International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-11, September 2020

Transfer learning using AlexNet Convolutional Neural Network for Face Recognition

Ridza Azri Bin Ramlee, Yvonne Yap, Siva Kumar Subramaniam, Mohamad Harris Misran, Asem Khmag

Abstract*: *This research is aimed to achieve high-precision accuracy and for face recognition system. Convolution Neural Network is one of the Deep Learning approaches and has demonstrated excellent performance in many fields, including image recognition of a large amount of training data (such as ImageNet). In fact, hardware limitations and insufficient training data-sets are the challenges of getting high performance. Therefore, in this work the Deep Transfer Learning method using AlexNet pre-trained CNN is proposed to improve the performance of the face-recognition system even for a smaller number of images. The transfer learning method is used to fine-tuning on the last layer of AlexNet CNN model for new classification tasks. The data augmentation (DA) technique also proposed to minimize the over-fitting problem during Deep transfer learning training and to improve accuracy. The results proved the improvement in over-fitting and in performance after using the data augmentation technique. All the experiments were tested on UTeMFD, GTFD, and CASIA-Face V5 small data-sets. As a result, the proposed system achieved a high accuracy as 100% on UTeMFD, 96.67% on GTFD, and 95.60% on CASIA-Face V5 in less than 0.05 seconds of recognition time.

Keywords* : *CNN; Data Augmentation; Face Recognition; Transfer Learning.

I. INTRODUCTION

Face recognition technology has a long history, but related studies are still popular today [1]. Face recognition is a biometric technology in the field of computer vision, aimed to identifying faces in still image or videos. It is widely used in the fields of biometrics, payment, business, advertising, health care. Moreover, it has been used extensively as security systems and crime identification, for access to the restricted areas [2]. Face recognition is also one of the most challenging biometric methods. The performance of it has been improved steadily from conventional to deep learning methods [3].

Revised Manuscript Received on September 30, 2020.

* Correspondence Author

Ridza Azri Bin Ramlee*, FKEKK, UTeM, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia. Email: ridza@utem.edu.my.

Lim Yu Guan, Centre for Telecommunication Research & Innovation, UTeM, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia.

Siva Kumar Subramaniam, Centre for Telecommunication Research & Innovation, UTeM, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia.

Mohamad Harris Misran, Centre for Telecommunication Research & Innovation, UTeM, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia.

Asem Khmag,Department of Computer Systems Engineering University of Zawia, Libya.

M. D. Kelly et al. (1971), had conducted the research on the face recognition and the first face recognition algorithm was develop in early 1970s [4]. In the late 1980s, the development of computer technology and optical imaging technology was improved and the real entry into the application phase of face recognition in the late 1990s. In early research on face recognition, the researcher mainly focuses on methods called geometry methods used to match simple features with image processing techniques [3]. Later, the holistic method such as principal component analysis (PCA) and linear discriminant analysis (LDA) appeared and become popular. Then, the feature-based method developed for matching all the local features across face image. As time passed, a holistic and feature-based method keeps development and then combined into hybrid methods. Hybrid methods remained the state-of-the-art until recently when deep learning emerged as a leading approach to most applications of computer vision, including face recognition. Deep Learning is a machine learning technique that gets attention lately because it can achieve high accuracy when trained with large amounts of data [5]. In addition, Convolutional Neural Networks (CNNs) is one of the deep learning algorithms and it is the most popular neural network model used for image classification problems after the first won in the ILSVRC competition in the years 2012 [6]. The ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) was held annually starting in 2010, after the birth of the large-scale dataset, ImageNet. Recently, CNN models are deeper and deeper and achieved a good result in the ILSVRC competition. But the main problems are large scales labelled data might be difficult to collect and we do not have powerful hardware for train a deep neural network. Therefore, training deep neural network is time-consuming. As a result, a CNN based deep transfer learning for face recognition using small datasets is proposed. Transfer learning is a popular deep learning approach where the knowledge gained from a related task is transferred to a new task [7]. Compared with training deep neural networks from scratch, this proposed method can reduce training time. In addition, high accuracy can be achieved even with small data sets.

II. METHODOLOGY

The evaluation of face recognition system using transfer learning techniques with pre-trained CNN model will be presented in this work.

A. Software

The proposed system, of face recognition algorithm is implemented in MATLAB R2019b environment.

Many toolbox functions are available as source. Deep Learning Toolbox, Deep Learning Toolbox Model for AlexNet Network, Computer Vision Toolbox, and Parallel Computing Toolbox™ are the toolboxes used in this proposed system.

B. Databases

Three different face data-sets are used for training, validation, and testing on the proposed method. There are Georgia Tech face database (GTFD), CASIA Face Image Database Version 5.0 (FASIA-FaceV5), and (UTeMFD) from UTeM students. The UTeMFD data-set, consists of 420 images in total, from 14 student's faces from University, Teknikal Malaysia Melaka (UTeM) collected as a sample in which each student had 30 images. The images are captured using a USB 2.0 HD IR UVC Webcam in one session, and the resolution of the images are 1280x720 pixels. For GTFD [10], it was constructed by Georgia Institute of Technology, which contains 750 images in total, which taken from 50 individuals. The resolution of the images is 640x480 pixels and the average size of the faces in these images is 150x150 pixels. The last database is CASIA-FaceV5 [11] was collected by the Chinese Academy of Sciences' Institute of Automation (CASIA) which contains 2,500 color facial images of 500 subjects. The images are captured using a Logitech USB camera in one session. Figure 2 until 4, show the sample face images of UTeMFD, GTFD, and CASIA-FaceV5 data-sets. Detail information of three data-sets shown in Table 1.

Table1: Details Information of the Datasets.

Database	pixels	No images	Image per subject	Subjects	Format
UTeMFD	1280x720	420	30	14	JPG
GTFD	640x480	750	15	50	JPEG
CASIA-Face V5	640x480	2500	5	500	BMP

Figure 2: Sample Face Images of UTeMFD Dataset.

Figure 3: Sample Face Images of GTFD Dataset [8].

Figure 4: Sample Face Images of CASIA-FaceV5 Dataset [9].

C. Data Splitting

The first step is to split the dataset into a few sets. In MATLAB, input images were stored inside data-store by using the 'imageDatastore' function and the dataset is randomly split into two sets which are training and test set using the 'splitEachLabel' function. Images from the training set are then randomly split into training and validation set for train and evaluate the model. The training set is used to train the network while the validation set is used to predict the accuracy of the trained model. The flow chart of the dataset splitting is shown in Figure 5.

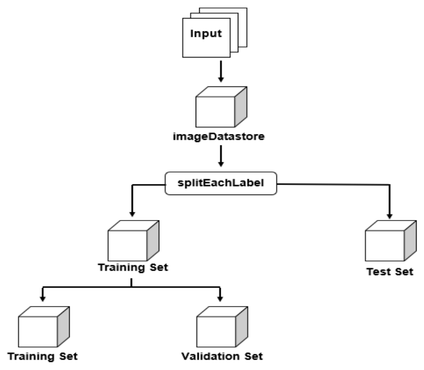


Figure 5: Flow Chart of Dataset Splitting.

D. Pre-processing

Pre-processing is one of the important steps in the face-recognition system. It can help in improving the performance of the system. The input data are processed by following step: face detection, crop face, and data augmentation. For the first step, a cascade object detector is used to detect the human frontal face by the Viola-Jones algorithm. Then, the ROI face region is cropped. Apart from that, the data augmentation technique also be used. The augmented process, will make the image data-store can be transformed batches of training and test set with optional pre-processing such as resizing, rotation, and reflection. The augmented image data-store is used to resize the training images to 227x227 pixels, which match with the input image of the AlexNet CNN pre-trained model.

The 'RandXReflection' function is used to flip the train images along the vertical axis randomly, of x and y axis with a pixel range of -20 and 20 is used to translate the training image up to 20 pixels horizontally and vertically, respectively. This data augmentation helps prevent the network from over-fitting and memorizing the exact details of the training images. Figure 6, the pre-processing steps are shown.

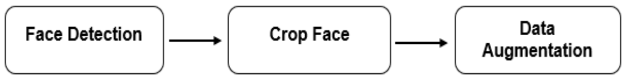


Figure 6: Pre-processing Steps.

E. Compose the model

In this proposed system, pre-trained CNN is used as a feature extractor and transfer learning by fine-tuning technique is used to transfer the extracted features from the pre-trained CNN model to a new task. In this section, the pre-trained CNN architecture and the fine-tuning pre-trained CNN architecture is described in detail.

1) Pre-trained CNN Model as Feature Extractor

AlexNet pre-trained CNN model is selected as the feature extractor in this proposed system. AlexNet has been trained on more than a million images from the ImageNet database and it can classify images into 1000 object categories with about 60 million parameters. The architecture of AlexNet consists of eight learned layers, five convolution layers followed by three fully connected layers. The eight layers AlexNet architecture is shown in Figure 7 below. In MATLAB platform, AlexNet architecture consists of 25 layers: The first 23 layers are for feature extraction, whereas the last three layers are for classifying these features into 1000 classes. AlexNet has an image input size of 227x227x3 images with 'zerocenter' normalization. Then, the first convolution layer filters the 227x227x3 image input image with 96 kernels of size 11×11×3 with a stride of 4 pixels and followed by ReLU non-linear activation, cross channel normalization with five channel per element, and 3x3 max pooling with the stride of 2 and zero padding layer. For the second convolution layer, 256 feature kernels with a size of 5x5x48 filters 27x27x96 feature image and carry out further feature extraction. Same as the first convolution layer, second convolution layer also followed by ReLU nonlinearity activation, cross channel normalization with 5 channels per element, and a 3x3 max pooling with the stride of 2 and zero padding and output of 13x13x256 image.

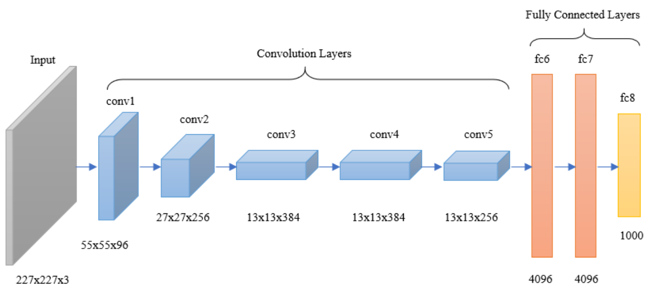


Figure 7: AlexNet Pre-trained CNN Architecture.

Table 2: The Detail of AlexNet Layers in MATLAB Platform.

Layer	Name	Number of Kernels	Kernel Size	Stride	Padding	Output
1	data					227x227x3
2	conv1	96	11x11x3	4	-	55x55x96
3	relu1					55x55x96
4	norm1					55x55x96
5	pool1		3x3	2	-	27x27x96
6	conv2	256	5x5x48	1	2	27x27x256
7	relu2					27x27x256
8	norm2					27x27x256
9	pool2		3x3	2	-	13x13x256
10	conv3	384	3x3x256	1	1	13x13x384
11	relu3					13x13x384
12	conv4	384	3x3x192	1	1	13x13x384
13	relu4					13x13x384
14	conv5	256	3x3x192	1	1	13x13x256
15	relu5					13x13x256
16	pool5		3x3	2	-	6x6x256
17	fc6	1				1x1x4096
18	relu6					1x1x4096
19	drop6					1x1x4096
20	fc7	1				1x1x4096
21	relu7					1x1x4096
22	drop7					1x1x4096
23	fc8	1				1x1x1000
24	prob					1x1x1000
25	output					-

After that, third and fourth convolution layers followed by ReLU nonlinearity activation given an output image of 13x13x384.

For layer 14, fifth convolution layer (conv5) filters 13x13x384 image by 256 feature kernels with 3x3x192 kernels size with padding of 1 and get an output of 13x13x256 image.

Then, it followed by a ReLU activation and a 3x3 kernel max pooling with the stride of 2 to 6x6x256 image. Furthermore, fully connected layers, ‘fc6’ and ‘fc7’ is followed by a ReLU activation and a 50% dropout to overcome over-fitting.

A total of 4096 of 6x6x256 kernels perform convolution operation on the input data and output the operation result through 4096 neurons. 4096 neurons from both ‘fc6’ and ‘fc7’ are fully connected. Lastly, fully connected layer ‘fc8’ which has 1000 neurons are connected to the input which consists of 4096 neurons from ‘fc7’, and softmax activation is used for classification. The detail of AlexNet layers in the MATLAB platform is shown in Table 2.

F. Fine-tuning pre-trained CNN model

In this research, transfer learning by fine-tuning of a pre-trained CNN model is used to classify the images by using a smaller number of training images. Pre-trained CNN model has learned rich feature representations for a large scale of images. Therefore, transfer learning from the pre-trained CNN model is much faster and easier than training a network from scratch.

This proposed approach is using AlexNet pre-trained CNN architecture and transfer the layers to the new classification task by replacing the last three layers with a fully connected layer, a softmax layer, and a classification output layer for a new task.

From layer 23 which name ‘fc8’ with 1000 fully connected layer is replaced to the fully connected layer which has the same size as the number of classes in the new data by using the function 'WeightLearnRateFactor' of 10 and 'BiasLearnRateFactor' of 20 for increase the learning rate in the new replaced layers for learning faster compared to the transferred layers.

Besides, the output layer from pre-trained architecture is replaced to match the number of target class exist on the dataset used. Therefore, the output layer from AlexNet architecture which can classify into 1000 classes is changed to 14 classes , 50 classes for the GTDB dataset, and 500 classes for the CASIA-FaceV5 dataset.

Then, the fine-tuning model is retrained with the training set of the task specified dataset.

The detail of the fine-tuned Alexnet by replaced the last three layers is shown in Table 3 below.

Table 3: Replacement of Last Three Layers of AlexNet for Three Datasets.

AlexNet	Name	Description
	fc8	FCL (1000), pre-trained weight & biases
	prob	SoftMax Layer
	outpu t	Classification Layer (1000 classes)
Fine-tuned Alexnet (GTFD)	**Name**	**Description**
	fc8_2	FCL (50), weight of 10 & biases of 20
	prob	SoftMax Layer
	outpu t	Classification Layer (50 classes)
Fine-tuned Alexnet (CASIA-Face V5)	**Name**	**Description**
	fc8_2	FCL (500), weight of 10 & biases of 20
	prob	SoftMax Layer
	outpu t	Classification Layer (500 classes)
Fine-tuned Alexnet (UTeM FD)	**Name**	**Description**
	fc8_2	FCL (14), weight of 10 & biases of 20
	prob	SoftMax Layer
	outpu t	Classification Layer (14 classes)

Besides, there are five different settings used to compare the effectiveness of the model in this study.
AlexNet CNN model consists of 8 layers such as five convolutional layers (CL1, CL2, CL3, CL4, CL5) and there fully connected layers (FC6, FC7, FC8), the different setting can be tested by varied the numbers of transferred layers.
For setting A, the first layers (CL1) to the seventh layer (FC7) are transferred directly from AlexNet and only the last layer from AlexNet is fine-tuned.
For setting B, layers FC7 and FC8 are fine-tuned by the three layers mentioned in Table 3.
Besides, layers from CL1 to CL5 are transferred directly from AlexNet in setting C and layers from CL1 to CL4 are transferred directly from AlexNet in setting D. Lastly, only layers from CL1 to CL3 are transferred from AlexNet in setting E.
The details of the five different transfer learning setting are shown in Figure 8.

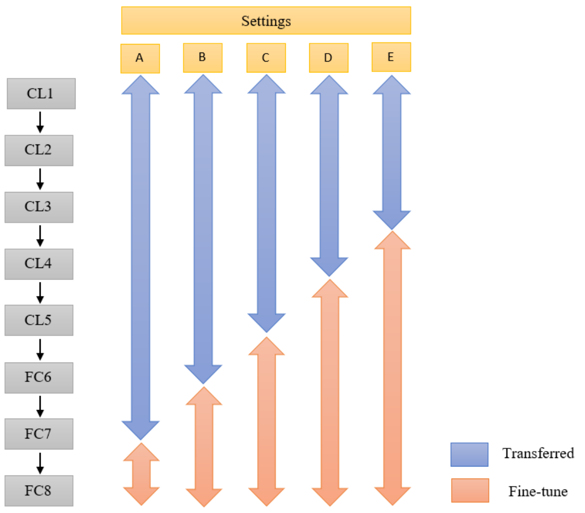


Figure 8: Different Transfer Learning Settings.

G. Train and Evaluate the Model

From the figure 9, the flow chart of the train and evaluate the model was shown in detail. In this section, both the training and evaluating process was described in detail.

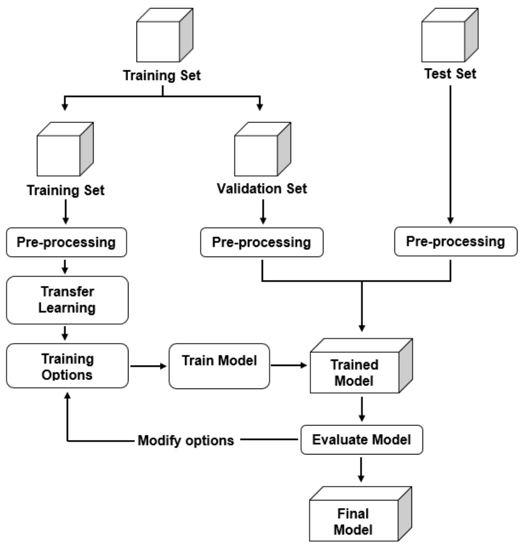


Figure 9: Flow Chart of Train and Evaluate the Model

1) Training Process

In transfer learning, training data with labels, fine-tune network, and training algorithm options were the three important things that need before train a network. Training data is store in an augmented data store and pre-processed while the fine-tuned network is the network that transfers learning from AlexNet CNN pre-trained network to perform a new task. Training data and fine-tune networks are described in detail in 3.5 and 3.6 respectively. For training algorithm options, it was the most important component in the training process as it can control the behavior of the training algorithm. In MATLAB, the ‘trainingoptions’ function is used to set up the training options such as optimizer uses, initial learning rate, mini-batch size, maximum epochs, optimizer, validation frequency, and many more should be specified before starting to train the model. In this proposed system, stochastic gradient descent with momentum (SDGM) optimizer is used as the solver for a training network with the default momentum of 0.9. Optimizer is updated the weight parameters to minimize the loss function. For the learning rate, it was the most important hyper-parameter when configuring the network as it controls the change of the model in response to the estimated error each time the model weights are updated.

Choosing a learning rate is challenging as the value too small may result in a long training process that could get stuck, whereas the value too large may result in learning a sub-optimal set of weights too fast or an unstable training process. At first, I am fine-tuning the learning rate and choose a suitable learning rate for the network.

A larger value such as 0.1 is started to try on the network then reduce the initial learning rate exponentially until 0.0001. Finally, the learning rate was set to a small value as 0.0001 to slow learning down, since the pre-trained CNN is used.

Besides, the size of the mini-batched is set to 30. In this study, different number of epochs are used to test the model until it reaches a good accuracy. The loss and gradient calculated for each mini-batched approximates the loss and gradient for the full training set.

2) Evaluating on Trained Model

In this study, training accuracy, validation accuracy, testing accuracy, mini-batch loss, and validation loss are the evaluation parameters used for the experiments.

For the training process, the validation set is used to evaluate and validate the performance of during training. In MATLAB, training progress can be plot using 'Plots' and 'training-progress' function in training options.

From the training progress, training accuracy, validation accuracy, training loss value, and validation loss value can be observed to prevent over-fitting.

It happens if the training accuracy is greater than the validation accuracy and the validation loss value training loss value is greater than the training loss value.

In the other hand, if the validation accuracy is higher than the training accuracy and the validation loss value is lower than the training loss value, it was under-fitting. Apart from that, the confusion matrix table also used to observe the correct and error between the ground truth images and the predicted images.

H. Test on final model

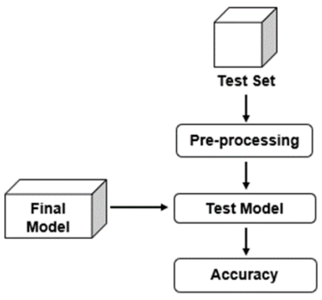


Figure 10: Flow Chart of Testing on Trained Model.

In this study, training accuracy, validation accuracy, testing accuracy, mini-batch loss, and validation loss are the evaluation parameters used for the experiments. For the training process, the validation set is used to evaluate and validate the performance of during training.

III. Result and Discussion

In this session, all the results are presented in table and graph and discussed in detail. Different experiments were done to evaluate the performance approach such as fine-tuning the hyper-parameter to get better accuracy, compared the transfer learning setting, and transfer leaning AlexNet with and without data augmentation technique. Besides, this approach was also compared with other state-of-art models.

A. Setting

This proposed method was conducted using the platform of Windows 10 with the Intel (R) Core (TM) i5-8265U CPU @ 1.6GHz and 1.80GHz processor and 12.0GB of installed RAM. All the algorithms, performance, and analysis were done in MATLAB 2019b tool. All the experiments were tested on three small datasets which are GTFD, Casia-FaceV5, and collected dataset. The dataset was divided into three groups such as training, validation, and testing set with 60%, 20%, and 20% respectively. The dataset split for training, validation, and testing per class are shown in Table 4.

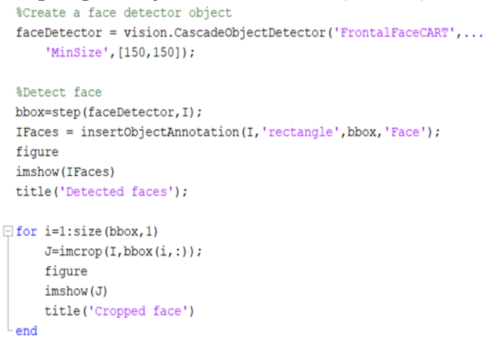
Table 4: Datasets Split for Training, Validation and Testing per Class.

Database	Training images	Training images	Training images	Total
UTeMFD	18	6	6	30
GTFD	9	3	3	15
CASIA-Face V5	3	1	1	5

B. Data Pre-processing and Augmentation

The first step for face recognition algorithms is to detect the human face. In this step, 'vision.CascadeObjectDetector' function is used to create a face detector object with default classification model, 'FrontalFaceCART' and MinSize of [150 150], size of the smallest detectable object. Then, a bounding box with annotated with 'face' is created and the detected ROI region was cropped. The pre-processing algorithm and the step are shown in Figure 11 and 12, respectively.

Step 1: Recognize human face, size 150x150, using 'vision.CascadeObjectDetector'.
Step 2: Bounding box with annotated 'face', 'insertObjectAnnotation'
Step 3: Detect ROI
Step 4: Crop region, by the ratio '1:size(bbox,1)'



```
%Create a face detector object
faceDetector = vision.CascadeObjectDetector('FrontalFaceCART',...
    'MinSize',[150,150]);

%Detect face
bbox=step(faceDetector,I);
IFaces = insertObjectAnnotation(I,'rectangle',bbox,'Face');
figure
imshow(IFaces)
title('Detected faces');

for i=1:size(bbox,1)
    J=imcrop(I,bbox(i,:));
    figure
    imshow(J)
    title('Cropped face')
end
```

Figure 11: Pre-processing Algorithm.

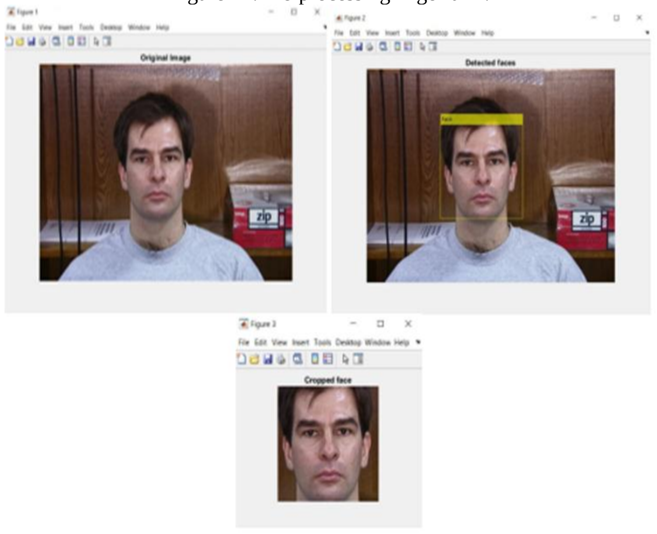


Figure 12: Pre-processing Steps. (a) Original Image (b) Detected Face (c) Cropped Face

Furthermore, data augmentation techniques were used to prevent overfitting and enhance the performance of the model. In this research, DA techniques such as automated resize, reflection, and translation are used.

The training sets are automatically resized to 227 x 227 pixels which match the input size of the AlexNet pre-trained CNN model by using augmented image datastore with 'augmentedImageDataStore' function. Besides, 'imageDataAugmenter' function is used for additional data augmentation such as randomly flip the training images along the vertical axis, translate them up to 20 pixels horizontally and vertically. Figures 13 until 15 show the data augmentation techniques used in the experiment on the GTFD dataset.

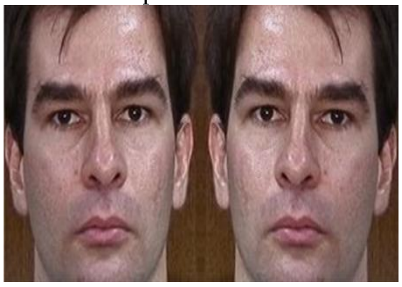


Figure 13: Randomly Flip the Train Images Along the Vertical Axis.

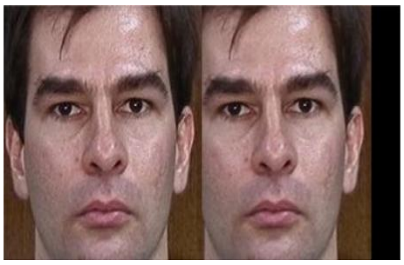


Figure 14: Randomly Translate up to 20 Pixels Horizontally.

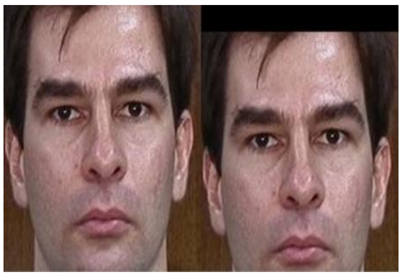


Figure 15: Randomly Translate Up to 20 pixels vertically.

C. Experiments Result and Discussion

The purpose of this research is to verify the effectiveness and accuracy of the proposed method on three different small datasets. UTeMFD (18 training images per class), GTFD (9 training images per class), and CASIA-Face V5 (3 training images per class). Three different experiments are done to achieve the objectives. In this research, different types of datasets as the number of images per subject are different are used where UTeMFD consists of 18 training images per class, GTFD consists of 9 training images per class, and CASIA-Face V5 consists only 3 training images per class. The performance of the proposed method is assessed according to the accuracy and the time discussed in in methodology.

1) Experiments 1: Fine-tuning Hyper-parameter

In deep learning, there is no fixed hyper-parameter used to train a model. Here different number of epochs is used to test the model until it reached a good accuracy, 20th epochs have the best performance as shown in Figure 16. The setting mini-batch size equal to 30 and the global learning rate is 0.0001. The validation set with the validation frequency of 10 is used for validating the network during training.

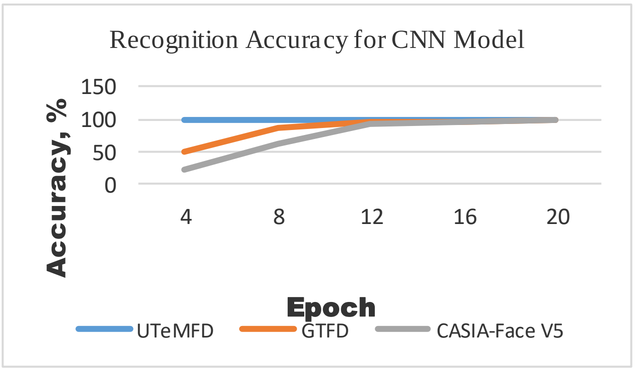


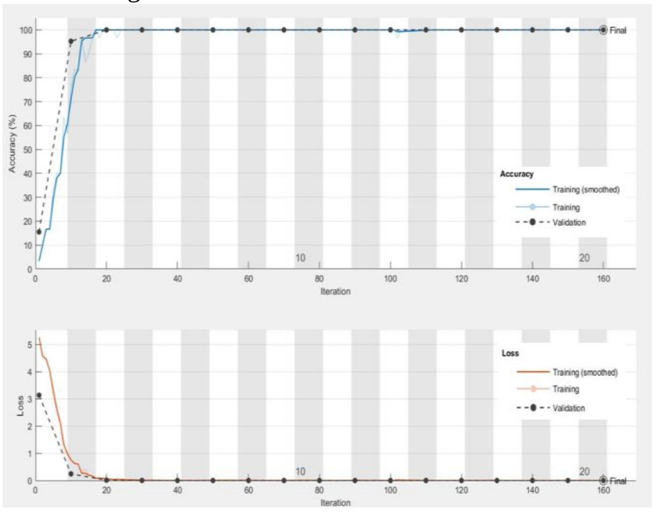
Figure 16: Recognition Accuracy for CNN Model.

From Figure 16, the highest accuracy was obtained with an epoch equal to 20 for three of the datasets. Besides, UTeMFD achieved 100% in less than 4 epochs. For the GTFD, it achieved 100% within 16th to 20th epochs while the CASIA-FaceV5 achieved 100% at 20th epochs. All the experiment is tested in 5 times on GPU environment which can speed up on training process. In this research, Stochastic Gradient Descent with momentum (SGDM) optimization algorithm with 0.9 momentum is used to update the parameter during training. The detailed hyper-parameter settings are shown in Table 5 below. The settings are used for experiments 2 and 3.

Table 5: Hyper-parameter Settings.

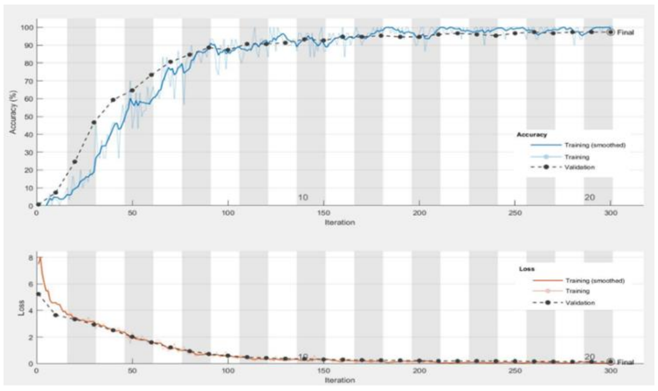
Hyper-parameter	Selected value
Learning Rate	0.0001
Epochs	20
Mini-batch size	30
Validation Frequency	10
Momentum	0.9
Executive Environment	GPU

Besides, the performance of the CNN model can be evaluated and observed from the training progress graph as shown in Figure 17.

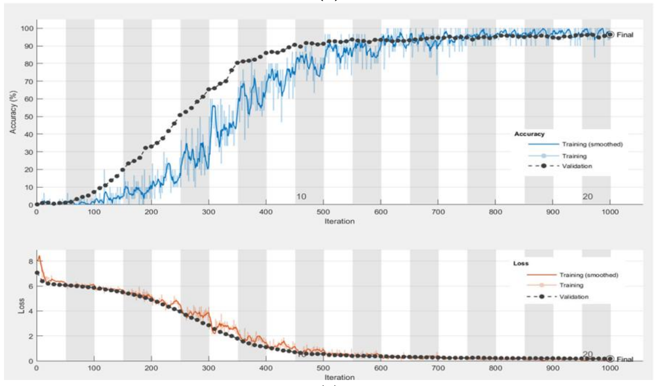


(a)

Figure 17: Training Progress Graph for Accuracy and Loss for Different Datasets (a) UTeMFD (b) GTFD (c) CASIA-Face V5



(b)



(c)

(Continued) Figure 17: Training Progress Graph for Accuracy and Loss for Different Datasets (a) UTeMFD (b) GTFD (c) CASIA-Face V5

From the training progress plot, overfit model is easily diagnosed by monitoring the training accuracy, validation accuracy, training loss, validation loss by evaluating on both training and validation set. If the model is overfitting, the training accuracy will greater than the validation accuracy while the validation loss value will greater than the training loss value. Besides, the average values for 5 run of the results using the same hyper-parameter settings are shown in Table 6 below.

Table 6: Results of Training, Validation and Testing on Different Datasets.

Data	Training Accuracy, %	Validation Accuracy, %	Mini-Batch Loss, %	Validation Loss, %	Training Times, s
UTeMFD	100	100	0.18	0.02	121
GTFD	100	96.40	3.04	15.48	231
CASIA-Face V5	100	94.72	5.80	23.27	1265

As a result, the number of images inside per subject in a dataset will affect the performance of the model. For collected dataset, UTeMFD has the largest number of images in each subject which got 100% accuracy during training and validation and it got the lowest training and validation loss value as 0.18% and 0.02% respectively. For the GTFD dataset, it has a medium number of images per subjects and get the 100% training accuracy while the validation accuracy is lower 3.6% than UTeMFD dataset. The mini-batch loss was 3.04% while the validation loss is 15.48% which was also higher than the UTeMFD dataset. For the CASIA-Face V5 dataset, it consisted of 5 images per subject which was referred to as a small number size per subject. It gets 100% training accuracy while the validation accuracy was the worst as getting only 94.72%. Besides, it also gets a higher mini-batch and validation loss as 5.80% and 23.27% respectively. In this experiment, I also found that a larger size of the subjects will have a longer training time. CASIA-Face V5 has 500 subjects which was a large size and the average training time was 1265 seconds while the training time for UTeMFD was only 121 seconds. Besides, the model starts over-fitting at 80 iterations in Figure 17 (b) while 600 iterations in Figure 17 (c). Smaller training sets have the opportunity to obtain over-fitting models.

2) Comparison of Transfer Learning Setting

This experiment is aimed to select suitable layers to transfer as the feature extractor from three datasets. In this experiment, five different TL setting which described in detail in methodology are compared on the validation set. The experiment is conducted in 5 times for each setting. Then, the results are compared accuracy on the validation set, training and validation loss, and the training time. Besides, I calculate the standard deviation to test the stability of the model. The comparison results for the five run of validation performance of transfer learning using setting A to E for three datasets are shown in Table 7, 8, and 9.

Table 7: Comparison of Different Setting on UTeMFD dataset

Setting	Accuracy, %	Mini-Batch Loss,	Validation Loss	Training Time, s
A	100	0.0018	0.0002	121
B	100	0.00007	0.00006	96
C	100	0.0001	0.0006	76
D	15	2.3336	2.4612	69
E	75.48	0.6775	0.7938	66

Table 8: Comparison of Different Setting on GTFD Dataset.

Setting	Accuracy, %	Mini-Batch Loss	Validation Loss	Training Time, s
A	96.40	0.0304	0.1548	231
B	93.47	0.0366	1.2823	209
C	76.93	0.0041	0.9648	153
D	12.26	3.3339	3.3950	127
E	23.73	2.4108	3.2809	120

Table 9: Comparison of Different Setting on CASIA-FaceV5 Dataset.

Setting	Accuracy, %	Mini-Batch Loss	Validation Loss	Training Time, s
A	94.72 ± 1.24	0.056 ± 0.02	0.23 ± 0.04	1265
B	91.36 ± 1.97	0.014 ± 0.01	0.41 ± 0.05	1124
C	78.48 ± 1.67	0.003 ± 0.001	0.96 ± 0.10	962
D	68.52 ± 2.66	0.001 ± 0.001	1.84 ± 0.32	1165
E	62.16 ± 8.74	0.158 ± 0.261	2.462 ± 0.679	1089

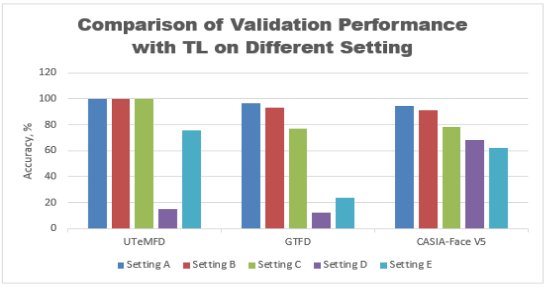


Figure 18: Comparison of the Validation Accuracy of Transfer Learning using Different Settings.

From Table 7, I found that setting A, B and C have 100% accuracy on the validation set while the lowest training loss was 0.0002% and 0.00006% of validation loss using setting B on the UTeMFD dataset. Besides, Setting B had the lowest standard deviation values. On the other hand, setting D which transferred only four layers where C1 to CL4 as the feature extractor has the worst results as the accuracy was only 15% and has the highest training and validation loss value. As a result, the best setting of transfer learning on UTeMFD is setting B which fine-tuned the last two layers, FC7 and FC8. For the GTFD dataset, Setting A which only fine-tuned the last layer (FC8) achieved the highest performance among all five settings concerning all measures which show in Table 8. Setting A achieved 96.40 ± 0.89% accuracy, 0.030 ± 0.02 mini-batch loss, and 0.15 ± 0.03 validation loss. While setting D obtained the worst performance which only achieved 12.26 ± 5.00% accuracy and the highest training and validation loss value which are 3.33 ± 0.40 and 3.40 ± 0.23, respectively. For the CASIA-Face V5 dataset, Setting A was the best performance compared to others as shown in Table 9. The average results run for 5 times achieved 94.72 ± 1.24% accuracy, 0.056 ± 0.02 training loss, and 0.23 ± 0.04 in 1265 seconds. On the other hand, setting E has the worst performance as it only achieved 62.16 ± 8.74% accuracy and the highest mini-batch loss and validation loss which are 0.158 ± 0.261 and 2.462 ± 0.679, respectively. From Figure 18, the comparison of the accuracy of TL using different settings is shown in the chart. The accuracy was decreased gradually from setting A to E on CASIA-Face V5. While the accuracy of the Setting D was dramatically dropped below 20% on the other two datasets, UTeMFD and GTFD. As a result, Setting B in which fine-tuning FC7 and FC8 layers are selected for UTeMFD while Setting A which only fine-tuning the last layer (FC8) is selected for GTFD and CASIA-FaceV5 datasets.

3) Experiments 3: The Effects of Data Augmentation Techniques

This experiment is conducted to answer the question: Does the data augmentation technique can prevent overfitting and improve the performance for training on small datasets? In this experiment, the comparison of the performance of run the transfer learning with data augmentation against runs the transfer learning without data augmentation was discussed. DA techniques are applied on the training set such as resize, reflection, and translation. This technique can be used to prevent overfitting as the network is unable to memorize the exact details of the training data and enhance the performance. For UTeMFD dataset, transfer learning with Setting B was used for the experiment as it shown the best performance obtained from Table 7. Apart from this, configuration of transfer learning for GTFD and CASIA-Face V5 datasets were set to Setting A. All the experiments were runs in 5 times on validation set and the performance of transfer learning without using DA are shown in Table 10 below. The results in terms of training and validation accuracy are above 94%. Besides, Table 11 show the transfer learning's performance with using DA and Table 12 show the comparison of TL with DA against without DA on three datasets. From Table 10, by using UTeMFD, data-set achieved the best performance with 100 % for the training and validation accuracy. Furthermore, it has the lowest training and validation loss value, which are 0.0002 ± 0.0001 and 0.0001 ± 0.0001, respectively. Besides, it's found that strong over-fitting was occurred since the training accuracy is greater than validation accuracy. For training loss value is smaller than validation loss value on both GTFD and CASIA-Face V5 data-sets.

Table 10: The Performance of the Transfer Learning without using DA.

Dataset	Training Accuracy, %	Validation Accuracy, %	Training Loss	Validation Loss
UTeMFD	100 ± 0	100 ± 0	0.0002 ± 0.0001	0.0001 ± 0.0001
GTFD	100 ± 0	96.40 ± 0.89	0.030 ± 0.02	0.15 ± 0.03
CASIAFace-V 5	100 ± 0	94.72 ± 1.24	0.06 ± 0.02	0.23 ± 0.04

Table 11: The Performance of the Transfer Learning with using DA.

Dataset	Training Accuracy, %	Validation Accuracy, %	Training Loss	Validation Loss
UTeMFD	100 ± 0	100 ± 0	0.00005 ± 0.00003	0.0000004 ± 0.0000002
GTFD	93.33 ± 0	96.40 ± 1.00	0.26 ± 0.01	0.16 ± 0.02
CASIAFace-V 5	91.33 ± 1.82	95.24 ± 1.04	0.27 ± 0.05	0.22 ± 0.04

Using UTeMFD still achieved the best performance where it is obtained the highest training and validation accuracy and the lowest training and validation loss value. 100% training and validation accuracy was achieved, and the training and validation loss is much lower than the results from Table 11 as it decreases from 0.0002 ± 0.0001 to 0.00005 ± 0.00003 and 0.0001 ± 0.0001 to 0.0000004 ± 0.0000002, respectively. The over-fitting problems, can be solved after applied data augmentation (DA) technique on both GTFD and CASIA-Face V5 data-sets.

Table 12: Comparison the Performance of TL with DA and without DA.

Dataset	Accuracy without DA, %	Accuracy with DA, %	Different, %
UTeMFD	100	100	-
GTFD	96.40	96.40	-
CASIA-Face V5	94.72	95.24	0.52

From Table 12, the comparison of the performance of TL with DA and without DA shown. The performance of both UTeMFD and GFTD dataset remains the same as 100% and 96.40% validation accuracy with and without using DA. On the other hand, the validation performance is improved by 0.52% which from 94.72% to 95.24% on CASIA-Face V5 after applying DA. From this experiment, it show that the data augmentation technique helped in prevent over-fitting and improve the performance.

D. Results on Proposed Method

In this research, hyper-parameter settings used for training is shown in Table 5. Moreover, Setting A is chosen and applied on GTFD and CASIA-Face V5 as shown in Table 8 and 9. On the other hand, Setting B is chosen and applied to collected dataset, UTeMFD as shown in Table 7. Besides, data augmentation techniques are used to prevent over-fitting and improve the performance. In this section, the retrained neural network was tested on the test set and each dataset was run 5 times to get the average recognition accuracy and time. Table 13 shows the results of the proposed method on the test set.

By refer form Table 4.13, the average recognition accuracy achieved 100% while the recognition time only 0.03 seconds on the UTeMFD dataset. Besides, 96.67% of recognition accuracy and 0.03 seconds recognition time is achieved on the GTFD dataset. Furthermore, the proposed method achieved 95.60% of recognition accuracy and 0.04 seconds of recognition time on the CASIA-Face V5 dataset.

Table 13: Results of Proposed Method on the Test Set.

Dataset	Recognition Accuracy, %	Recognition Time, s
UTeMFD	100	0.03
GTFD	96.67	0.03
CASIAFace-V5	95.60	0.04

E. *Comparison with Previous Researches*

In this section, the results obtained from the proposed study, have been compared with the findings of other researchers. The comparisons are in terms of benchmarking performance measure, as shown in Table 14. The statistical analysis was performed based on the recognition accuracy and time. All the comparative test results among the state-of-the-art methods submitted on the GTFD and CASIA-Face V5 databases. The comparison of my proposed system and other researcher's methods for on GTFD and CASIA-FaceV5 datasets, including TCLBP [10], HOG+SVM [11], sgFKNN1 [12], GC-RF [13], SCN [14], fLLC−SPMScheme1 [15], and SESRC + LDF [16].

Table 14: Comparison with Different State-of-the-Art Approach.

Source	Method	Dataset	Accuracy, %	Time, s
Ze Lu et. Al [38]	Ternary-Color LBP (TCLBP)	Color FERET	86.73	-
		GTFD	94.57	-
		FRGC	75.32	-
		LFW	78.57	-
(Harihara et al. 2016) [39]	HOG + SVM	ColorFERET	68.50	-
		Yale Database	92.00	-
		BioID	75.67	-
		GTFD	81.25	-
		FEI	80.13	-
		LFW	64.60	-
(Kasemsumran et al. 2016) [40]	sgFKNN1	GTFD	79.57	-
(See et al. 2017) [41]	GC-RF	GTFD	89.20	-
	PCA		50.40	-
	LDA		65.60	-
	Gabor-PCA		55.60	-
(Minaee et al. 2018) [42]	Scattering Convolutional Network	GTFD	90.00	-
(Umer et al. 2015) [43]	fLLC−SPMScheme1	**CASIA-Face V5**	67.56	-
(Mengmeng et al. 2020) [44]	SESRC + LDF	GTFD	92.43	-
	VGGFace		90.27	-
	VGGFace2		91.43	-
	ArcFace		92.00	-
Proposed Method	TL + CNN	UTeMFD	100	0.03
		GTFD	96.67	0.03
		CASIA-FaceV5	95.60	0.04

For the GTFD dataset, this proposed system is 2.1% better than other researchers' methods. The higher results among these six methods, TCLBP [10] achieved the highest performance which is 94.57%. Where other researcher methods such as HOG+SVM [11], Scattering Convolutional Network [14], and SESRC+LDF [16] achieved 90% accuracy and above. Besides, sgFKNN [12] is the worst performance compared to others as it only achieved 79.57%. This is because the GTFD dataset is captured at different scale (sections), so the system cannot function properly and a misclassification has occurred. For the CASIA-FaceV5, my proposed system is 28.04% improvement compared to fLLC−SPMScheme1 [15]. CASIA-FaceV5 consists only 5 images in each subject and the image quality is low, conventional machine learning method cannot perform well in this dataset.

IV. CONCLUSION

In this experiment, a face-recognition algorithm using transfer learning and AlexNet pre-trained CNN model on small data-sets is proposed. Three data-sets such as UTeMFD, GTFD, and CASIA-Face V5 are used. Since training deep learning using small data-set, hence the data augmentation technique is applied. This can prevent over-fitting and enhance the performance of the model. Several experiments were conducted get the best performance of the model. From the experiments, found that by increasing the training set will achieve better performance. This experiment achieved 100% validation accuracy on UTeMFD, 96.40% accuracy on the GTFD, and 95.24% on the CASIA-Face V5 data-set. For recognition time for three data-sets is less than 0.05 seconds for recognition single face. Besides, AlexNet pre-trained CNN model was the only model used as a transfer learning model in this project. For future work, other deeper pre-trained CNN models such as ResNet, DenseNet, VGG-Net, GoogleNet can be applied for the transfer learning model by comparing and analyzing which model can give good performance and stability.

ACKNOWLEDGMENT

Special thanks to Universiti Teknikal Malaysia Melaka (UTeM), and the team work from CeTRI group.

REFERENCES

1. Z. Akhtar and A. Rattani, "A Face in any Form: New Challenges and Opportunities for Face Recognition Technology," Computer, vol. 50, no. 4, pp. 80–90, 2017.
2. S. Singh and S. Prasad, "Techniques and Challenges of Face Recognition: A Critical Review," Procedia Computer Science, vol. 143, pp. 536–543, 2018.
3. D. S. Trigueros, Li Meng, and Margeret Hartnett, "(PDF) Face Recognition: From Traditional to Deep Learning Methods," ResearchGate, 2018.
4. M. D. Kelly, Visual identification of people by computer. 1971.
5. I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. Cambridge, MA: The MIT Press, 2017.
6. N. Aloysius, M. Geetha, A review on deep convolutional neural networks, in: Communication and Signal Processing (ICCSP), 2017 International Conference on, IEEE, 2017, pp. 0588–0592.
7. C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A Survey on Deep Transfer Learning," Artificial Neural Networks and Machine Learning – ICANN 2018 Lecture Notes in Computer Science, pp. 270–279, 2018.

8. "Georgia Tech face database," 17-June-2020 [Online]. Available: http://www.anefian.com/research/face_reco.htm.
9. "CASIA-FaceV5," 17- June-2020 [Online]. Available: http://biometrics.idealtest.org/
10. Z. Lu, X. Jiang and A. Kot, "A novel LBP-based Color descriptor for face recognition," 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, 2017, pp. 1857-1861, doi: 10.1109/ICASSP.2017.7952478.
11. H. S. Dadi and G. K. M. Pillutla, "Improved Face Recognition Rate Using HOG Features and SVM Classifier," IOSR Journal of Electronics and Communication Engineering, vol. 11, no. 04, pp. 34–44, 2016.
12. P. Kasemsumran, S. Auephanwiriyakul and N. Theera-Umpon, "Face recognition using string grammar fuzzy K-nearest neighbor," 2016 8th International Conference on Knowledge and Smart Technology (KST), Chiangmai, 2016, pp. 55-59, doi: 10.1109/KST.2016.7440531.
13. Y. C. See, N. M. Noor, J. L. Low and E. Liew, "Investigation of face recognition using Gabor filter with random forest as learning framework," TENCON 2017 - 2017 IEEE Region 10 Conference, Penang, 2017, pp. 1153-1158, doi: 10.1109/TENCON.2017.8228031.
14. S. Minaee, A. Abdolrashidi and Y. Wang, "Face recognition using scattering convolutional network," 2017 IEEE Signal Processing in Medicine and Biology Symposium (SPMB), Philadelphia, PA, 2017, pp. 1-6, doi: 10.1109/SPMB.2017.8257025.
15. S. Umer, B. C. Dhara and B. Chanda, "Biometric recognition system for challenging faces," 2015 Fifth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), Patna, 2015, pp. 1-4, doi: 10.1109/NCVPRIPG.2015.7490063.
16. M. Liao and X. Gu, "Face recognition approach by subspace extended sparse representation and discriminative feature learning," Neurocomputing, vol. 373, pp. 35–49, 2020.

AUTHORS PROFILE

Ir. Dr. Ridza Azri Ramlee holds a Bachelor Engineering (Horns.) (Electrical) in 2000 and Msc. in Telecommunication and Information Engineering in 2008 from University Teknologi MARA, Malaysia. He was awarded as Doctorate in communications and network engineering from Universiti Putra Malaysia in 2019. The current position is a senior lecturer of Universiti Teknikal Malaysia Melaka (UTeM). He also obtains the corporate member MIEM and professional engineer with practicing certificate (PEPC) BEM since 2012. In addition, he was one of the mentors in the IEM Mentorship Program. He was appointed as a reviewer in several publications, conference chair for many national and international journals and conferences. In additional, he is active in publication with h-index and i10-index 9 of Google's scholar and h-index 5 of Scopus index. He has total 15 years of working experience as a Biomedical Engineer at Pusat Perubatan UKM and as a lecturer at UTeM.

Siva Kumar Subramaniam CEng MIET, AFEA. Awarded PhD Electrical and Electronics Engineering in 2017. His research on various factors in WSN which influence long range data transmission in linear topology using WiFi IEEE 802.11 for the application in oil and gas pipelines. The area of research includes WSN for condition monitoring in oil and gas pipelines (for NDT). Develop novel concepts to improve/introduce better solutions for the research challenges in WSN with proof of concept by means of simulations and demonstrations for various scenarios of long range data transmission in linear topology. Master of Science (M.Sc.) Electronics (2009), and B.Eng Electronic (2006).

Mohamad Harris Misran is awarded PhD from Universiti Teknologi Malaysia in 2018. Master Engineering from University of Wollongong 2009 and B.Eng Hons from University Surrey 2006.

Asem Khmaj awarded Doctor of Philosophy (PhD) in field Of Study Computer and embedded system engineering 2017. Now serve as HOD - Faculty of Engineering, Computer and Communication systems, Zawiya University.

Yvonne Yap is a student in Faculty of Electronics and Computer Engineering in Universiti Teknikal Malaysia Melaka. Awarded Bachelor of Electronic Engineering with Honours in 2019.