Evolution in Electrical and Electronic Engineering Vol. 2 No. 2 (2021) 214-223

EEEE

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/eeee
e-ISSN : 2756-8458

Face Recognition using Deep CNN Models

Teng Kai Lum[1], Munirah Ab Rahman[1]*

[1] Department of Electronic Engineering, Faculty of Electrical and Electronic Engineering,
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author Designation

DOI: https://doi.org/10.30880/eeee.2021.02.02.026
Received 22 July 2021; Accepted 26 August 2021; Available online 30 October 2021

Abstract: This paper presents the use of a deep learning convolutional neural network to recognize different human faces images. This study used three pre-trained deep CNN models to solve a three-class classification problem, which are three different person faces images. This work also aims to provide a quantitative assessment of the classifier's dependability based on performance measures. In this preliminary study, we used 120 human faces images to train deep CNN models. Using the results from three different deep CNN models (AlexNet, DenseNet201 and GoogLeNet), this study assesses the performance of the trained classifier and finds that it performs quite well in the prediction of validation data. The best performance result is AlexNet. The parameter metrics of the batch size, the number of epochs, validation frequency and learning rate are using to train CNN models of AlexNet are 10, 25, 35 and 0.001, respectively. Therefore, the mean accuracy, training accuracy, validation accuracy and error rate using AlexNet are 90.48 percent, 87.00 percent, 90.48 percent and 0.52 percent, respectively. It is concluded that the trained model is capable of classifying three different person faces well. In the future, more data could be utilized in the training network to improve the accuracy of the models. Another possible approach would be to integrate human faces images from different imaging modalities in the training to increase the variety of datasets on which a deep learning model can develop.

Keywords: Face Recognition, CNN, Deep Learning, Image Classification

1. Introduction

Face recognition is a technology that uses a picture, video, or other audiovisual feature of a person's face to identify or authenticate a person [1]. This identifier is typically used to get access to a programme, system, or service. It is a biometric identification approach that employs body measurements, in this case the face and head, to validate an identity of a person through their facial biometric pattern. Biometrics has gained popularity as a result of the need for trustworthy personal identification in computerized access control. Fingerprints [2], speech [3], signature dynamics [4], and face recognition [5] are among the biometrics being studied. Identity verification products have surpassed $100 million

Corresponding author: munira@uthm.edu.my

in sales [6]. Face recognition has the advantage of being a non-intrusive, passive method of confirming personal identity.

In terms of algorithms, the convolution layer of CNN share parameters. This has the advantage of lowering memory needs and lowering the number of parameters to train. As a result, the performance of the algorithm has been enhanced. Other machine learning techniques, on the other hand, need us to conduct preprocessing or feature extraction on the images. When utilizing deep CNN for image processing, however, we rarely need to perform these procedures. Other machine learning algorithms are unable to achieve this. In order to obtain high accuracy, more research on effective and time-saving face recognition is required. First, the transformation of labelled data into a usable format prior to the training phase is commonly required for the success of machine learning systems. This makes it possible for the chosen machine learning algorithm to understand the data. Pre-processing a large amount of data, on the other hand, is expensive because it frequently needs a lot of human effort. Besides, the qualities of the data used as inputs have a significant impact on the performance of present machine learning algorithms. Furthermore, a solid feature representation is required for many real-world machine learning applications. The deep learning method, which comprises of multi-layers with many parameters in order to categorise objects with high accuracy, frequently require a long training period. As a result, a different strategy such as max-pooling is employed to reduce the size of feature maps in order to reduce computation complexity and, as a result, training time. The use of pre-trained deep CNN models, AlexNet, DenseNet201 and GoogLeNet, for quick classification of face recognition using 40 images for each person is presented in this paper.

2. Methods

The method and process used in this project to do the performance comparison of face recognition with AlexNet, DenseNet201 and GoogLeNet CNN models. All of the work is done with the MATLAB R2021a software. Figure 1 depicts the methodology workflow.

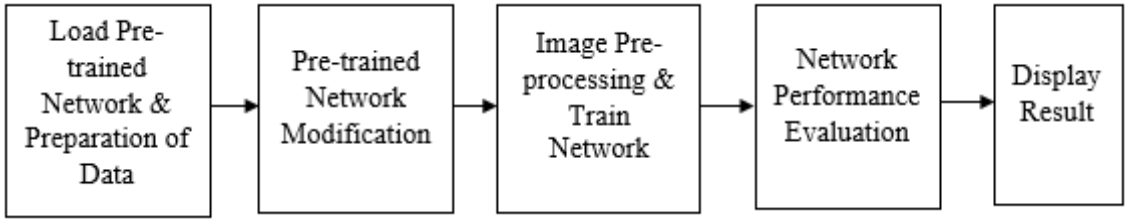


Figure 1: General workflow of the tasks in this project

2.1 Face detection using Haar cascades

An accurate numerical definition in a face detection algorithm must be used, so that human faces are set apart from other objects in a given image. With weak classifiers, such a committee can be created to form a strong classifier by employing a voting process. To construct classifiers, the Viola-Jones algorithm uses Haar-like rectangle characteristics. Between the image and some Haar-like pattern, a Haar-like rectangle function is a scalar product. This Haar-like rectangle box will be used to do face detection and classified it as different part on facial features.

2.2 Load pre-trained network and preparation of data

The image capture is done by using MATLAB software with webcam and snapshot syntax. The image size is set to 227x227x3 pixels. After the human face images had been collected, deep CNN model is used to do the training for all data that had been collected. The entire path to the folder where the images were saved using the MATLAB syntax, *fullfile*. The MATLAB syntax of *imageDatastore* used to load human faces images. These images were named after the folders which were: Andy, Kai Lum, and YongHong. These folders are stored as object property data in MATLAB file format after these folders were called into the MATLAB software platform. A data store allows big datasets to be

stored even if they do not fit in memory, and it allows for fast image batch reading during deep CNN training. Following that, each classes of human faces images were separated into training and validation data sets. This research employed 70% of the images picked at random of each folder for training and 30% for validation. The image files were partitioned into new datastores using the MATLAB syntax of *splitEachLabel*. Then, AlexNet, DenseNet201 and GoogLeNet deep CNN models were used in MATLAB, these pre-trained CNN models were invoked. Before the pre-trained deep CNN models could be accessed, a Deep Learning Toolbox had to be installed. The *analyzeNetwork* MATLAB syntax was used to illustrate the AlexNet, DenseNet201 and GoogLeNet network architecture and gather information from each layer. The *inputSize* MATLAB syntax was used to resize the input images to the desire input image size of pre-trained deep CNN models.

2.3 Pre-trained Network Modification

The layer graph was extracted from the pre-trained AlexNet, DenseNet201 and GoogLeNet networks for training on the studied human face images using the *lgraph* MATLAB syntax. This used to adapt the network for the unique goal of recognize human faces. For this three deep CNN models, the retrieved layers were replaced with new layers which were, Fully Connected Layer, Softmax Layer and Classification Output Layer, that were tailored to the new datasets.

2.4 Network pre-processing and train network

The network was pre-trained. The MATLAB syntax of augmentedImageDatastore used to increase the variety and amount of images used in the training process. This study used two augmentation strategies (for training and testing), which is a well-known strategy for avoiding overfitting. Each image was randomly augmented using the techniques outlined above, with these images substituting the original images in the training, total 84 images from three different label folders. It is critical to define hyperparameter values during training, the performance of deep CNN models were strongly dependent on them. In this training process, the batch size was set to 10, validation frequency was set to 35 iterations and the learn rate was set to 0.001. The number of epochs was set to 6 for AlexNet, 25 for DenseNet201 and 2 for GoogLeNet. The number of epochs was depending on the training accuracy and validation accuracy to whole training progress with 100%, hence the number of epochs can be observed to avoid over training. The modified deep CNN models was trained ten times in total for each deep CNN models. The top three runs results of training accuracy and validation accuracy were then utilized to evaluate and analyze the performance of deep CNN model performance.

2.5 Network performance evaluation

After the training sessions, the performance of trained deep CNN models were evaluated using validation data from three of the best-trained models. In this paper, accuracy, error rate, training accuracy and validation accuracy are all used to evaluate the deep CNN models prediction capability. The ideal number for the error rate is 0%, while the worst value is 100%. Meanwhile, the highest accuracy value is 100%, while the lowest is 0%.

2.6 Display result

At the end of the training deep CNN models sessions, the predicted image and actual image of human faces images were chosen from the validation dataset would be provided at the conclusion.

3. Results and Discussion

AlexNet, DenseNet201 and GoogLeNet were trained to recognize specific person using different datasets of images in the MATLAB. Figure 2, Figure 3 and Figure 4 illustrate the schematic of the pre-trained AlexNet, DenseNet201 and GoogLeNet layer architecture interactively displayed with the MATLAB syntax of *analyzeNetwork* respectively. In order to extract significant information, the architecture displays the different layers and arrangements of convolutional, max pooling, softmax,

fully connected, and rectified linear activation (ReLU) layers. The MATLAB syntax used for AlexNet, DenseNet201 and GoogLeNet to train the network to recognize images were *g=alexnet*, *g=densenet201* and *g=googlenet*.

Figure 5 depicts the performance of AlexNet network in image label classification as the training process progressed. The accuracy of the trained model using validation data, as well as the training time utilizing are also presented in the figure. The simulation is repeated ten times in this work. On average, it takes 306 seconds to train an AlexNet network. The graph in Figure 5 depicts loss as a function of iteration, illustrating that the loss decreases as the training process progresses.

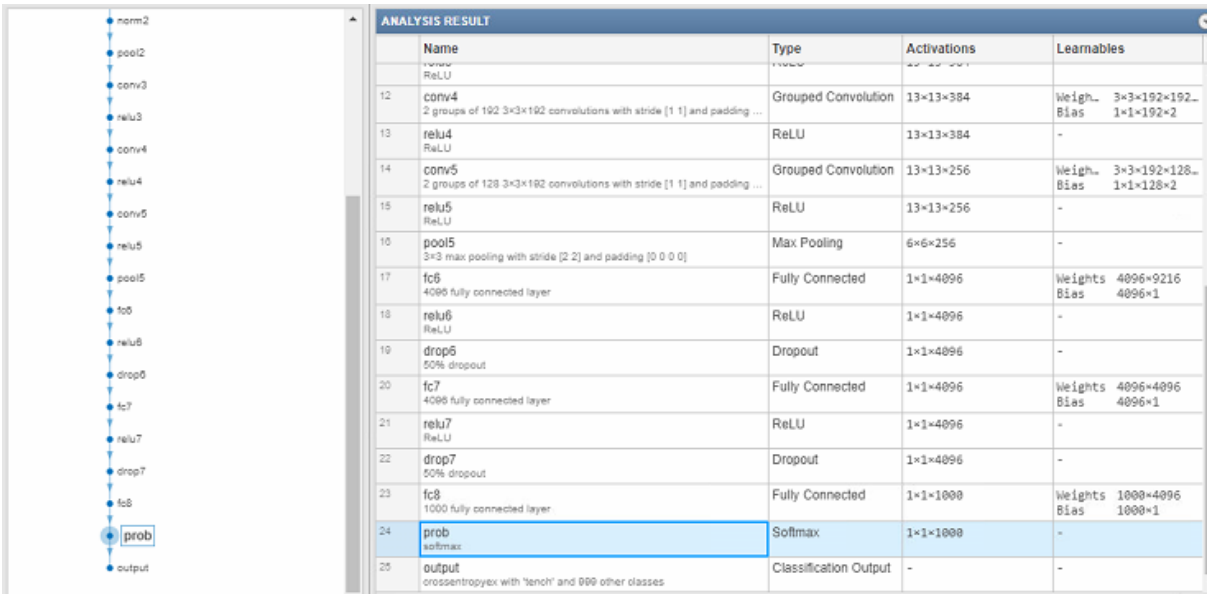


Figure 2: Transfer learning layers in AlexNet

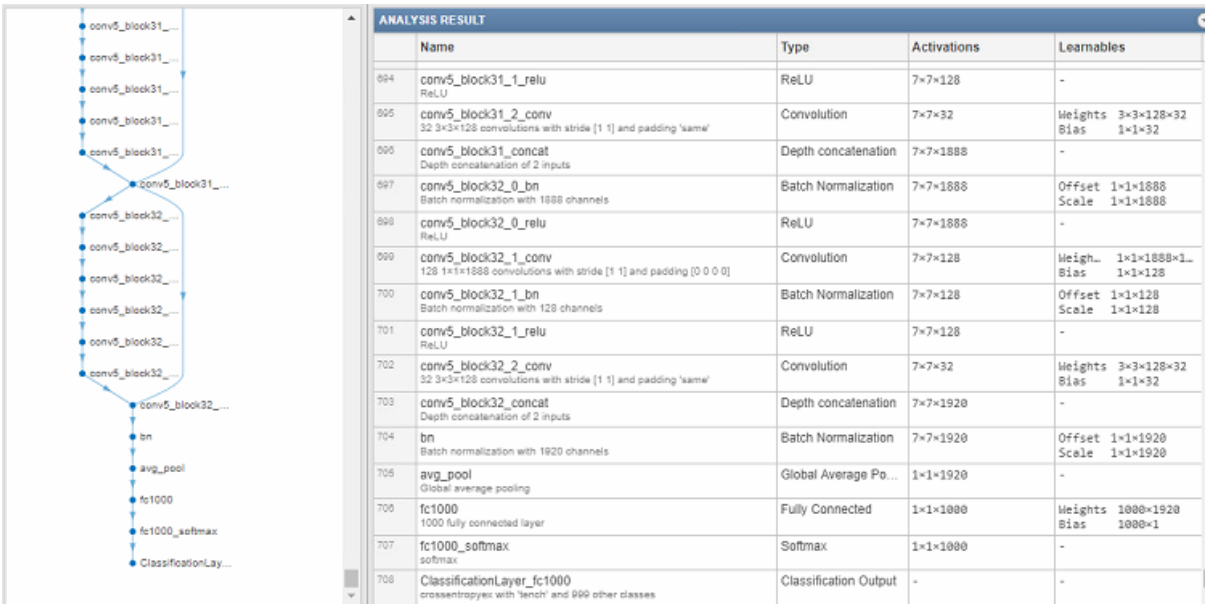
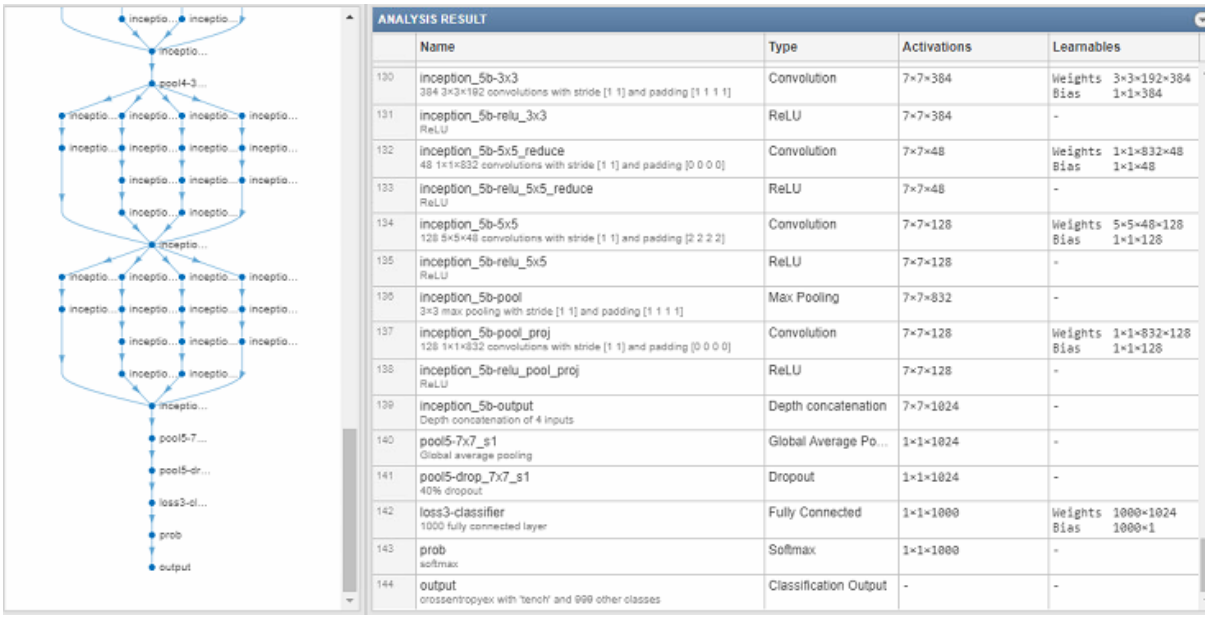


Figure 3: Transfer learning layers in DenseNet201



ANALYSIS RESULT

	Name	Type	Activations	Learnables
130	inception_5b-3x3 384 3×3×192 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	7×7×384	Weights 3×3×192×384 Bias 1×1×384
131	inception_5b-relu_3x3 ReLU	ReLU	7×7×384	-
132	inception_5b-5x5_reduce 48 1×1×832 convolutions with stride [1 1] and padding [0 0 0 0]	Convolution	7×7×48	Weights 1×1×832×48 Bias 1×1×48
133	inception_5b-relu_5x5_reduce ReLU	ReLU	7×7×48	-
134	inception_5b-5x5 128 5×5×48 convolutions with stride [1 1] and padding [2 2 2 2]	Convolution	7×7×128	Weights 5×5×48×128 Bias 1×1×128
135	inception_5b-relu_5x5 ReLU	ReLU	7×7×128	-
136	inception_5b-pool 3×3 max pooling with stride [1 1] and padding [1 1 1 1]	Max Pooling	7×7×832	-
137	inception_5b-pool_proj 128 1×1×832 convolutions with stride [1 1] and padding [0 0 0 0]	Convolution	7×7×128	Weights 1×1×832×128 Bias 1×1×128
138	inception_5b-relu_pool_proj ReLU	ReLU	7×7×128	-
139	inception_5b-output Depth concatenation of 4 inputs	Depth concatenation	7×7×1024	-
140	pool5-7x7_s1 Global average pooling	Global Average Po...	1×1×1024	-
141	pool5-drop_7x7_s1 40% dropout	Dropout	1×1×1024	-
142	loss3-classifier 1000 fully connected layer	Fully Connected	1×1×1000	Weights 1000×1024 Bias 1000×1
143	prob softmax	Softmax	1×1×1000	-
144	output crossentropyex with 'tench' and 999 other classes	Classification Output	-	-

Figure 4: Transfer learning layers in GoogLeNet

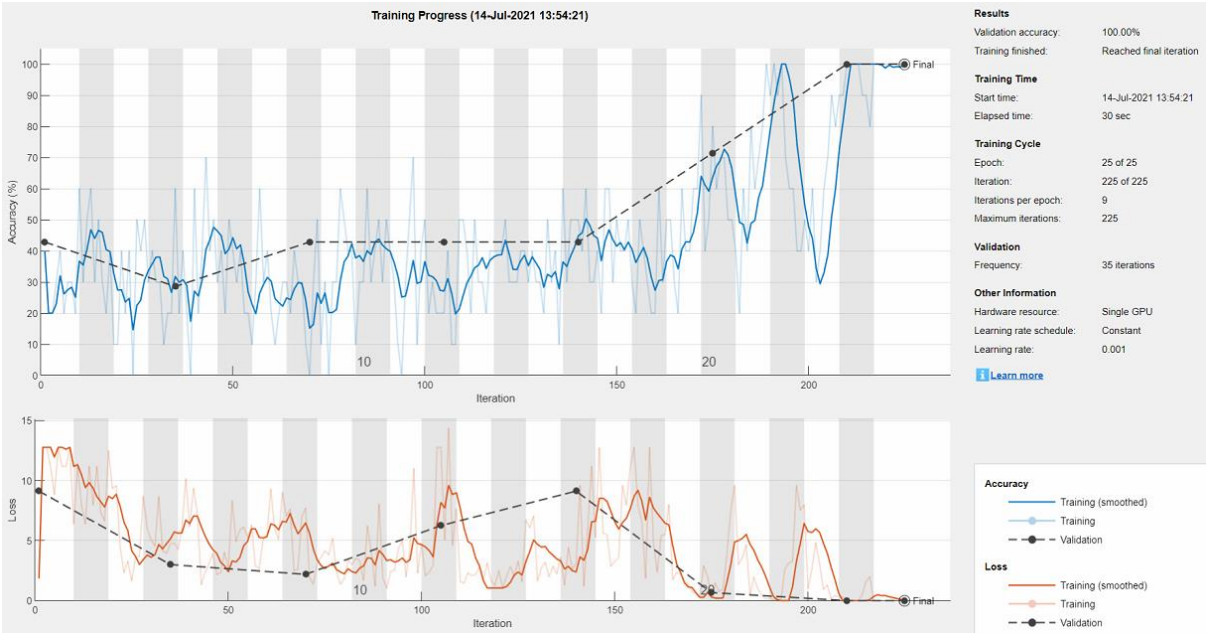


Figure 5: Plot of training progress for human faces images recognition using AlexNet

As the training process progressed, Figure 6 displays the performance of the DenseNet201 network in picture label categorization. The figure also shows the accuracy of the trained model when using validation data, as well as the training time. In this work, the simulation is performed ten times. A DenseNet201 network takes 679 seconds to train on average.

As the training process progressed, Figure 7 displays the performance of the GoogLeNet network in image label classification. The figure also shows the accuracy of the trained model when using validation data, as well as the training time. In this work, the simulation is performed ten times. A GoogLeNet network takes 421 seconds to train on average.

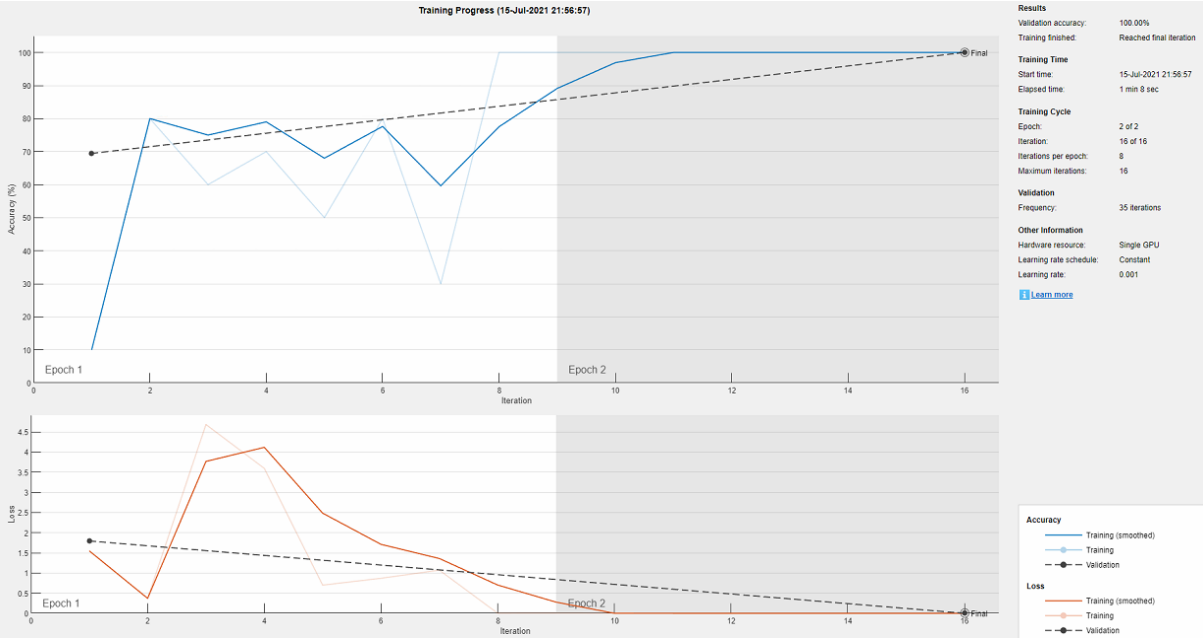


Figure 6: Plot of training progress for human faces images recognition using DenseNet201

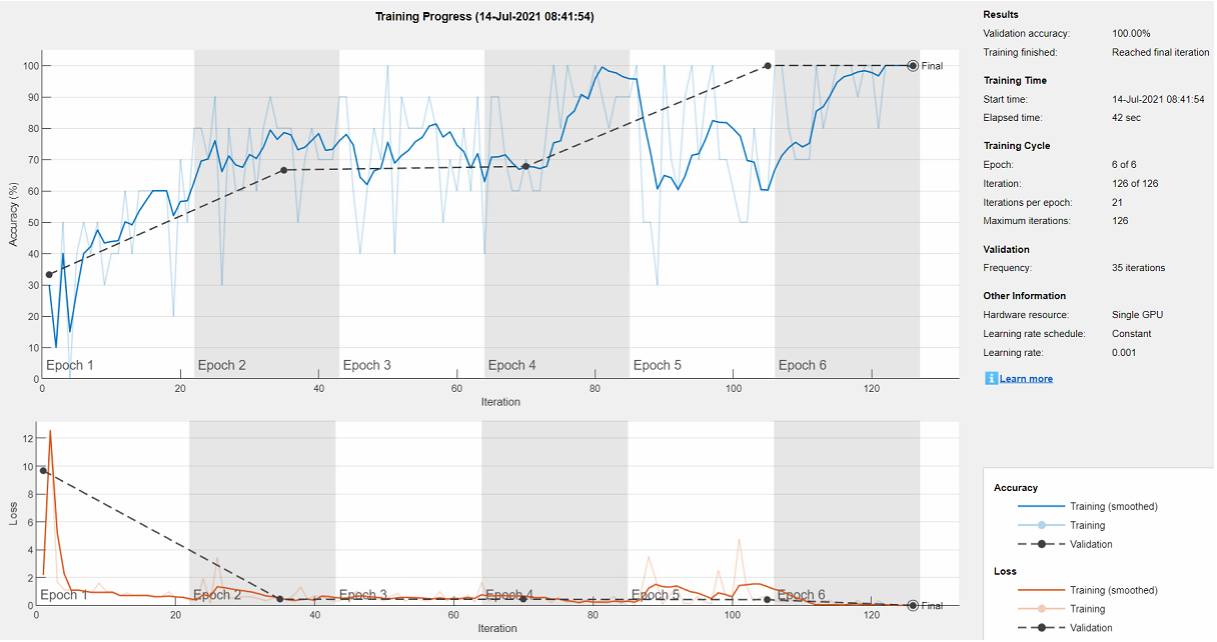


Figure 7: Plot of training progress for human faces images recognition using GoogLeNet

Table 1, Table 2 and Table 3 show the accuracy of the training and validation of ten different training sessions using AlexNet, DenseNet201 and GoogLeNet respectively. These training sessions were done by LENOVO Legion Y540 with 8GB RAM of DDR4 2666 MHz and ninth generation Intel Core i7 processor. The training time between AlexNet, DenseNet201 and GoogLeNet were different. DenseNet took the most time, GoogLeNet took the second most time and AlexNet took the least amount of time to train the images. AlexNet has eight layers with learnable parameters, hence it took least of the amount of time to train the network. DenseNet201 has 201 layers deep used to do training sessions, so it took the most of the time. GoogLeNet architecture consists of 22 layers which the amount of layers not more than DenseNet and less than AlexNet, therefore, it took the training time between them.

Table 1: The accuracy of training and validation for 10 different training sessions of AlexNet

Training session index, *i*	Training accuracy (%)	Validation accuracy (%)
1	100.00	100.00
2	100.00	100.00
3	40.00	71.43
4	100.00	100.00
5	100.00	100.00
6	100.00	100.00
7	50.00	47.62
8	80.00	85.71
9	100.00	100.00
10	100.00	100.00
Average	**87.00**	**90.48**

Table 2: The accuracy of training and validation for 10 different training sessions of DenseNet201

Training session index, *ii*	Training accuracy (%)	Validation accuracy (%)
1	100.00	100.00
2	80.00	86.11
3	70.00	91.67
4	100.00	97.22
5	100.00	100.00
6	60.00	33.33
7	90.00	97.22
8	100.00	100.00
9	80.00	94.44
10	60.00	58.33
Average	**84.00**	**85.83**

Table 3: The accuracy of training and validation for 10 different training sessions of GoogLeNet

Training session index, *iii*	Training accuracy (%)	Validation accuracy (%)
1	100.00	100.00
2	70.00	65.56
3	20.00	33.33
4	98.89	98.89
5	70.00	66.67
6	40.00	66.67
7	100.00	92.22
8	90.00	100.00
9	100.00	100.00
10	30.00	33.33
Average	**71.89**	**75.67**

Table 1, Table 2 and Table 3 show that the results, in terms of both training and testing accuracy, among the ten times training sessions. From the comparison, AlexNet has the best outcome, in term of both training and testing accuracy, compared with others. Table 4 shows the average performance metrics of trained deep CNN models that were chosen for further analysis.

Table 4: Performance metrics based on the average values of training sessions

	Deep CNN Models		
Performance metrics (Average)	**AlexNet**	**DenseNet201**	**GoogLeNet**
Error rate	0.52%	14.17%	24.33%
Accuracy	90.48%	85.83%	75.67%
Training accuracy	87.00%	84.00%	71.89%
Validation accuracy	90.48%	85.83%	75.67%

Figure 8 shows the confusion matrix of validation images using the trained model from *i* = 1, *ii*=1 and *iii*=1.

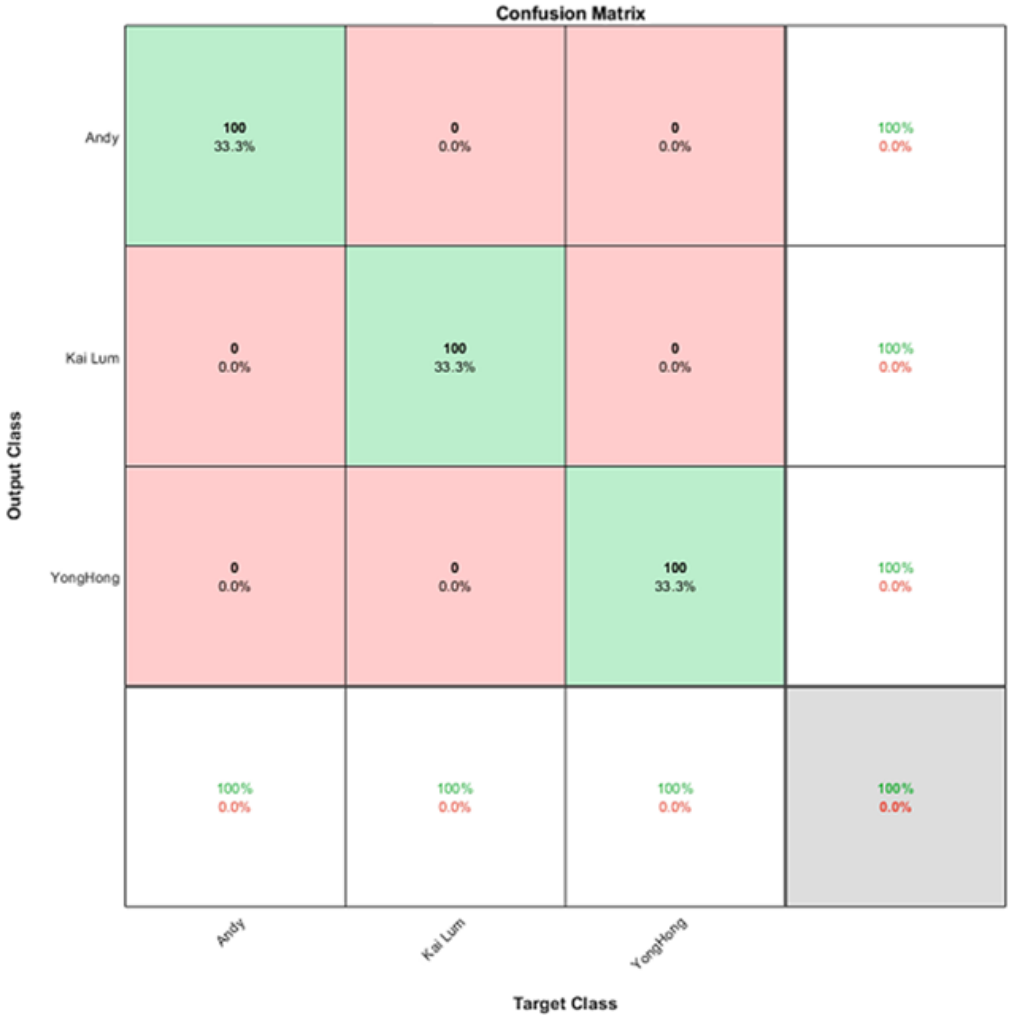


Figure 8: Confusion matrix on validation dataset from training session of *i* = 1, *ii*=1 and *iii*=1

Figure 9 shows the testing of random human faces images recognize results of different person images for AlexNet, DenseNet201 and GoogLeNet for one training dataset image, *i* = 1, *ii*=1 and *iii*=1.

Figure 9: Example of predicted image and actual image using training deep CNN models

4. Conclusion

From the results, it can be concluded that the accuracy of face recognition using AlexNet convolutional neural network is the highest, compare to DenseNet201 and GoogLeNet. The viability of utilizing a modified pre-trained AlexNet, DenseNet201 and GoogLeNet to recognize human faces images dataset was demonstrated in this research. In majority of the validation sessions, this research discovered higher training accuracy and validation accuracy, as well as a lower error rate between these three deep CNN training models. This study suggested that future improvements to the accuracy of deep CNN model classification can be made. A softmax classification layer, a fully connected layer and several convolution layers make up the deep CNN models. This model is used to optimize the model parameters by training the input face image data set. Lastly, the performance metric demonstrates that deep CNN model, AlexNet, is an excellent choice for face recognition and that it may be enhanced further.

Adaptive cross-checking techniques, such as class activation mapping (CAM), could also be utilized for this. CAM can be used to see if a certain part of an input image of human face "confused" the convolutional neural network, resulting in inaccurate prediction. The capacity to detect incorrect predictions in the foundation of deep CNN models would allow for the creation of a network with improved classification performance. This will need to increase the amount of dataset being included in the training progress. This could involve the usage of airport or images in the future to boost picture diversity and dataset size.

Acknowledgement

The authors would like to thank the Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia for its support.

References

[1] Shams, N., Hosseini, I., Sadri, M.S. and Azarnasab, E., 2006, October. Low cost FPGA-based highly accurate face recognition system using combined wavelets with subspace methods. In 2006 International Conference on Image Processing (pp. 2077-2080). IEEE

[2] J.L. Blue, G.T. Candela, P.J. Grother, R. Chellappa, and C.L. Wilson. Evaluation of pattern classifiers for fingerprint and OCR applications. Pattern Recognition, 27(4):485–501, April 1994

[3] D. K. Burton. Text-dependent speaker verification using vector quantization source coding. IEEE Transactions on Acoustics, Speech, and Signal Processing, 35(2):133,

1987

[4] Y.Y. Qi and B.R. Hunt. Signature verification using global and grid features. Pattern Recognition, 27(12):1621–1629, December 1994

[5] R. Chellappa, C.L. Wilson, and S. Sirohey. Human and machine recognition of faces: A survey. Proceedings of the IEEE, 83(5):705–740, 1995

[6] B. Miller. Vital signs of identity. IEEE Spectrum, pages 22–30, February 1994