The Regularization Effect of Pre-activation Batch Normalization on Convolutional Neural Network Performance for Face Recognition System Paper

Abu Sanusi Darma[1]
University Sultan Zainal Abidin, Faculty of Informatics and Computing, Campus Besut, 22200, Terengganu, Malaysia[1]
Al-Qalam University Katsina, School of Natural & Applied Sciences, Department of Mathematical Sciences, P.M.B. 2137, Katsina, Nigeria[1]

Fatma Susilawati Binti Mohamad[2]
University Sultan Zainal Abidin
Faculty of Informatics and Computing
Campus Besut, 22200
Terengganu, Malaysia

Abstract—**Face recognition is of pronounced significance to real-world applications such as video surveillance systems, human computing interaction, and security systems. This biometric authenticating system encompasses rich real human face characteristics. As such, it has been one of the important research topics in computer vision. Face recognition systems based on deep learning approaches suffer from internal covariate shift problems that cause gradients to explode or gradient disappearance, which leads to improper network training. Improper network training causes network overfitting and computational load. This reduces recognition accuracy and slows down network speed. This paper proposes a modified pre-activation batch normalization convolutional neural network by adding a batch normalization layer after each convolutional layer within each of the four convolutional units of the proposed model. The performance of the proposed model is validated with a new dataset, AS-Darmaset, which is built out of two publicly available databases. This paper compared the convergence behavior of four different CNN models: the Pre-activation Batch Normalization CNN model, the Traditional CNN without Batch Normalization, the Post-Activation Batch Normalization CNN model, and the Sparse Batch Normalization CNN Architecture. The evaluation results show that the recognition performance of Pre-activation BN CNN has training and validation accuracies of 100.00% and 99.87%, the Post activation Batch normalization has 100.00% and 99.81%, and the traditional CNN without BN has 96.50% and 98.93%. The sparse batch normalization CNN has 96.25% and 97.60% success rate, respectively. The result shows that the Pre-activation BN CNN model is more effective than the other three deep learning models.**

Keywords—Face recognition; pre-active batch normalization; convolutional neural network

I. Introduction

Face recognition systems have witnessed a lot of recent advancements in terms of selective human-machine interfaces, such as the future ATM authentication and authorization system, driving licenses identification and verification, and so on [1]. Although face biometric authentication systems have made significant progress and are now widely used in a variety of applications such as criminal investigation, lie detection systems, clinical medicine, distance education, security systems, access control, video surveillance, commercial areas, and even use in social networks such as Facebook [2, 3]. The standard tradition machine learning for face recognition are still plagued by a slew of issues and are only effective in a limited number of scenarios. In terms of major intrapersonal changes in illumination, facial expressions, posture, occlusions, views, and other factors, that lead their performance begins to deteriorate [4]. Furthermore, light intensity, the number of light sources, light direction, and camera angle are all unpredictable. However, these methods extract a small number of image features in lower recognition accuracy, which cannot satisfy human face recognition in complex conditions [5]. With deep learning-based approaches for image feature extraction, superior performance has been achieved. Convolutional Neural Network has become a popular method for face recognition system. The CNN, which automatically extracts a variety of features of the image and classify, has good robustness to complex environments [2]. The architecture of CNN is inspired by biological processes and loosely based on the responses of the neurons in the receptive field of the human brain visual context [7]. Convolutional Neural Networks (CNN) are usually composed of convolutional layer, normalization layer, activation layer, max-pooling layer, and fully connected layers [8]. The driving factor for their successes has been the abundance of available data via the internet and the huge efforts of the research community to create large hand label dataset such as ImageNet [9]. A recent improvement called batch normalization [8], accelerate the learning process by computing batch statistic, it makes normalization an internal part of the model architecture. These changes allow for much faster convergences, diminishes the impact of model initialization, and act as a regularization method [10].

In this paper, we explore the impact of key architectural elements of a convolutional neural network. These are the batch normalization and dropout layers in the context of pre-active batch normalization architecture of convolutional neural networks [11].

II. Major Contribution

The main contribution of this paper is to enhance the performance of Convolutional neural network architecture for face recognition with a higher recognition rate. In this research, we built an improved Pre-active Batch Normalization CNN algorithm by applying a batch normalization layer immediately after each convolutional layer before the non-linear (ReLu) activation function to perform the normalization operation thereby reducing the internal covariate shift. We again added a dropout layer in between the two fully connected layers to help improve the network performance. The general structure of the proposed model is made up of four (4) convolutional units, one dropout layer, two fully connected layers, one softmax layer, and one classification layer. First, confirm that you have the correct template for your paper size.

The rest of this paper is organized as follows: Section discusses various works of literature. The proposed research methodology is highlighted in Section IV. Sections V and VI present the research experiments. Finally, Section VII consists of the research paper conclusions.

III. Related Work of Literatures

Compare to previous literature work, our main contribution is the application of four batch normalization layers to the four convolutional units of a simple convolutional neural network for face recognition application. A dropout layer is also added in between the two fully connected layers. This is done to prevent gradient exploding or gradient disappearance, prevent network overfitting and increase performance with higher recognition accuracy. Convolution Neural Network was first proposed by LeCun and it was firstly applied in handwriting recognition [12]. Literature [3] proposed a modified CNN architecture for face recognition application by adding two normalization layers for the output of the first and last convolutional layers to accelerate the network. The result showed a satisfying recognition rate of 98.8%. Literature [2]. Explored the efficiency of a new sparse batch normalization CNN to overcome the problem of gradient disappearance and gradient exploration faced by the facial expression recognition model. There proposed model uses continuer convolution at the begging of the network to enhance the integrity of the facial regional features. Batch Normalization is sparely added to facilitate network training. The experiment shows that the model has a sufficient performance of 96.87% recognition rate to satisfy the 7 classes of the express. Literature [13] proposed a CNN model for a real-time face recognition system. Model architecture has no batch normalization layer, but it has a dropout layer. The performance of the model architecture is evaluated by turning various parameters of the model to enhance the recognition rate. Maximum accuracy of 98.75% and 98.00% is obtained. Literature [14] proposed an end-to-end face recognition system based on 3D face texture. Combining the geometric invariants, histogram of orientated gradient, and the fine-tuned Residual Neural Network. Batch Normalization is added to each convolutional layer to affine transformation on the input of each layer. The experimental results show that the best top 1 accuracy is up to 98.26% and the top 2 accuracy is 99.40% respectively. Literature [15] tested the performance of their proposed CNN for face recognition with three well-known image recognition methods PCA, LBPH, and KNN. Batch normalization and dropout are not employed in the model architecture. The experimental result shows that the proposed CNN has obtained the best recognition rate of 98.3%. The proposed method based on CNN outperforms the state-of-the-art methods. Literature [16] proposed to used and integrate deep learning CNN for human face Analytic and recognition for diversified applications. In their study the profound learning-based methodology of CNN with fuzzy logic is introduced, so the higher level of exactness in the face grin should be possible. With this method, the predictive feature of the human face can be used for a criminal investigation of the social analytics-based application. This model was training without a batch normalization layer. The model achieves a recognition accuracy of 98.12%. Literature [17] Evaluates the robustness of three well-known approaches by combining CNN as a powerful feature extraction algorithm followed by SVM as a high classifier and PCA for feature dimensional reduction technique. The result of the combined models provides a significant performance improvement and enhances recognition rate up to 95.2%.

IV. Methodology

A. *Pre-Activation-BN-Convolution Neural Network (PABNCNN) Approach for Training and Features Classification*

In this research, we aim to enhance the performance of the face recognition system, based on Deep Learning Convolutional Neural Network (CNN). The study proposed to reduce covariate shift, gradient disappearance and gradient explosion for better network convergence. Therefore, to achieving these aims the study proposed a new method called Pre-Activation Batch Normalization Algorithm. The method is powerful in preventing vanishing gradient and overfitting of the model. The proposed model is trained with Mini-Batch Stochastic Gradient Descent training algorithm. The block scheme of the proposed algorithm is shown in Fig. 1.

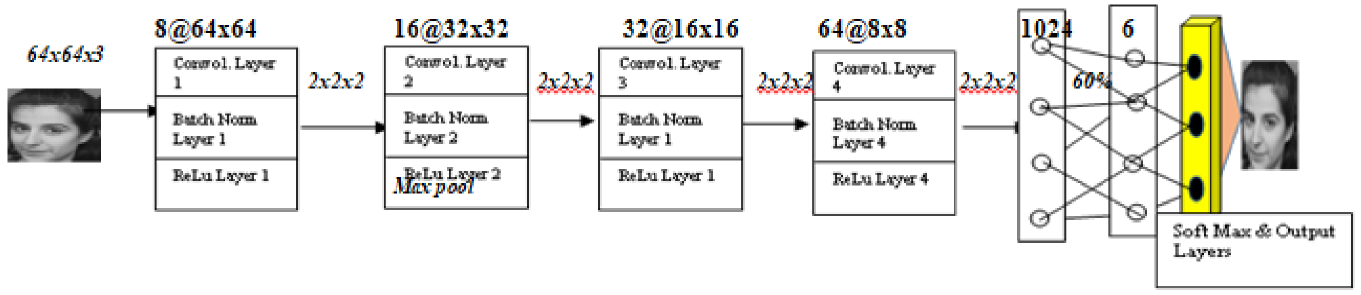


Fig. 1. The Structure of the Proposed Pre-Activation BN-CNN.

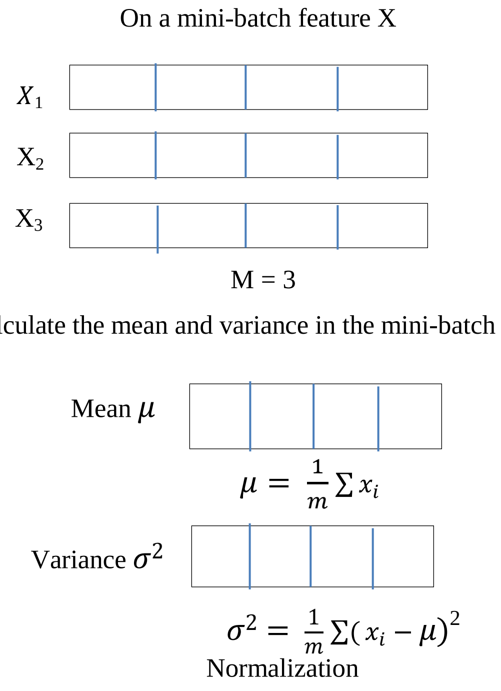
- Convolutional Neural Network (CNN)

A CNN is made up of numerous layers with image processing activities that are built into its structure. This deep learning model was built using three structural representations: shared weights, local receptive field, and subsampling. The convolution kernel shapes shared weights to reduce the number of free parameters. For the feature maps at each layer, the convolution kernels are subjectively changed to build a noise filter as well as an edge detector. Both the convolution and subsampling kernels benefit from the local receptive field since it enchants a set of nearby pixels for further processing before transferring the result of a coarser resolution to the next convolutional layers. While reducing the feature map scope at the corresponding layer, the subsampling process involves local averaging [6]. A new trend in CNN comes with a batch normalization algorithm integrated into the architecture, thereby enhancing the network performance and training speed.

- Batch Normalization

The goal of batch normalization is to achieve a stable distribution of activation values through training. Batch normalization has been established as a component in deep learning, largely helping to push the frontiers of computer vision [18]. BN normalized the means and variance computed within a mini-batch. This contributes tremendously to simplifying the optimization and enabling the network to converge [19]. Therefore, in batch normalization, the data distribution has the attribute that the mean of the data distribution is 0 and the variance is 1 [8]. The batch normalization performs well at medium and large batch sizes and has good generalization to multiple vision tasks [20]. Step of batch normalization operation can be presented as seen in Fig. 2.

- During Training


On a mini-batch feature X
X_1
X_2
X_3
M = 3
Calculate the mean and variance in the mini-batch
Mean μ
$\mu = \frac{1}{m}\sum x_i$
Variance σ^2
$\sigma^2 = \frac{1}{m}\sum(x_i - \mu)^2$
Normalization

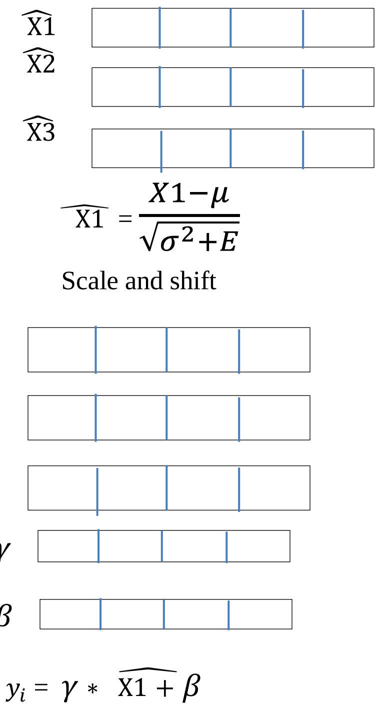


Fig. 2. Figurative Representation of Batch Normalization Operation.

These can be represented in the formula below:

$$\mu = \frac{1}{m}\sum x_i$$

$$\sigma^2 = \frac{1}{m}\sum(x_i - \mu)^2$$

$$\widehat{X1} = \frac{X1-\mu}{\sqrt{\sigma^2+E}}$$

$$y_i = \gamma \ \widehat{X1+\beta}$$

Where μ is the mean value of the output layer $l,$ and σ^2 is the variance value, while $\widehat{X1}$ is the normalization output obtained after subtracting the mean and dividing it by standard deviation$\frac{\boldsymbol{x1-\mu}}{\sqrt{\boldsymbol{\sigma^2+\Sigma}}}$, while y_i set the mean and variance to the new value. γ *and* $+\beta$ are learnable parameters.

With this operation by BN, after processing the problem of multiple layers, mutual coupling in network training weight updating can be solved, and the possibility of gradient disappearance or gradient explosion can be reduced.

B. *The Design Principle of the Proposed Pre-activation BN-CNN Approach*

The proposed Pre-activation BN-CNN consists of one input layer, four convolution units, one dropout layer, two fully connected layers, one Soft-Max layer, and one classification layer. The first convolution unit involves the first convolution layer labelled as C1 in Fig. 1. This layer is followed by the batch normalization layer, which is labelled as BN1, then the Rectifier linear unit (ReLu) activation function, which is labelled as R1, and the Max pooling (down sampling) layer, which is labelled as Mp1. The second convolution unit, the third, and the fourth all follow the same design format as the first convolutional unit, having the batch normalization layer after each convolutional layer before the activation function.

In this paper, the relevant parameters of the proposed Pre-Activation Batch Normalization CNN Architecture are selected according to the proposed face images. The size of the input images is 64x64x3. The four convolutional kernels' sizes are set to 3x3, K1 = 3, K2 = 3, K4 = 3. All the convolutional strides, S1, S2, S3, and S4 are set to 1. In all the four convolutional units, we set the size of the convolutional layers as C1 = 64, C2 = 32, C3 = 16, C4 = 8. Each convolution layer is followed by a batch normalization layer before the activation function. Rectifier linear Unit (ReLu) is the activation function in all the four convolutional units, followed by a max-pooling layer for the down sampling operation. The operation has k = 2, with a stride of 2. This gives us the max-pooling result as C1 = 64, C2 = 32, C3 = 16, C4 = 8. All the feature maps F1 = 8, F2 = 16, F3 = 32, F4 = 64, and all the C4 = 8x8 max pool to 4x4x64, which were expanded into a one-dimensional vector and then connected to the first fully connected layer that is composed of 1024 neurons, a dropout is added between the first and second fully connected layers. And then 6 neurons are connected as the category of the six classes of different variations.

C. Data

This research paper proposed using two publicly available databases, namely the Label Faces in the Wild database and the Caltech 101_Object_Category database. These two databases are used to build a suitable database of 5280 face images, which we named AS_Darma. The 5280 face images were selected from the above two databases. The Label Faces in the Wild has 13,233 target face images of 5749 different individuals. In this database, there are 1680 individuals with two or more images. The remaining 4069 people have just single images. In the database, the image size in this database is 250x250 pixels and is in JPEG image format. Many of the images are in the r.g.b. color scheme. In this research, we selected 4,200 face images of 105 individuals. All of the selected images are in the same r.g.b. color scheme, resized to 64x64 pixel size in the same jpeg format, Fig. 3.

While the Caltech 101_Object_Category database has 450 face images of 27 unique subjects, The images are 325x495 pixels in jpeg format with a different expression, background, and light, but this research proposed cropping and resizing each face image to 64x64 pixels.

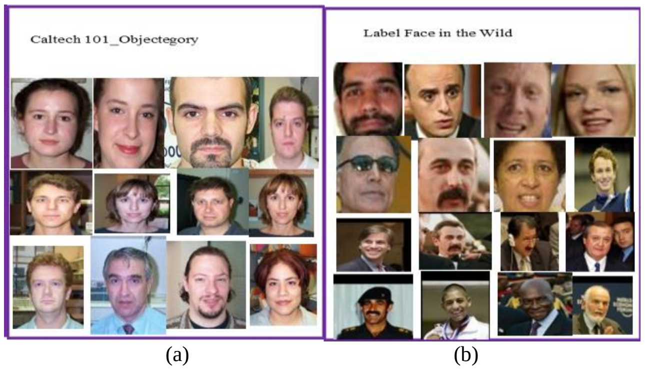


Fig. 3. (a) A Sample of Face Images from the Caltech 101_Object Category Database. (b) Label Face in the Wild Face Images.

In this research, we proposed using 5,280 human face images of 132 individuals. Each individual has 40 face images. To achieve this, we used dynamic data augmentation and preprocessing techniques to produce several synthetic images for each face image of an individual.

We produced 1,080 face images for 27 individuals from the Caltech 101_objects_categories database and 4,200 images for 105 individuals from the LFW. This gives us a total number of 5280. The 5280 images were split into 5280/100 x 70 = 3,696, which is 70%, and 5280/100 x 30 = 1,584, which is a 30% ratio. For both training and testing, 70% is for the training and the other 30% is for the testing. The 5280 images are used to form the proposed dataset called AS_Darmaset for the training and testing of the proposed deep learning architectures.

V. Experiment

This section provides an overview of the experimental setup that is used to verify the effect of our proposed Pre-Activation-Batch-normalization-CNN-Architecture [21]. The experiments were conducted using the newly built AS_Darmaset of 5280 face images. To understand the benefits of adopting a powerful deep learning batch normalization algorithm for enhancing the performance of the face recognition system. In this research, we compared the accuracy and loss errors produced by the four deep learning architectures of different batch normalization approaches. To help in determining the best and most robust batch normalization algorithm between the four models in terms of performance and recognition accuracy rate, the experiment will look at how these regularization techniques can improve network stability and performance.

MATLAB R2018b is used to develop and implement the four deep learning batch normalization CNN architectures. It is used for conducting the experiments as well. The Matlab software is used because it is the best programing tool for engineering and artificial intelligence systems. The architecture of our models is meant to run on the HP Elite Book 854w, Mobile Workstation. The CPU is an Intel Core i7 M620 @ 2.67GHz processor with internal physical memory of 8.00GB. Four experiments were conducted.

A. Experiment with Pre-activation Batch Normalization CNN Architecture

We first conducted the experiments using a Pre-activation Batch Normalization CNN architecture. With this batch normalization algorithm, the batch normalization layers are placed immediately after the convolutional layer in each convolutional unit of the network. This means that the batch norm layer is applied before each of the Rectifier Linear Unit (ReLu) activation functions. The performance of this architecture is evaluated using the proposed AS_Darmaset. In this research, the dataset is categorized into six classes of different face image variations. The six classes of variations include Facial Expiration, Facial Makeup, Occlusion, Old Age, Pose variation, and Younger Age variation. In each class, there are 880 face images of 22 individuals, and each person has 40 face images of size 64x64 pixels.

- Design Principle of Pre-activation Batch Normalization CNN Architecture.

Compute the input $\rightarrow$ Batch Norm $\rightarrow$ Applied Activation $\rightarrow$ Compute Next layer input $\rightarrow$ Batch Nor $\rightarrow$ Applied Activation

The performance of this model for face recognition is evaluated across six different classes of human face variation. Facial Expression, Makeup, i.e., cosmetic effects, occlusion, faces of older age, pose variation, and faces of younger age. There are 880 images in each of the different classes. Each class has 22 individuals, and each of the individuals has 40 face images of 64x64 pixel size.

Table I shows the information obtained from the training plot of the experiment, which shows the training accuracy, validation accuracy, training loss, and validation loss as illustrated in Fig. 4.

In the above figure, the first graph at the top shows the training accuracy (i.e., classification accuracy). The X-axis has 90 epochs and 810 iterations, while the Y-axis shows the accuracy values in percentages. The second graph at the bottom shows the loss function (cross-entropy loss). The training plot is for monitoring the status of the network. It shows how the network's accuracy is increasing. The upper side of the graph shows the performance accuracy, while the lower side of the graph shows the loss function. The graph shows the training matrices at each.

TABLE I. TRAINING PLOT DETAILS FOR PRE-ACTIVATION BATCH NORMALIZATION CNN TRAINING FROM THE SCRATCH

Parameters	Values
Training Accuracy	100.00%
Validation Accuracy	99.87%
Training Status	Completed
Elapsed Time	39 min 20 sec
Number of Epoch per Iteration	9
Mux. Number of Iteration	810
Validation Frequency	80 iteration

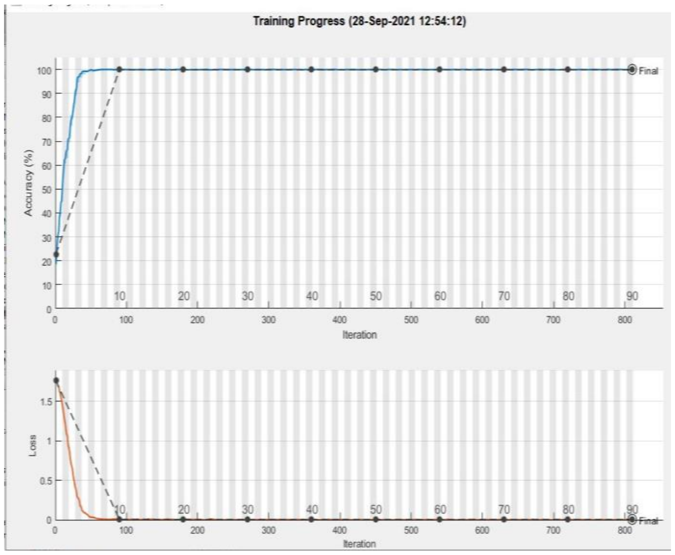


Fig. 4. Training Progress Plot Graph for Pre-Activation BN CNN Architecture.

Iteration: That is the estimation of the gradient [22]. The classification accuracy is represented by a light blue line, while the dark blue line represents the accuracy obtained by applying a smoothing algorithm to the training accuracy. While an interrupted black dotted line is defined as the classification accuracy of the whole validation dataset. At epoch 10 and iteration 90, the network started to converge with 100% training accuracy and validation accuracy of 99.87%. At epoch 80 and iteration 720, the training accuracy decreases to 99.75%, while the testing (validation) continues to maintain its accuracy value of 99.87%. Then, at epoch 84, iteration 750, the training accuracy improved to its normal 100% accuracy. Finally, the training and testing accuracies are 100% and 99.87%, respectively, at last epoch 90 and iteration 810 [21]. The Pre-Activation-CNN Architecture has yielded a better classification performance of 100% training accuracy and 99.87% validation accuracy. Without network overfitting, network convergence is successful. The loss function is shown on the second graph at the lower end. The light orange line is training loss, the smooth training loss is a dark dotted line, and the validation loss is a disrupted line, meaning the loss on each mini-batch and the validation dataset [23]. The figure shows that both the training and validation loss functions have converged to the minimum as the learning rate reached 1.600e-05. While the number of iterations reached 810 at 39 min 20 sec of training time.

B. *Experiment with Traditional CNN Model without BN Algorithm*

The traditional CNN model has a simple architecture of four convolutional units. In each of the units, there is one convolutional layer followed by a Rectifier linear unit (ReLu) activation function, then a max-pooling layer and a down sampling layer. This shows that there is no batch normalization algorithm in any of the four convolutional units. The architecture of this CNN is comprised of four (4) convolution layers, four (4) max-pooling layers, and two (2) fully connected layers. There is no batch normalization layer in the design principle of this model. The model was implanted in MATLAB R2018b and all the trainable parameters (i.e., layer weights and biases) were initialized with the Rectifier Linear Unit (ReLu) activation function at each convolutional process. Fig. 5, shows the training progress plot graph and the details regarding the training phase are listed in Table II below.

Table II shows the training details with training and validation accuracies of 96.50% and 98.93%, respectively as shown in the training graph below.

TABLE II. TRAINING PLOT DETAILS FOR PRE-ACTIVATION BATCH NORMALIZATION CNN TRAINING FROM THE SCRATCH

Parameters	Values
Training Accuracy	96.50%
Validation Accuracy	98.93%
Training Status	Completed
Elapsed Time	32 min 50 sec
Number of Epoch	9
Mux. Number of Iteration	810
Validation Frequency	80 iteration

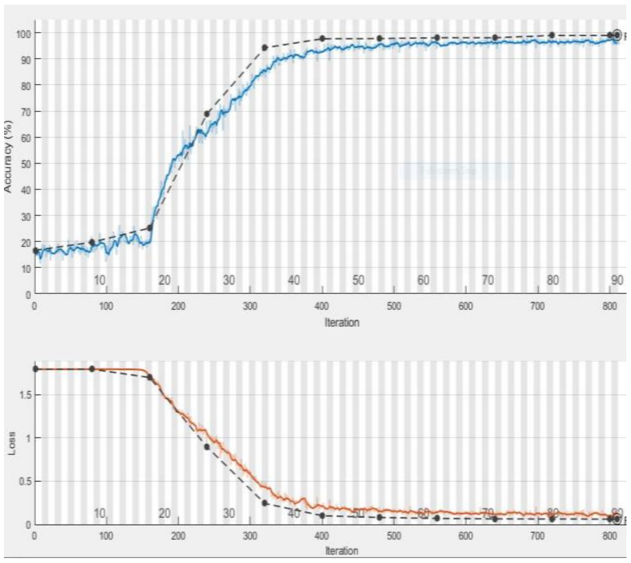


Fig. 5. Training Progress Plot Graph for Traditional CNN without BN Architecture.

In the above graph, the training and testing accuracies at epoch 1 and iteration 1 are initialised to 17.25% and 16.67%, respectively. The accuracy sharply increases at epoch 27 and iteration 240 to 62.00% and 69.00%, respectively. After epoch 50 and iteration 450, the accuracy reaches 93.50% and 97.66%. Similarly, at epoch 63, iteration 560, both the training and validation accuracies continue to increase to 96.50% and 98.04%. At epoch 78, iteration 700, the training accuracy decreased to 95.75%, while the testing accuracy of 98.04% was maintained. Finally, at the last epoch, 90 and iteration 810, the training and testing accuracies increased to 96.50% and 98.93%, respectively. The result is shown in the graph, which shows that the training data accuracy is higher than the testing dataset accuracy throughout the training process. This implies that there is large overfitting that occurred, particularly at the beginning of the training because the batch normalisation has not been implemented. While in the second graph, The training and testing losses, which represent the degree of differences between the model prediction and the real classes, decrease with increasing epoch number [23]. At epoch 1 and iteration 1, the training and testing losses were 1.7917 and 1.7918, respectively. While the number of epochs reaches the final stage, which is 90 epochs and 810 iterations, the training and testing losses were 0.1094 and 0.0628. This indicates that the model has overfitting and gradient disappearance caused by covariate shift.

C. Experiment with Post-activation Batch Normalization CNN Architecture

The Post-activation Batch Normalization CNN architecture has the following setup in the model design principle: Four (4) convolution layers, four (4) batch normalization layers, four (4) max-pooling layers, and two fully connected layers. In this architecture, the batch normalization layer is applied after each of the rectifier linear unit (Relu) activation functions within all four convolutional units. Table III shows the training details and Fig. 6: The Post-Activation Batch Normalization CNN Algorithm Training Progress Plot Graph.

TABLE III. TRAINING PLOT DETAILS FOR PRE-ACTIVATION BATCH NORMALIZATION CNN TRAINING FROM THE SCRATCH

Parameters	**Values**
Training Accuracy	100.00%
Validation Accuracy	99.81%
Training Status	Complete
Elapsed Time	39 min 8 sec
Number of Epoch per iteration	9
Mux. Number of Iteration	810
Validation Frequency	80

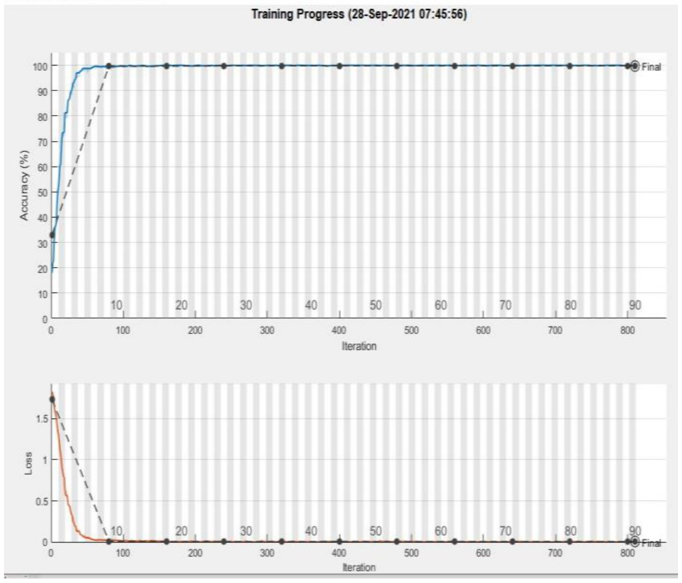


Fig. 6. Training Progress Plot Graph for Post-Activation-Batch-Normalization-CNN Model.

In the above training graph, it can be seen that the training and testing values at the initial epoch of 9 and iteration 80 have an accuracy value of 99.75% and 99.75%, respectively. Both the training and testing accuracies start to converge at epoch 18 and iteration 160 with the accuracy values of 100.00% and 99.68%. At epoch 27 and iteration 240, the training accuracy decreases to 99.75%, while the testing accuracy increases to 99.81%. At epoch 39 and iteration 350, the training accuracy reached 100.00% and the testing accuracy still maintained its accuracy value of 99.81. At the final epoch of 90 and iteration of 810, the training and testing accuracies continued to be 100.00% and 99.81%, respectively. The result shows that the testing dataset accuracy is close to that of the training dataset throughout the training process. This implies that no overfitting occurred and that the implementation of the batch normalization operation is working very well [24]. While in the lower part of the graph, which is the loss error curve, the training and testing losses, which represent the degree of differences between the model prediction and the real classes, decrease with increasing epoch number [23]. At epoch 1 and iteration 1, the training and loss errors were initialized as 1.7920 and 1.7915, respectively. While the number of epochs reaches its final stage, of epoch 90 and iteration 810, the training loss decreases to 0.0026 and the testing loss decreases to 0.0052. These show that both the training and validation loss functions have convergence at the learning rate that reaches 1.600e-05. So the network has no overfitting concerning its testing dataset since the training and testing loss values are closed.

D. Experiment with Sparse Batch Normalization CNN Architecture

The fourth model is a deep learning model with a sparse batch normalization architecture. This model was developed and used by [2]. The design principle of this deep learning model is characterized by the following:

An input layer convolutional layer 1; Convolutional layer 2, max-pooling layer, batch normalization layer; Convolutional layer 3, max pooling layer; Convolutional layer 4, max-pooling layer. It could be seen that the first convolutional layer unit has only the first convolutional layer, which has no operational layer attached to it. The batch normalization operation only takes place at the second convolution unit after the max-pooling operation. The third and fourth convolutional units have only convolutional layers with max-pooling layers each. It can be noticed that the architecture of this network does not utilize any activation function using the convolutional operation units. It only uses the soft-max function as an output function after the two fully connected layers within the output units of the network.

Table IV shows the training details and Fig. 7 shows the Sparse Batch Normalization CNN Architecture Training Progress Plot Graph.

The sparse Batch normalization model In Fig. 6, was implemented using MATLAB R2018b for the training and validation. This model runs around 270 epochs with 3 iterations per epoch and a batch size of 400. This is done to train this model with our proposed dataset very well [25]. Stochastic gradient descent and momentum term are the optimizers used for this model. The training graph in Fig. 6 above illustrates performance and classification accuracy and losses between both the training and validation phases with the number of epochs and iterations. At an initial learning rate of 0.0100, Epoch 1, and iteration 1, the training and validation accuracy values are initialized. The training and validation accuracy values reach 96.00% and 97.54%, respectively, at Epoch 107 and iteration 320.650increases %. Finally, at the last epoch of 270 and last iteration 810, the values of both the training and validation accuracies increase to 96.25% and 97.60%, respectively.

TABLE IV. TRAINING PLOT DETAILS FOR SPARSE BATCH NORMALIZATION CNN ARCHITECTURE TRAINING FROM THE SCRATCH

Parameters	**Values**
Training Accuracy	96.25%
Validation Accuracy	97.60%
Training Status	Complete
Elapsed Time	132 min 32sec
Number of Epoch per iteration	3
Mux. Number of Iteration	810
Validation Frequency	80

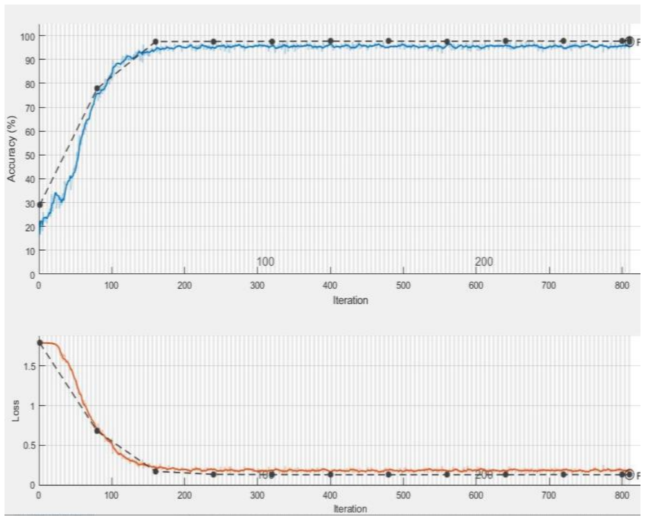


Fig. 7. Training Progress Plot Graph for Sparse Batch Normalization-CNN Architecture.

In the above training graph, the training and validation losses that represent the degree of differences between the model prediction and the real classes decrease with increasing epoch numbers [44]. At epoch1 and iteration 1, the training and validation losses were 1.7918 and 1.7913, respectively. At epoch 107 and iteration 320, the training and validation losses decrease to 0.1747 and 0.1335, respectively. While the number of epochs reaches its final stage, of epochs 270 and iteration 810, the training loss value increases to 0.1768, and the testing loss decreases to 0.1294. This result shows that the network has over fitted concerning its training and testing datasets since there are many differences between the loss values of the training and loss values of the testing dataset.

VI. COMPARISON OF THE TRAINING AND VALIDATION ACCURACIES AND LOSS ERRORS FOR DIFFERENT DEEP LEARNING CNN ARCHITECTURES

To find suitable deep learning CNN architectures for the proposed face recognition system, it is vital to recall the alleged effects of batch normalization, which can be broadly categorized as convergence speed and generalization performance improvement. In this paper, we trained and tested four different deep learning convNets with different model architectures. By comparing the experimental results of the four Fig. 9, comparison of Validation Accuracies of the deep learning ConvNet architectures, we will be able to rule out the robust ConvNet model that leads to higher classification performance and it will also rule out the type of model that leads to insignificant results. This section focuses on comparing the experimental results of the four deep learning models by observing the training and validation behavior of the different architectures [10]. Here we aim to isolate the effect of batch normalization in general, concerning our proposed Pre-Activation-Batch Normalization CNN Architecture. In this experiment, the Pre-Activation-BN-CNN-Architecture is abbreviated as PRACBNCNN, the Post-Activation-BN-CNN is presented as PSTACBNCNN, the Traditional-CNN-without-BN is denoted as TRCNNWITHOUTBN, and lastly, the Sparse-BN-CNN is represented as SPSBNCNN.

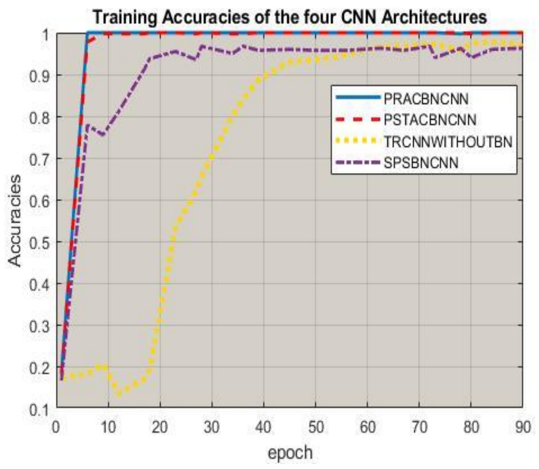


Fig. 8. Comparison of Training Accuracies of the Four CNN Architectures.

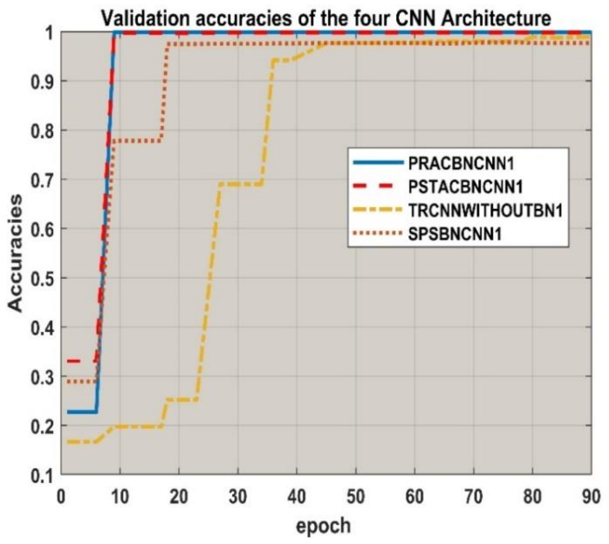


Fig. 9. Comparison of Validation Accuracies for the Four CNN Architectures.

Fig. 8 and Fig. 9 above show the training and validation accuracies overtime for all the four deep learning models. In terms of the training and validation accuracies, we can observe that both the Pre-activation-BN-CNN and Post-Activation-BN-CNN models have to reach overall accuracies in all cases by simply adding batch normalization before and after the rectifier linear unit activation function. The training and validation accuracies (solid blue lines) of the proposed Pre-active-BN-CNN follow by the Post-Activation-BN-CNN Training and validation accuracies (dotted red lines) consistently achieve higher accuracies and gained immediate convergence speed improvement on both the training and validation accuracy. While the Training and validation accuracies (zigzag dashed mustard lines) of the Traditional CNN without batch normalization architecture and that of Sparse Batch Normalization CNN (Fluctuated dotted purple line) are not ideal this is because there is a lot of overfitting. The results show that the training dataset of both TRCNNWTHOUTBN and SPSBNCNN are harder than validation datasets of the model. In the two figures the Pre-activation-BN-CNN model started to converge at epoch 10 with training and validation accuracies of 100.00% and 99.87 respectively, the Post-Activation-BN-CNN started its convergence at epoch 18 with training and validation accuracies of 100.00% and 99.68%. On the other hand, the training and validation accuracies of Traditional-CNN-without-BN start to improve at 62 with 96.50% and 97.73%, respectively. The Accuracies of Sparse increases at epoch 23 with 95.50% and 97.47%. The training and validation results in the two figures tried to show that the performance of Post-Activation-BN-CNN is about the same as the Pre-Activation-BN-CNN, but the Pre-Activation-BN-CNN outperforms all the three deep learning CNN architectures. This is supported by the higher accuracy of each data. The training data has a curacy value of 100.00% and 99.87% on the Testing data [26].

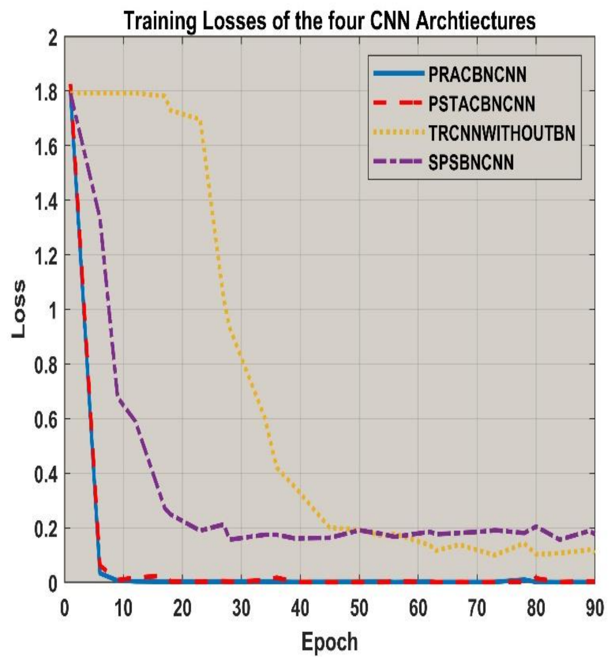


Fig. 10. Comparison of Training Losses for the Four CNN Architectures.

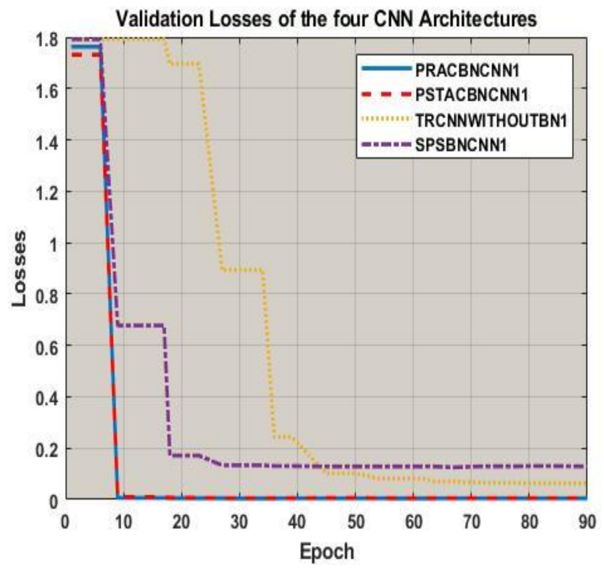


Fig. 11. Comparison for Validation Losses of the Four CNN Architectures.

The curve in Fig. 10 is the comparison of training loss error curve for the four CNN architectures, and the other graph in Fig. 11, is the validation loss error curve. Each with a different number of epochs ranging from 0 to 90. The two curves show the comparison results of the cross-entropy loss errors of the four deep learning models. By observing the losses over time of both the Pre-activation-BN-CNN- model and that of the Post-Activation-BN-CNN model we can see how the losses repeatedly fall to the lower level. In the figures, a noteworthy observation can be made at the begging of the training on the Traditional CNN without BN and Sparse BN CNN Networks. During the first phase of the training epochs training and validation losses of two models plateaus for the higher amount loss errors, until the gradient update escapes the unfavorable local minimum at epoch 72, iteration 640 with training and validation loss of 0.1059 and 0.653 for the Traditional CNN without BN.

While for the Sparse BN CNN model at epoch 84, 750 iterations with training and validation loss of 0.1565 and 0.1305 prospectively. The results show that the two models suffer from large covariate shifts, which lead to the network gradient disappearance and higher overfitting. In both the training and validation losses of the four models in the two graphs, we can see that there is separation between the models the Pre-activation BN-CNN training and validation loss errors (the solid blue lines) start to converge to the minimal error at epoch 10 with training and validation losses of 0.0067 and 0.0065. Finally, at the last epoch 90, the training and validation loss decreases to 0.0013 and 0.0048. While The Post-Activation-BN-CNN started to converges at epoch 18 and its last epoch 90 the training and validation loss decreases to 0.0026 and 0.0052 respectively. Fig. 9 and 10 illustrates the training and testing error rate of the proposed Pre-Activation Batch Normalization CNN Architecture, over 90 training Epochs, and 810 iterations. The training Errors are always lower than that produce by post Batch Normalization CNN and there is a higher split between the Pre-Activation-BN-CNN and the Traditional CNN without BN Architecture. In terms of the training and validation performance, Post-Activation Batch Normalization CNN architecture is about the same as Pre-Activation Batch Normalization Architecture, but Pre-Activation BN CNN Architecture has outperformed all of the other deep learning models. This can be seen in the two figures. The resulting training and validation loss values obtained from the Pre-Activation-BN-CNN are 0.0013 and 0.0048 when compare with the training and validation loss values of the other three models are very small. So that these values can be said to be quite low. This implies that the resulting model can be said to be able to classify well because it has lower loss error values and high accuracies values. This can be illustrated in Table V.

In the table, the Training and Testing of Different Deep Learning Models after 810 iterations, as in, the experiment results of the four deep learning CNN models are given. Some of the models have batch normalization layers and some without.

TABLE V. PERFORMANCE COMPARISON OF THE (FOUR) DIFFERENT DEEP LEARNING MODELS WITH AND WITHOUT BATCH NORMALIZATION LAYERS

Deep Learning Models	**Face Recognition Accuracies Rate**		
	Model Type	***Training Accuracy***	***Validation Accuracy***
Model 1	PRACBNCNN	100.00%	99.87%
Model 2	PSTACBNCNN	100.00%	99.81%
Model 3	TRCNNWITHUTBN	96.50%	98.93%
Model 4	SPSBNCNN	96.25%	97.60%

The batch normalization techniques give a classification improvement. The Pre-Activation-BN-CNN model performed better than the other three models, with training and validation accuracies of 100.00% and 99.87 as shown in the table. This is because the model architecture encompasses a batch normalization layer in each of the four convolutional units, which is placed before the Rectifier linear unit (ReLu) activation function. According to the training and validation results in the table, the Post-Activation is the next model with better training and validation accuracies of 100.00% and 99.81%. This model has a batch normalization layer in each of its four convolutional units after the rectified linear unit (ReLu) activation function. On the other hand, the table result shows that the traditional CNN has training and validation accuracies of 96.50% and 98.93%. This model has no batch normalization techniques in any of its four convolutional units, but it has the rectified linear unit in each of the convolution units. That is why its accuracy results outperformed those of the Sparse BN CNN model, which has the training and validation accuracies of 96.25% and 97.60% as shown in the table. This model has only one batch normalization layer at the second convolutional unit. The model has no rectifier linear unit (ReLu) activation function.

VII. PERFORMANCE EVALUATION MATRICS

It is critical to define performance measures that are appropriate for the job at hand when evaluating the performance of deep learning models. In this study, we proposed the most critical performance indicators for accuracy, precision, f-score, and recall, as given in the equations below, to analyze our results and to demonstrate that the above results explained in the preceding section are correct [26].

$$\text{Precision} = \frac{TP}{TP+FP} \quad (1)$$

$$\text{Recall} = \frac{TP}{TP+FP} \quad (2)$$

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \quad (3)$$

$$F_1\text{-Score} = \frac{2(Precision)*(Recall)}{Precision+Recall} \quad (4)$$

TABLE VI. COMPARISON OF THE RESULTS FROM THE PERFORMANCE EVALUATION MATRICS

Deep Learning Models	Performance Evaluation				
	Model Type	**Accuracy**	**Precision**	**Recall**	**F_1-Score**
Model 1	PRACBNCNN	0.99873	0.9987	1.00	0.9993
Model 2	PSTACBNCNN	0.9981	0.9867	1.00	0.9990
Model 3	TRCNNWITHUTB	0.98926	0.9892	0.994	0.9932
Model 4	SPSBNCNN	0.97601	0.97595	0.9899	0.9831

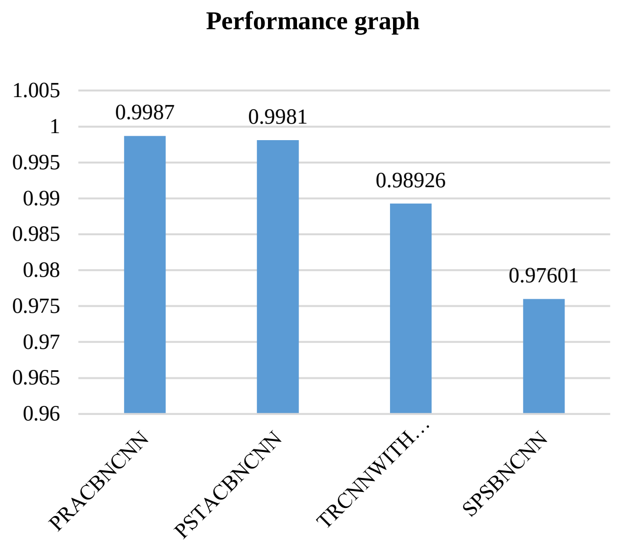


Fig. 12. Performance Evaluation Results.

As we mentioned in the above sections, this research aim to enhance the performance of the face recognition system, using a deep learning approach that involves a convolutional neural network (CNN). In order to evaluate the robustness of our proposed model we compared it result with the results of three other deep learning models. In this section in Table VI, we again compare the results in terms of Accuracy, Precision, Recall and F_1-Score to show the correctness of the previous explanations of our study results. These results were obtained after we evaluated the result obtained from the confusion matrices produce by each of the four models.

When observing the results in the table it can be seen that the PRACBNCNN model 1 is having the high accuracy of 0.99873 and precision of 0.9987. This goes along with the previous result in Table V, where the model has 99.87% validation accuracy. These results are represented in Fig. 12.

VIII. SUMMARY AND CONCLUSION

This research paper presented four deep convolutional neural network models. The architectural styles of the models were divided into two categories: The first architectures were deep learning CNN architectures with batch normalization techniques in each of the model convolutional units, and the second deep learning architecture was deep learning models without batch normalization techniques. In this research, the best deep learning CNN architecture for recognizing human face images was obtained from the Pre-Activation-BN-CNN Architecture. This model is powered by a batch normalization layer at each of its four convolutional units. The batch normalization layers are all placed before the rectified linear units (Relu) activation function. The result of this research shows that placing the Batch Normalization layer before the ReLu activation function improved network classification power, as the model training and validation results showed 100.00% and 99.87%, respectively. This shows the regularization effect of the Pre-Activation-BN-CNN model over the other three CNN architectures for face recognition systems. In this research work, the application of the Pre-Activation-BN-CNN Architecture has enhanced the performance of the face recognition system. Therefore, reducing covariate shift prevents gradient disappearance and gradient exploration. This leads to better network convergence. The experiment results also show that having the rectified linear unit (Relu) activation function in a model architecture stabilizes network training and improves model classification. This is justified by comparing the results of the TRCNNWITHOUTBN, which has no batch normalization layer but has rectifier linear units, and the results of the SPSBNCNN, which has only one batch normalization layer at the second convolutional unit but has no rectifier linear activation unit. The training and validation accuracy of TRCNNWITHOUTBN are 96.50% and 98.93%, while that of SPSBNCNN is 96.25% and 97.60%, respectively.

IX. ACKNOWLEDGMENT

We would like to show our appreciation to the University Sultan Zainal Abidin, more especially the faculty of Informatics and computing, as well as the UNISZA Research Management Center (CRIEM) for their support on this research work financially and academically.

REFERENCES

[1] S. D. Abu and F. S. Mohamad, “Approaches Of Deep Learning In Persuading The Contemporary Society For The Adoption Of New Trend Of AI Systems: A Review,” Researchgate.Net, vol. 9, no. 12, pp. 163–177, 2020, [Online]. Available: https://www.researchgate.net/profile/Sanusi_Darma_Abu/publication/347784009_Approaches_Of_Deep_Learning_In_Persuading_The_Contemporary_Society_For_The_Adoption_Of_New_Trend_Of_AI_Systems_A_Review/links/5fe3d7aca6fdccdcb8f71f6d/Approaches-Of-Deep-Learning-.

[2] J. Cai, O. Chang, X. L. Tang, C. Xue, and C. Wei, “Facial Expression Recognition Method Based on Sparse Batch Normalization CNN,” Chinese Control Conf. CCC, vol. 2018-July, pp. 9608–9613, 2018, DOI: 10.23919/ChiCC.2018.8483567.

[3] Y. D. Coşkun, Musab, Ayşegül Uçar, Özal Yıldırım, “Face Recognition Based on Convolutional Neural Network,” in International Conference of modern Electrical Energy System, 2017, no. January 2018, pp. 1–5, DOI: 10.1109/MEES.2017.8248937.

[4] L. Shen and L. Bai, “A review on Gabor wavelets for face recognition,” Pattern Anal. Appl., vol. 9, no. 2–3, pp. 273–292, 2006, DOI: 10.1007/s10044-006-0033-y.

[5] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, “Eigenfaces vs. fisher face: Recognition using class specific linear projection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 19, no. 7, pp. 711–720, 1997, DOI: 10.1109/34.598228.

[6] S. A. Darma, F. Aliyu, and A. Kurfi, “The Role of Social Media in

Empowering the Involvement of Women in Information Technology : A Case Study of Al-Qalam and U maru Musa Yar ' Adua Universities," vol. 2, no. 1, 2018.

[7] D. H. H. A. T. N. WIESEL, "RECEPTIVE FIELDS AND FUNCTIONAL ARCHITECTURE OF MONKEY STRIATE CORTEX," J. Physiol., vol. 195, no. 1, pp. 215–243, 2017.

[8] S. I. C. Szegedy, "Australian Literary Journalism and 'Missing Voices': How Helen Garner finally resolves this recurring ethical tension," Journal. Pract., vol. 10, no. 6, pp. 730–743, 2016, DOI: 10.1080/17512786.2015.1058180.

[9] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conf. Comput. Vis. Pattern Recognit., no. May 2014, pp. 248–255, 2009, DOI: 10.1109/CVPRW.2009.5206848.

[10] FABIAN SCHILLING, "The Effect of Batch Normalization on Deep Convolutional Neural Networks," 2016. [Online]. Available: http://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A955562&dswid=-5716.

[11] N. Gajhede, O. Beck, and H. Purwins, "Convolutional neural networks with batch normalization for classifying hi-hat, snare, and bass percussion sound samples," ACM Int. Conf. Proceeding Ser., vol. 04-06-Octo, no. October 2016, pp. 111–115, 2016, DOI: 10.1145/2986416.2986453.

[12] M. Bojarski et al., "End to End Learning for Self-Driving Cars," 2016, pp. 1–9.

[13] K. B. Pranav and J. Manikandan, "Design and Evaluation of a Real-Time Face Recognition System using Convolutional Neural Networks," Procedia Comput. Sci., vol. 171, no. 2019, pp. 1651–1659, 2020, DOI: 10.1016/j.procs.2020.04.177.

[14] S. Zheng, R. W. O. Rahmat, F. Khalid, and N. A. Nasharuddin, "3D texture-based face recognition system using fine-tuned deep residual networks," PeerJ Comput. Sci., vol. 2019, pp. 1–21, 2019, DOI: 10.7717/PEERJ-CS.236.

[15] P. Kamencay, M. Benco, T. Mizdos, and R. Radiol, "A New Method for Face Recognition Using Convolutional Neural Network Face Recognition System – State of the Art," pp. 663–672, 2017, DOI: 10.15598/area.v15i4.2389.

[16] O. A. H. Hayder Najm, Hayder Asaf, "An N EFFECTIVE IMPLEMENTATION OF F ACE R RECOGNITION USING," 2019.

[17] M. K. Benkaddour and A. Bounoua, "Feature extraction and classification using deep convolutional neural networks, PCA and SVC for face recognition," Trait. du Signal, vol. 34, no. 1–2, pp. 77–91, 2017, DOI: 10.3166/TS.34.77-91.

[18] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent Neural Network Regularization," Conf. Pap., no. 2013, pp. 1–8, 2015, [Online]. Available: http://arxiv.org/abs/1409.2329.

[19] Y. Wu and K. He, "Group Normalization – Facebook Research," Eccv, no. Figure 1, 2018, [Online]. Available: https://research.fb.com/publications/group-normalization/.

[20] X.-Y. Zhou et al., "Batch Group Normalization," 2020, [Online]. Available: http://arxiv.org/abs/2012.02782.

[21] R. Lemlich, "Foam fractionation and allied techniques," Ind. Eng. Chem. Res, vol. 60, no. 10, pp. 16–29, 2015, [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Foam+Fractionation+and+Allied+Techniques#1.

[22] B. Eritzke, "A self-organizing network that can follow non-stationary distributions," Lect. Notes Comput. Sci. (including Subsea. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 1327, pp. 613–618, 1997, DOI: 10.1007/bfb0020222.

[23] M. Shyu, S. Chen, and S. S. Iyengar, "A Survey on Deep Learning : Algorithms, Techniques," vol. 51, no. 5, 2018.

[24] D. M. Ibrahim, N. M. Elshennawy, and A. M. Sarhan, "Deep-chest : Multi-classification deep learning model for diagnosing COVID-19, pneumonia, and lung cancer chest diseases," Comput. Biol. Med., vol. 132, p. 104348, 2021, DOI: 10.1016/j.compbiomed.2021.104348.

[25] J. Yang and G. Yang, "Modified convolutional neural network based on the dropout and the stochastic gradient descent optimizer," Algorithms, vol. 11, no. 3, pp. 1–15, 2018, DOI: 10.3390/a11030028.

[26] J. Alsamiri and K. Alsubhi, "Internet of things cyber attacks detection using machine learning," Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 12, pp. 627–634, 2019, doi: 10.14569/ijacsa.2019.0101280.