In the first version, the system begins with the creation of DataCollection.py. This code is used to collect data on the faces of people who want to be trained. Next, the photos are trained in Modeling.py. Modeling.py contains the CNN Algorithm which is the main core of this research. Here, the photos will become a dataset to create a model. After adjusting the size and parameters per layer, the training results will be stored in the form of model.keras. In addition, this model.keras also contains important information such as name labels for each face data that has been trained. Finally, using a code called Testing.py, the model results are used as data for this code. This code is used to upload a photo that will be passed through a series of logic gates and can be used to determine a person's face data based on the model.keras with the name label in the code. Unfortunately, the use of this system was not perfectly successful. Even though all the steps went well, the results were not as expected. This happened because when saving the model in Modeling.py and loading the model in Testing.py there is an inequality of variables or units. This causes Testing.py to show only one name label to all tests.

In the second version, the system runs more simply and only requires two files. DataCollection.py like the previous version and CNN_Train_Test.ipynb as the main code. DataCollection.py still functions as in the previous version and is used to collect face data in the form of photos. CNN_Train_Test.ipynb contains the creation of models for the three architectures that will be used, along with training data, and testing data. In this code, the code begins with determining the epoch, batch, learning rate, number of classes, and continues with the size of the input photo for each architecture. There are three sizes that can be used according to the number of architectures to make it easier to change the size. This is done by creating a class to retrieve the dataset and then load the data into another class to be resized according to the architecture needs. Next, create a class to store labels according to the number of classes in the dataset, in this project there are five classes. Then, the next step is model building. In this research, the three architectures used are AlexNet, VGG16, and a custom CNN. This step is followed by creating a class called model three times but each class contains the name of the architecture that can be used. This is done for the same reason as creating the photo input size class, which is to make it easier to replace if needed. In addition, in these steps, the optimizer to be used is also determined, namely Stochastic

Gradient Descent (SGD). The next step is training. In this step, the epoch data that has been determined at the beginning is drawn. Training is done as many times as the number of epochs. The last step is the search for training and testing accuracy. This code runs perfectly and successfully produces the desired results, namely testing accuracy.

From the results of this research, it can be seen that the use of save and load models has its own risks. As happened in version one, if there are different variables or units in the save or load model, it can cause problems in reading. In addition, to correct these differences, more accuracy and understanding of detailed variables and units are required. For the second version, the code is simpler. After modeling for each architecture, dataset load, training, testing to get training and testing accuracy are done in the same code. This behavior reduces the risk of incorrect variables or units because everything is declared in the same file.

4.3. Result

In this first test, the variable used is an arbitrary epoch with batch as the control variable with a value of 10. The data used is 20% of the entire photos picked evenly. In the second test, the data used is one type of photo, namely the right face data, which has been removed from the training data and used in the testing data. The changing variable is epoch with a control variable in the form of batch with a value of 10. The results are shown in tables below. Each table shows screenshots of the results of the executed code showing the training and testing accuracy and the test accuracy results are written below the screenshots on each trial for easy viewing.

Tabel 4.1 First Test Results

Epoch	**VGG16**	**AlexNet**	**Custom**
10	100%	100%	100%
20	100%	100%	100%
30	100%	100%	100%
40	100%	100%	100%

Tabel 4.2 Second Test Results

Epoch	VGG16	AlexNet	Custom
10	52%	77.2%	63.2%
20	59.8%	78.2%	62.4%
30	78.2%	75%	72.8%
40	71.4%	83.4%	63.8%
50	81.6%	81%	74.2%

Tabel 4.3 Third Test Results

Epoch	VGG16	AlexNet	Custom
10	81.4%	99.8%	33%
20	99.8%	99.6%	39.4%
30	99.2%	92.2%	40.6%
40	95.4%	99%	25.8%
50	98.3%	97.7%	38.4%

4.4. Discussion

In the first test, the accuracy results obtained were 100% in all testing results. This is influenced by the dataset which is 20% of the overall data. Data in this form gives rise to perfect accuracy because the testing data and training data are very similar, with only slight differences such as eyes closing or face orientation more tilted or straighter to the camera. Here the epoch used starts from epoch 10 and continues to increase until epoch 40 with an interval of 10 epochs in each trial.

In the second test, the resulting accuracy changed a lot. This was influenced by the dataset where one type of photo consists of 100 items was taken and used for testing data. One type of data is a photo with the face facing right. The results produced varied. At low epochs, AlexNet produced the best testing accuracy and was quite far from other architectures. With 10 epochs, AlexNet managed to get 77.2% accuracy followed by custom CNN at 63.2% and finally VGG16 with 52%. As the epochs increase, the accuracy of each architecture continues to increase as well until at epoch 30 VGG16 starts to become the best architecture among the three architectures used. At this point, VGG16 increased to 78.2% followed by AlexNet at 75% and the lowest Custom CNN with 72.8%. These results changed at epoch 40 where VGG16 dropped to 71.4% while AlexNet increased considerably to 83.4%. But at the last epoch of 50 VGG16 again had the highest accuracy at 81.6, AlexNet at 81% and Custom CNN at 74.2.

In the third test, accuracy results fluctuate very roughly. This is influenced by the dataset where the original 700 photos used as training and testing is using new 100 photos with face facing forward and wearing glasses. In this test, a similar pattern emerged compared to previous results. In low epoch AlexNet got the highest accuracy while VGG16 accuracy is not as good but still quite high. With only 10 epochs, AlexNet got 99.8% accuracy, followed by VGG16 at 81.4% and at the last place is Custom with 33%. Next, with only 20 epoch VGG16 managed to get 99.8% accuracy followed by AlexNet with 99.6% and Custom at 39.4%. This position did not change, even though all accuracy dropped, until epoch 40. Just like the second test, where all architecture accuracy drops and AlexNet once again has the highest accuracy with 99% while VGG16 gets

95.4%. At the end, with 50 epochs VGG16 still got the best accuracy with 98.3%, followed by AlexNet with 97.7% and custom ended up with 38.4%.

From the results of this experiment, with the same batch and epoch, VGG16 produced the highest testing accuracy but it must be trained with a high epoch. On the other hand, AlexNet has relatively high accuracy on lower epochs. AlexNet may be more suitable for devices that do not have much computing power. The creation of Custom CNN is less recommended because to create and determine the variation of each layer requires a deeper understanding in its creation. Especially in this research where the custom architecture is using smaller photo sizes. With this decision, the architecture is harder to collect details which make it get horrible accuracy in the third test. Other than that, the results produced cannot be guaranteed to be better than well-known architectures. In this research, Custom CNN produces the worst results, but if other researchers have a better understanding of Custom CNN creation, it can produce better results than well-known architectures such as VGG16 and AlexNet.

5. CONCLUSION

In conclusion, CNN algorithm has successfully recognized the faces on the test dataset. With similar training and testing dataset in the first test, all architectures did not have any problem and managed to get perfect accuracy. For the second test, since the testing data is quite extreme, the results still show quite good results. On the other hand, in the third test, all well-known architecture got above 80% accuracy, while the custom barely get 40% accuracy. The best architecture in this study is VGG16, which got the best result, although it is not far from AlexNet. For custom CNN, it is not recommended unless the researcher has a very good understanding of CNN modeling and the need of the dataset. During the research, custom CNN never got the best test accuracy results. On the other hand, AlexNet got good accuracy results at low epochs with quite far results compared to the other two architectures. For the final accuracy with epoch 50 in dataset number 2, VGG16 got 81.6% accuracy, followed by AlexNet with 81%, and custom CNN got 74.2% accuracy. With dataset number 3, VGG16 got 98.3% accuracy, followed by AlexNet with 97.7%, and custom CNN got 38.4% accuracy. Future research is encouraged to try changing the number of batches, the learning rate, or even the type of photos to be used for testing. If possible, increasing the number of epochs can get higher or more diverse accuracy results. A combination of these changes may result in more varied findings. This research was unable to explore those variables due to physical hardware limitations and the free Google collab software which has a limit on the number of computing units.

REFERENCES

[1] A. K. Dubey, and V. Jain. "Automatic Facial Recognition Using VGG16 Based Transfer Learning Model." Journal of Information and Optimization Sciences 41 (7): 1589–96. 2020, doi:10.1080/02522667.2020.1809126. Diakses dari https://sci-hub.scrongyao.com/10.1080/02522667.2020.1809126

[2] N. S. Irjanto and N. Surantha, “Home Security System with Face Recognition based on Convolutional Neural Network,” IJACSA, vol. 11, no. 11, 2020, doi: 10.14569/IJACSA.2020.0111152. Diakses dari https://dx.doi.org/10.14569/IJACSA.2020.0111152

[3] I. Q. Mundial, M. S. Ul Hassan, M. I. Tiwana, W. S. Qureshi, and E. Alanazi, “Towards Facial Recognition Problem in COVID-19 Pandemic,” in 2020 4rd International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM), Medan, Indonesia: IEEE, Sep. 2020, pp. 210–214. doi: 10.1109/ELTICOM50775.2020.9230504. Diakses dari https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9230504

[4] M. Coskun, A. Ucar, O. Yildirim, and Y. Demir, “Face recognition based on convolutional neural network,” in 2017 International Conference on Modern Electrical and Energy Systems (MEES), Kremenchuk: IEEE, Nov. 2017, pp. 376–379. doi: 10.1109/MEES.2017.8248937. Diakses dari https://sci.bban.top//pdf//10.1109//MEES.2017.8248937.pdf

[5] S. Almabdy and L. Elrefaei, “Deep Convolutional Neural Network-Based Approaches for Face Recognition,” Applied Sciences, vol. 9, no. 20, p. 4397, Oct. 2019, doi: 10.3390/app9204397. Diakses dari https://www.mdpi.com/2076-3417/9/20/4397/pdf?version=1571881432

[6] E. Winarno, I. Husni Al Amin, H. Februariyanti, P. W. Adi, W. Hadikurniawati, and M. T. Anwar, “Attendance System Based on Face Recognition System Using CNN-PCA Method and Real-time Camera,” in 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia: IEEE, Dec. 2019, pp. 301–304. doi: 10.1109/ISRITI48646.2019.9034596. Diakses dari https://sci-hub.se/10.1109/isriti48646.2019.9034596

[7] S. Chen, Y. Liu, X. Gao, and Z. Han, “MobileFaceNets: Efficient CNNs for Accurate Real-Time Face Verification on Mobile Devices,” in Biometric Recognition, J. Zhou, Y. Wang, Z. Sun, Z. Jia, J. Feng, S. Shan, K. Ubul, and Z. Guo, Eds., in Lecture Notes in Computer Science, vol. 10996. Cham: Springer International Publishing, 2018, pp. 428–438. doi: 10.1007/978-3-319-97909-0_46. Diakses dari https://arxiv.org/pdf/1804.07573

[8] University of Computer Studies, Hpa-an, H. P. Pa Win, P. T. Thu Khine, and K. Nwe Ni Tun, “Face Recognition System based on Convolution Neural Networks,” IJIGSP, vol. 13, no. 6, pp. 23–29, Dec. 2021, doi: 10.5815/ijigsp.2021.06.03. Diakses dari https://www.mecs-press.org/ijigsp/ijigsp-v13-n6/IJIGSP-V13-N6-3.pdf

[9] W. Liu, L. Zhou, and J. Chen, “Face Recognition Based on Lightweight Convolutional Neural Networks,” Information, vol. 12, no. 5, p. 191, Apr. 2021, doi: 10.3390/info12050191. Diakses dari https://www.mdpi.com/2078-2489/12/5/191/pdf?version=1619594131

[10] R. Liu, “Face Recognition Based on Convolutional Neural Networks,” HSET, vol. 16, pp. 32–39, Nov. 2022, doi: 10.54097/hset.v16i.2225. Diakses dari https://drpress.org/ojs/index.php/HSET/article/view/2225/2129

[11] K. Yadav, M. Kumar, S. Kumar, and A. K. Tiwari, “Real Time Face Recognition based on Convolution Neural Network,” SSRN Journal, 2019, doi: 10.2139/ssrn.3350324. Diakses dari https://sci.bban.top//pdf//10.2139//ssrn.3350324.pdf