Engineering Applications of Artificial Intelligence 85 (2019) 579–589

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

On-device facial verification using NUF-Net model of deep learning[☆]

Chakkrit Termritthikun[*], Yeshi Jamtsho, Paisarn Muneesawang

Department of Electrical and Computer Engineering, Faculty of Engineering, Naresuan University, Phitsanulok, 65000, Thailand

ARTICLE INFO

Keywords:
Deep learning
Face detection
Face verification
Convolutional Neural Networks
Face recognition
NUF-Net

ABSTRACT

The major problems for Convolutional Neural Networks (CNNs) on devices with low processing power, such as Raspberry Pi 3, are the processing time and the model storage space. This paper presents the development of the NUF-Net to be an efficient CNNs for on-device processing for face detection and verification systems. The development of the NUF-Net architecture was focused on reducing the number of parameters, the size of the network and the processing time, to ensure suitability on small devices. The NUF-Net demonstrated an accuracy of 99.13% for the LFW dataset with 7.1 million parameters, which is approximately 25% of that of the Inception-ResNet-V1 model. When the NUF-Net with-Residual technique was applied to an Automatic Door (AutoDoor) as a replacement for the Fingerprint Door Lock, the NUF-Net-with-Residual was able to perform detection and confirmation of faces on a Raspberry Pi 3 within 1–1.5 s, providing a recognition accuracy of 91.18% with a sample of 68 people, which is close to the Inception-ResNet-V1, with multiply-accumulation operations equal to 10.12 million FLOPs which is about four times faster than the Inception-ResNet-V1 (46.86 million FLOPs).

1. Introduction

Deep learning (DL) (Voulodimos et al., 2018; Nie et al., 2018; Goodfellow et al., 2016; LeCun et al., 2015; Schmidhuber, 2015; Zhang et al., 2018; Han et al., 2018; Macaluso and Shih, 2018) is one of the most popular artificial intelligence (AI) models because of its capacity to more accurately analyze large amounts of data than humans, by applying sophisticated learning techniques and in-depth analysis. Convolutional neural networks (CNNs) (Chen et al., 2017; Nogueira et al., 2017; Termritthikun et al., 2017; Termritthikun and Kanprachar, 2017, 2018; Han et al., 2017; Cheng et al., 2019) are one of the most popular algorithms in the field of DL, as CNNs include models, layers, optimizers and pre-processing, each of which functions differently. However, each of the CNNs has been developed and researched in a variety of ways, enabling research into CNNs to continually advance, with many models having been discovered that show increased performance from standard image datasets such as CIFAR-10, CIFAR-100 (Krizhevsky and Hinton, 2009).

Research into face verification is gaining popularity which has also promoted interest in CNNs. The face verification process consists of three parts: First, a face dataset is selected from amongst many datasets. In training, the chosen dataset should include many images, which can improve the training process. CASIA-WebFace (Yi et al., 2014) and LFW (Learned-Miller et al., 2016) are the most popular datasets used in testing accuracy. The second part is face detection, which detects faces from images. The detection accuracy of the CNNs is improved by using the Haar-like feature, which has been popular in the past. Face Recognition is a part of personalized image mapping with CNNs, and FaceNet is one of the ways that CNNs are trained with a selected face dataset. Also, FaceNet can be adapted to other models. Currently, Inception-ResNet-V1 (Szegedy et al., 2017) is the most effective model of FaceNet.

Improving accuracy is the main focus in Face verification. Improvements of up to 99% have been achieved with LFW dataset by adding new techniques. In the CNNs model, the size or number of parameters is increased. The most popular model, the Inception-ResNet-V1, uses a high number of parameters which requires substantial processing time. This has resulted in the development of the SqueezeNet (Iandola et al., 2016) model in which the number of parameters is reduced to a size that is small enough to be deployed on small storage devices and processors. This was achieved but performance accuracy was reduced. Consequently, Google introduced MobileNets (Howard et al., 2017), which are based on the same concept as SqueezeNet. The MobileNet leveraged the Depthwise Separable Convolution instead of the Standard Convolution because the Depthwise Separable Convolution can eliminate many multiply-accumulation operations and also enhances the accuracy of the models that are built from the Standard Convolution.

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.engappai.2019.07.018.

* Corresponding author.
E-mail addresses: chakkritt60@email.nu.ac.th (C. Termritthikun), yeshij61@email.nu.ac.th (Y. Jamtsho), paisarnmu@nu.ac.th (P. Muneesawang).

https://doi.org/10.1016/j.engappai.2019.07.018
Received 4 October 2018; Received in revised form 7 March 2019; Accepted 23 July 2019
Available online 29 July 2019

The Raspberry Pi is an embedded "computer-on-a-board" and is the size of a credit card, is cheap to buy, with Internet connectivity. It is, therefore, a very popular device for use in the Internet of Things (IoT). The specifications of the Raspberry Pi 3 Model B+ include a Broadcom BCM2837B0 Quad-Core ARM Cortex-A53 (ARMv8) CPU with 1 GB SDRAM running on the Raspbian 9.0 operating system. The Raspberry Pi can be applied to complicated CNNs models, such as the Inception-ResNet-V1 model (Szegedy et al., 2017) but its low-speed processor and small memory result in slow CNNs processing.

To deploy face-detection and recognition systems on the Raspberry Pi 3 device, or a similarly configured device, a CNNs model with similar or the same properties as SqueezeNet is required. In this paper, we discuss the development of the Naresuan University and Fiber One Network (NUF-Net) model that features a small number of parameters, commensurate with the SqueezeNet model, yet still provides precise accuracy for face detection and recognition systems. In Section 2, we review related work on face recognition. Section 3 presents the development of the proposed NUF-Net method, and Section 4 presents the experimental results. Conclusions are drawn in Section 5.

2. Related work

Face detection and face recognition are the two essential aspects of face verification. Deep learning has been shown to be a popular learning algorithm. We reviewed the use of a large-scale dataset and face verification processes which are now extensively used in the image processing.

2.1. Face dataset

Datasets for face recognition are available in LFW (Learned-Miller et al., 2016), WDRef (Chen et al., 2012), CelebFaces (Sun et al., 2014), SFC (Taigman et al., 2014), CACD (Chen et al., 2014), CASIA-WebFace (Yi et al., 2014) and VGGFace2 (Cao et al., 2018). In this paper, we discuss the datasets that we used as training datasets and others that were used as test datasets. The training datasets were sourced from CASIA-WebFace and VGGFace2 while only the LFW datasets, popularly used for benchmark face recognition accuracy, were used for testing.

In the CASIA-WebFace dataset, there are 453,453 photos of 10,575 people, with 3.31 million photos of 9131 people in VGGFace2 dataset, and the LFW dataset has 13,233 photos of 5749 people (see Fig. 1).

2.2. Face detection

Face detection is a system for detecting faces in the given image. The most widely used detection algorithm is Viola–Jones object detection (Viola and Jones, 2001), also known as the Haar-like feature, developed on popular libraries such as the OpenCV library. However, the Viola–Jones object detection process needs a full face on an image, and cannot detect the face accurately if the face is tilted to one side. Further, it generates an error in detection if there are other objects in the image that match the characteristics of the Haar detection objects, such as a human nose or human eye.

The Multi-task Cascaded Convolutional Network (MTCNN) (Zhang et al., 2016) is a face detection framework similar to Viola–Jones object detection and uses CNNs as a core algorithm for face detection and recognition. The framework includes three CNNs stages where the bounding boxes of faces in an image along with five-point face landmarks (left eye, right eye, nose, left mouth corner, and right mouth corner) are detected. Each stage passes its input through CNNs, improving detection results and returning candidate bounding boxes with their scores, followed by non-max suppression (see Fig. 2).

The first CNNs stage is the Proposal Network (P-Net) responsible for finding bounding box regression vectors that resemble the human face. In stage 1, an image pyramid is built from an input image by scaling down multiple times and each scaled version of the image is passed through its CNNs.

The second stage is the Refine Network (R-Net), which filters and applies the bounding box regression with other overlapping areas using non-maximum suppression (NMS). In stage 2, image patches for each bounding box are extracted, then resized (24×24) and forwarded through its CNNs.

Finally, the Output Network (O-Net), which is similar to R-Net but with a major difference in that the O-Net searches for facial landmarks from bounding box regression making it possible for the three networks; P-Net, O-Net, and R-Net, to accurately detect a human face. In stage 3, image patches for each bounding box are extracted, then resized (48×48) and forwarded through the CNNs of that stage. In addition to bounding boxes and scores, 5 face landmarks points for each bounding box are computed in this stage.

2.3. Face recognition

FaceNet (Schroff et al., 2015) is a popular library of facial images, where FaceNet uses CNNs. Inception-ResNet-V1 (Szegedy et al., 2017) has an embedding size of only 128-bytes per face. The structure of the model includes the Inception module (Szegedy et al., 2016, 2015; Ioffe and Szegedy, 2015) of Google and Residual Connections (He et al., 2016) of Microsoft. By combining these two techniques, the model was found to be highly efficient. The Inception module can accurately detect objects from images by using a convolution of a variety of kernels and the results of each kernel are output together to confirm the location of the object. Residual connections will preserve the data from residual representations and establish a connection between the previous layer and the current layer. In the experiment, the Inception-ResNet-V1 was trained with the CASIA-WebFace dataset and tested with the LFW dataset. The results achieved showed that the Inception-ResNet-V1 attained 98.83% accuracy.

In our work, the process of developing the face verification system was divided into three parts: the first was training the model with a large amount of standardized dataset, which are the important components for training the model. The second part was face detection, which is as equally important as training the model with a dataset since detecting the wrong face and sending the information to the recognition phase can cause a classification error. Face recognition is the final stage of development of the facial recognition system and the process leverages on the result of the face detection stage. These stages are the most appropriate steps in refining or modifying the CNNs' structure.

The NUF-Net face verification model that we developed is for devices that have limitations in processing time and storage space, by refining the model to replace the previously designed models. The description of the NUF-Net model is presented in the next section.

3. Method

The NUF-Net is a deep convolutional network that was evolved from the SqueezeNet model. The SqueezeNet model has a small storage size requirement, hence it has been adopted for face verification processing on devices with limited storage space. However, the SqueezeNet model has poor performance accuracy making it essential to improve on that model for accurate face recognition. In our development, this was achieved by refining the structure by adding Inception modules and Residual connections to the SqueezeNet model, thereby providing improved accuracy while at the same time maintaining the size of the CNNs model in the NUF-Net model to ensure that it is of a suitable size for a device with a small storage capacity, such as the Raspberry Pi 3 or the NVIDIA Jetson TX2.

Fig. 1. Sample of 18 people from 10,575 people on CASIA-WebFace dataset.

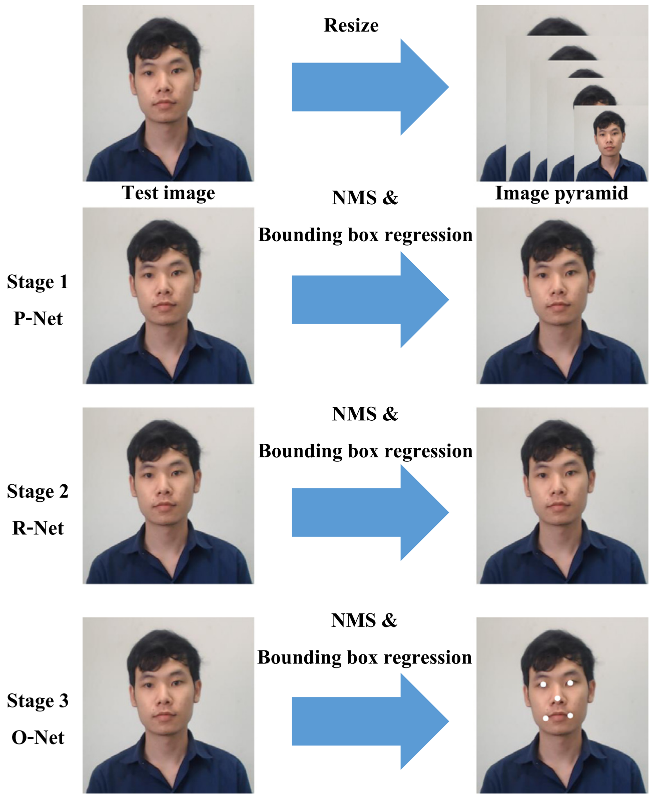

Fig. 2. Multi-task Cascaded Convolutional Networks (MTCNN).

3.1. *Facial verification system*

The process of developing the facial verification system on the Raspberry Pi 3 device begins with loading the dataset containing the facial features of all the people who have been registered in the system. The Raspberry Pi 3 then connects to the camera and reads the data frame by frame to search for the required facial image, using the MTCNN algorithm. When the required facial image is found it is forwarded to the feature extraction process with NUF-Net.

The feature vector is then compared to the individual feature vector that was previously recorded. The controlling device (in this case the automatic door lock) sets the threshold value to 90%. If the distance between the two feature vectors is less than the threshold, it is necessary to continue the frame search. However, if the distance is higher than the threshold, indicating that the images are similar, the controlling device will automatically open the door. Also, a voice command is given, such as "Hello Alex", to alert the user (i.e. the person attempting entry). The system also immediately sends an email to the system administrator to notify that the door has been unlocked, and the identity of the person entering, and this data is also logged in the system. The overall workflow is shown in Fig. 3 and the flowchart of the system is shown in Fig. 4.

3.2. *Deep convolutional neural network (DCNNs)*

Deep CNNs (DCNNs) consists of many connected neural network convolutional layers or pooling layers. DCNNs structures have many overlapping layers which increase the size of the features, including a fully connected layer. There are many models available to be selected: AlexNet (Krizhevsky et al., 2012), GoogLeNet (Szegedy et al., 2015),

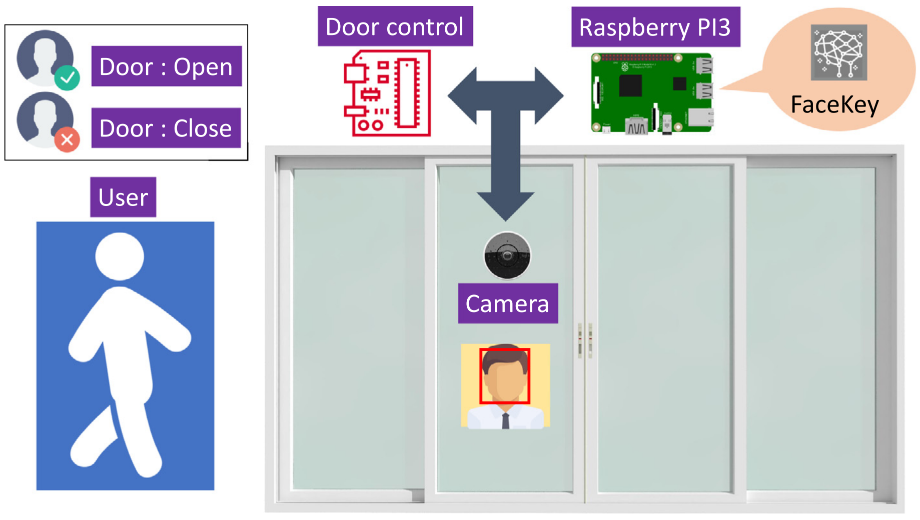

Fig. 3. Facial verification system.

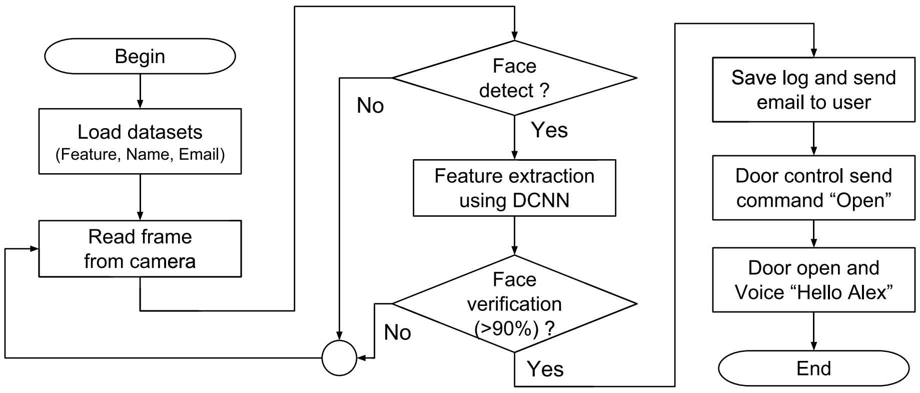
Begin
Load datasets (Feature, Name, Email)
Read frame from camera
Face detect ?
No
Yes
Feature extraction using DCNN
Face verification (>90%) ?
No
Yes
Save log and send email to user
Door control send command "Open"
Door open and Voice "Hello Alex"
End

Fig. 4. The Flowchart of the facial verification system.

ResNet (He et al., 2016), and SENet (Hu et al., 2018), all of which have won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) at one time or another between 2012–2017.

Let us, first, introduce the following equation for the explanation.

$$y = F(x, \{W_i\}) + x \tag{1}$$

Residual Networks, also known as ResNets, are popular DCNNs models, due to the fact that ResNet offers a shortcut connection, known as identity mapping, which helps to preserve data from previous layers. By identifying the variable x as the connection between the previous layer and the current layer, connected through the convolution layer, the output x is the value from the current layer, and is combined with the output from the other three convolution layers to obtain $y = F(x, \{W_i\}) + x$. This process and the use of x is illustrated in Fig. 5. The addition of Identity mapping to the model enhances accuracy. Based on the ILSVRC competition in 2015, it has been shown that plain models, which have 18 layers and do not have Identity mapping, had a higher accuracy than models of 34-layers depth. However, when adding Identity mapping to the ResNet 34-layer model, higher accuracy than the 18-layer plain models was achieved.

We adopted NU-LiteNet, which was previously presented in (Termritthikun et al., 2018), to developed NUF-Net with more depth than NU-LiteNet, by using DCNNs, as shown in Fig. 6 (left). Our purpose was to improve the recognition accuracy by converting the convolution layers with 5 × 5 filters into one convolution layer with two-layered 3 × 3 filters, and also by converting the convolution layers with 7 × 7 filters into three convolution layers of 3 × 3 filters, as shown in Fig. 6 (center).

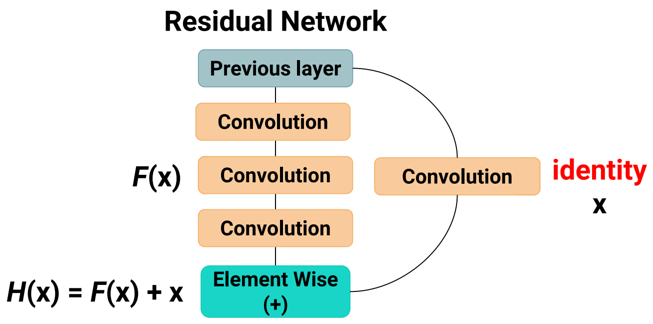

Fig. 5. Residual network.

Changing the size of the filters reduces the number of parameters in the model, while the accuracy remains equal to accuracy achieved in the Inception-v3 and Inception-v4 models. In addition, we added the Identity mapping technique (similarly as in ResNet) to the model as shown in Fig. 6 (right).

$$Conv(N, F, P) = ((N + P) - F/stride) + 1 \tag{2}$$

Eq. (2) helps to determine the size of the data that is imported into the convolutional layers where N is the size of the input, F is the size of the kernel and P is the size of the zero-padding. For example, where the input is equal to [32 × 32 × 3], the kernel size in the convolution layers is [5 × 5 × 3], zero-padding is 0, and stride is 1, from Eq. (2)

Table 1
This table shows the structure of NUF-Net.

Layer	Patch size (Stride-Pad)	Output size
Input Image	–	160 × 160 × 3
Convolution	7 × 7 (2-Same)	80 × 80 × 64
Max Pooling	3 × 3 (2-Valid)	39 × 39 × 64
NUF-Block-1,2	As in Fig. 6	39 × 39 × 64
NUF-Block-3	As in Fig. 6	39 × 39 × 128
Max Pooling	3 × 3 (2-Valid)	19 × 19 × 128
NUF-Block-4,5	As in Fig. 6	19 × 19 × 128
NUF-Block-6	As in Fig. 6	19 × 19 × 256
Max Pooling	3 × 3 (2-Valid)	9 × 9 × 256
NUF-Block-7	As in Fig. 6	9 × 9 × 256
NUF-Block-8	As in Fig. 6	9 × 9 × 512
Convolution	1 × 1 (1-Same)	9 × 9 × 1000
Average Pooling	–	1 × 1 × 1000
Fully Connected	–	128 or 512

we get $(((32+0)-5)/1)+1 = 28$ or $[28 \times 28 \times 3]$ pixels. With kernel size $[7 \times 7 \times 3]$ and zero-padding equal to 2, and stride 1, we still get $(((32+2)-7)/1)+1 = 28$ or $[28 \times 28 \times 3]$ pixels.

From the above, it was discovered that it is necessary to add two zero-padding pixels when the kernel size is increased from 5 × 5 to 7 × 7. The convolutional kernel size must be equal to 1, 3, 5, or 7. The addition of padding to the above data changes the kernel from 5 × 5 to 7 × 7, so zero-padding must be increased to $7 - 5 = 2$ pixels. Also, the determination of any parameters needs to be taken into account when considering other parameters of the convolutional layers, to prevent increasing the size of the data being modeled.

$$NU - LiteNet = Concat(Conv(N,1,0), Conv(N,3,2), Conv(N,5,4), Conv(N,7,6)) \tag{3}$$

For the NU-LiteNet model, the structure of the model was extended to include 5 × 5 and 7 × 7 convolution layers, as in (Termritthikun et al., 2018), to enhance the accuracy of the model in detecting objects using the kernel. There are a variety of sizes, as described in Eq. (3). Different outputs are obtained when N data is fed into the different layers with different kernel sizes: 1 × 1, 3 × 3, 5 × 5, and 7 × 7, and the concatenation of these data from the different layers shows that the NU-LiteNet model is better than SqueezeNet with 1 × 1 and 3 × 3 kernels, and the NU-LiteNet model provides better accuracy than the SqueezeNet model. However, both the processing time and the number of parameters are increased.

The NU-LiteNet model, therefore, had to be modified by converting the 5 × 5 into two 3 × 3 layers and the 7 × 7 convolution layer into three 3 × 3 layers to reduce both the processing time and the number of parameters (as described in Eq. (4)).

$$NUF - Net = Concat(Conv(N,1,0), 1 \times Conv(N,3,2), 2 \times Conv(N,3,2), 3 \times Conv(N,3,2)) \tag{4}$$

The structure of the complete NUF-Net model that was used in our experiments is shown in Fig. 7. For the first layer, a convolution layer with 7 × 7 kernel, was used to reduce the size of the image to 80 × 80 pixels, and then we used the max pooling layer with a 3 × 3 kernel to reduce the image size by using the max function to shrink the image to 39 × 39 pixels. Further, the NUF-Net block with no residual was used. The NUF-Net block can be applied to image resolutions of 39 × 39, 19 × 19 and 9 × 9 pixels. The 39 × 39 and 19 × 19 pixel resolution use NUF-Net of 3 blocks and 9 × 9 pixels uses 2 blocks. Each image uses a 3 × 3 max pooling layer to reduce the image size by half while increasing the number of dimensions or filters. Hence, the convolutional layer of the 1 × 1 kernel is used to increase the number of filters from 512 to 1000. An average pooling was used to reduce the image size from 9 × 9 pixels to 1 × 1 by finding the average of the areas. From a 9 × 9 pixel image, a fully connected layer to adjust the size of the data to 128 or 512 (based on the experiment) was to be used in the feature extraction process, as shown in Table 1. In the NUF-Net model, the patch size column, in Table 1, shows the size of the kernel consisting of rows × cols. In the patch size column the first number is the number of strides and the second number is the padding format which is either “same” or “valid”. The third column of Table 1 shows the output size, which is formatted as rows × cols × filters.

The simulation model is shown in Fig. 7, and the Inception-ResNet model depth and complexity are up to 43 blocks. While the SqueezeNet is a model of simplicity, the NUF-Net models, both non-residual-based and residual-based, are designed to be a less-depth network, although with more complex convolution layers in the kernel in NUF-Net block. The addition of the convolution layer to the image scale in the NUF-Net block resulted in increased both the processing time and the number of parameters. Therefore, the NUF-Net was designed to have a lower number of blocks in each depth of the image than Inception-ResNet. As a result of this, NUF-Net has the faster processing power, close to that of SqueezeNet, but has better performance accuracy than SqueezeNet.

3.3. Face representation

Face representation is the creation of data to represent the face image for easy recognition or clustering. The data will be in the form of 128-bytes for each of the 3 views [3 × 1 × 128D], with support for the left, center and right views of the face image.

Face representation consists of three stages. Stages 1 is face detection, facial recognition and reduction of the input image size to 182 × 182 pixels, using MTCNN. In stages 2, the features are extracted from the face image, transforming it into a vector form, such as [1 × 128D], by the NUF-Net model, and the vectors are then normalized using L2 for the values between −1 and 1. The final stages 3 brings the left, center and right vectors of each view together to create a face representation. In each view, the data [3 × 1 × 128D] that represents 1 person is put together, as shown in Fig. 7, and the vectors are compared using Eq. (5).

$$L_2(x,y) = \left(\sum_{i=1}^{k} |x_i - y_i|^2 \right)^{\frac{1}{2}} = \sqrt{\sum_{i=1}^{k} |x_i - y_i|^2} \tag{5}$$

Eq. (5) defines vector data and the size of the vector. In this paper, L2 distance is used to determine similar vector data from the dataset. The vector data from the feature extraction process is represented in Eq. (5), which will be able to determine how the vector data from the feature extraction (x) function is similar to the vector data from the dataset.

The data of any person consists of facial vectors, which are front, left, and right views of the face. For our purposes, at least 15 images for each view must be recorded and the system then extracts the features for transforming the face images into feature vectors, as in Fig. 8, with each face image being equal to vectors of [1 × 128D]. Thus, each person's data is composed of vectors of the face in the center view [N × 128D], in the left view [M × 128D], and in the right view [T × 128D] as shown in Eq. (6), where N is the number of faces in the center view, M is the number of faces in the left view, and T is the number of faces in the right view.

$$X_{position} = \begin{bmatrix} X_{center} \\ X_{left} \\ X_{right} \end{bmatrix} \tag{6}$$

Then, using the output of Eq. (6), we calculated the mean of all vectors using Eq. (7). For each view, only one vector of [1 × 128D] was used as a representative vector for searching and comparing vectors with L2 distance.

$$\overline{X}_{center}, \overline{X}_{left}, \overline{X}_{right} = \frac{\sum_{i=1}^{N} C_i}{N}, \frac{\sum_{j=1}^{M} L_j}{M}, \frac{\sum_{k=1}^{T} R_k}{T} \tag{7}$$

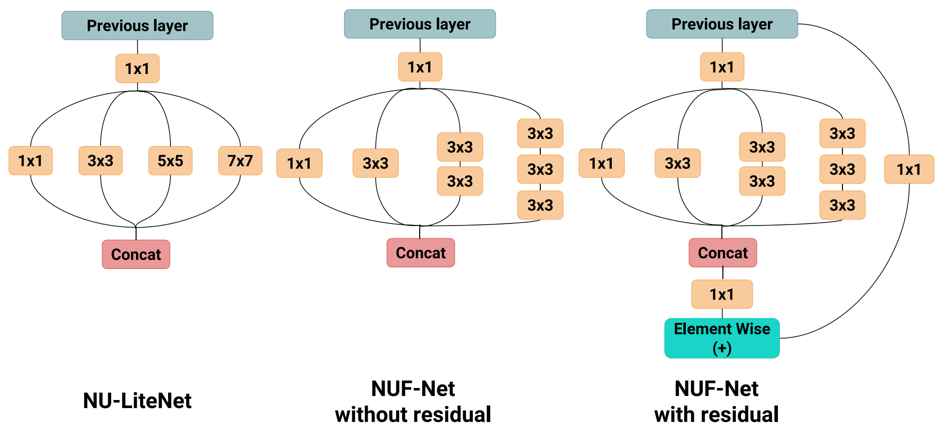

Fig. 6. NUF-Net and NUF-Net with residual modules. The middle figure is called NUF-Net block.

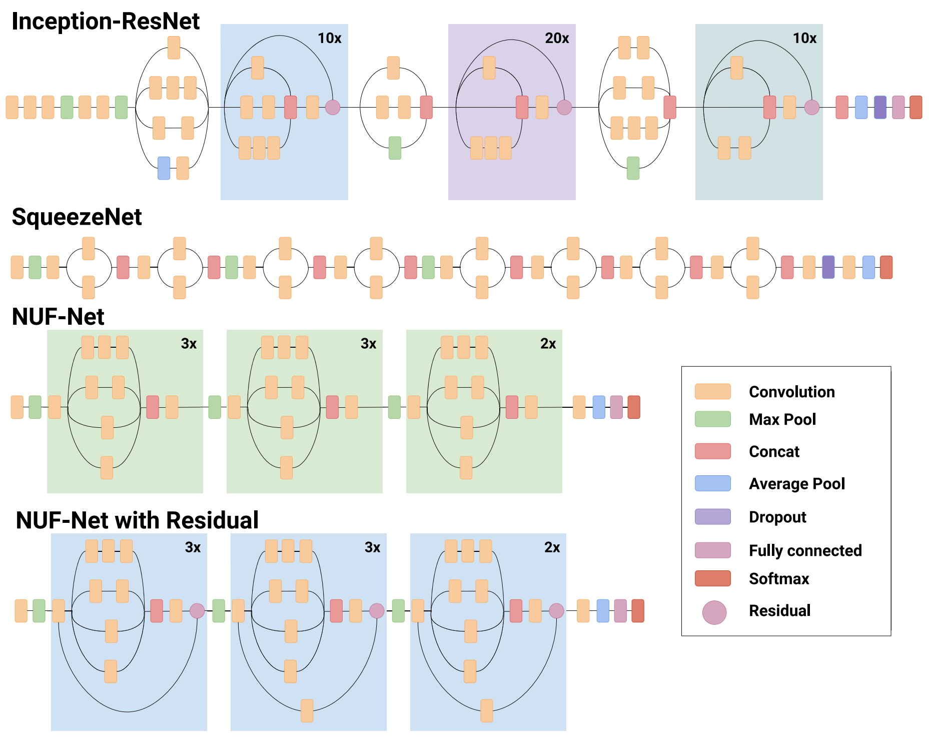

Fig. 7. Architectures.

where C represents the vectors of the central view, L represents the vectors of the left view, and R represents the vectors of the right view, N is the number of faces in the center view, M is the number of faces in the left view, and T is the number of faces in the right view.

4. Result

For the experiment, NUF-Net was designed to train neural networks on CASIA-WebFace and VGGFace2 datasets, using TensorFlow in the training process. When using the LFW dataset, performance accuracy and the number of parameters for Inception-ResNet-V1, SqueezeNet, MobileNets, NU-LiteNet, NUF-Net, and NUF-Net with Residual, were compared. This comparison was done using an Intel (R) Xeon (R) E5-2683 v3 @ 2.00 GHz 56 Core CPU, 64 GB RAM, NVIDIA Tesla K80 GPU, and the Ubuntu Server 16.04.3.

For the data-preprocessing, we used data augmentation. Images with 160 × 160 pixels were randomly cropped by adding a zero-pad to increase image size to 182 × 182 pixels, together with a random flip. The setting parameters were: Solver, Adam; Learning rate (LR) = 0.5; Epoch size = 90 for the CASIA-WebFace dataset and Epoch size = 275 for the VGGFace2 dataset; and the Activation function was Rectified linear unit (ReLU).

4.1. Experiments

The results showed that the Inception-ResNet-V1 model achieved high-performance accuracy. The SqueezeNet model had a lower number of parameters than MobileNets versions 1 and 2, NU-LiteNet models, and NUF-Net models with the same number of features. With two datasets, CASIA-WebFace and VGGFace2, the LFW dataset was used in the testing process. A comparison between the various networks was also done to determine the overall performance of each network. In the NU-LiteNet and NUF-Net models, a residual technique and a non-residual technique were applied. For the NUF-Net, the network was further tested with 512 features and 128 features.

NUF-Nets vs. SqueezeNet

Comparing the NUF-Net-512 model with the SqueezeNet model (see Table 2), the NUF-Net-512 yielded a higher accuracy than SqueezeNet,

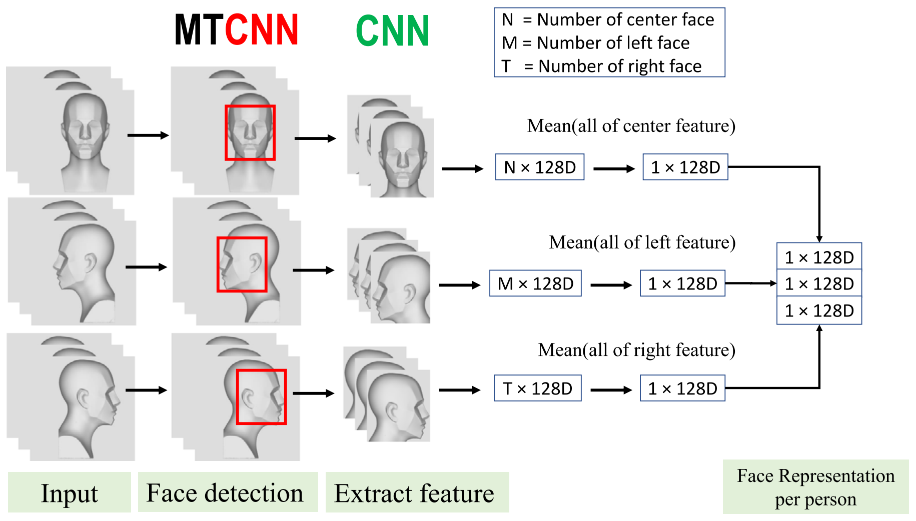

Fig. 8. Face representation.

Table 2
Results of face recognition using CASIA-WebFace and VGGFace2 training and LFW testing.

Model	Feature dimensionality	CASIA	VGGFace2	Params (Million)
Inception-ResNet-V1	512	98.83	99.60	28.18
SqueezeNet	512	97.36	98.30	6.45
MobileNet-V2	512	97.76	99.00	21.79
NU-LiteNet	512	97.85	98.86	7.46
NU-LiteNet with Residual	512	97.85	98.73	7.07
NUF-Net	512	97.91	99.13	7.10
NUF-Net with Residual	512	97.71	99.08	6.72
NUF-Net	128	97.41	98.88	3.21
NUF-Net with Residual	128	97.08	98.58	2.83

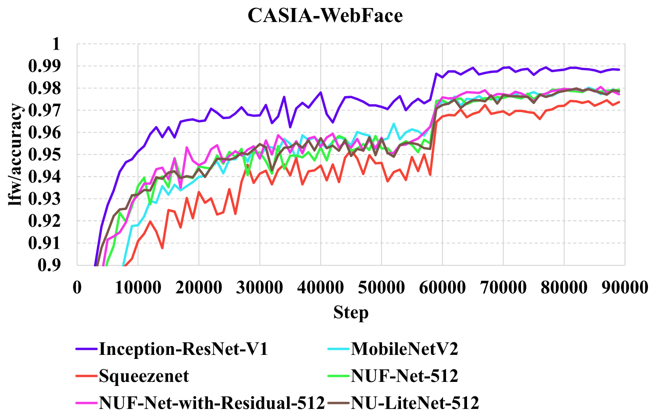

Fig. 9. All the models were trained on CASIA-WebFace. The curves denote the level of LFW Accuracy (The accuracy value range is 0.9 to 1.0) associated with the number of steps. The training process was terminated after 90,000 steps. Learning rates, mapped to ranges of steps, were: for the range 0–59,999 steps, the learning rate was 0.05, for the range 60,000–79,999 steps, the learning rate was 0.005, and for 80,000–90,000 steps, 0.0005.

with accuracy increasing by 0.55% for the CASIA-WebFace dataset, and by 0.83% for the VGGFace2 dataset. The NUF-Net-512 used 0.65 million parameters more than the SqueezeNet model. When the residual technique was applied to the NUF-Net-512, the accuracy was reduced by 0.2% for the CASIA-WebFace dataset and by 0.05% for the VGGFace2 dataset. However, the number of parameters decreased by 0.38 million in NUF-Net-with-Residual-512. Comparing NUF-Net-512 and NUF-Net-with Residual-512 against SqueezeNet, both demonstrated greater accuracy than SqueezeNet. However, NUF-Net-512 had 0.65 million more parameters than SqueezeNet and NUF-Net-with Residual-512 had 0.27 million more parameters than SqueezeNet.

NUF-Nets vs. Inception-ResNet-V1

The Inception-ResNet-V1 model achieved higher accuracy than both NUF-Net-512 and NUF-Net-with Residual-512. Using the CASIA-WebFace dataset, the accuracy of NUF-Net-512 was 0.92% lower than that of the Inception-ResNet-V1, and when the VGGFace dataset was used the accuracy was 0.47% lower. The NUF-Net-with Residual-512 showed 1.12% decrease in accuracy with the CASIA-WebFace dataset and 0.52% decrease with the VGGFace2 dataset. The results also showed that NUF-Net-512 and the NUF-Net-with-Residual-512 had only 25% of the number of parameters of the Inception-ResNet-V1.

NUF-Nets vs. MobileNets

For the MobileNets model, the source code was derived from the TensorFlow model official versions 1 and 2, which on completion, showed that MobileNetsV1 yielded only 55.13% accuracy for CASIA-WebFace and 63.31% for VGGFace2. As MobileNetsV2 achieved an accuracy of 97.76% for CASIA-WebFace and 99.00% for VGGFace2, MobileNetsV2 was chosen rather than MobileNetsV1. The accuracy of NUF-Net-512 was 0.21% higher than NUF-Net-with Residual-512 for the CASIA-WebFace dataset and 0.13% higher for the VGGFace2 dataset. Also, the accuracy of NUF-Net-with Residual-512 is 0.05% lower than MobileNetsV2 for the CASIA-WebFace dataset but was 0.08% higher for the CASIA-WebFace dataset. MobileNetsV2 had up to 21.79 million parameters, whereas NUF-Net-512 and NUF-Net-with Residual-512 had parameter size of 3 times smaller than that of MobileNetsV2.

NUF-Nets vs. NU-LiteNet

The NUF-Net model is the successor of NU-LiteNet model and it was expected to have better performance than NU-LiteNet model. NUF-Net was implemented with non-residual and with residual techniques. The results also showed that NUF-Net-512 had 0.06% higher accuracy than NU-LiteNet-512 for CASIA-WebFace dataset, and 0.27% higher accuracy for the VGGFace2 dataset. However, the performance accuracy of NUF-Net-with Residual-512 for CASIA-WebFace dataset was found to be decreased by 0.14% as compared to NU-LiteNet-with-Residual-512 while NUF-Net-with-Residual-512 for VGGFace2 dataset showed higher accuracy than NU-LiteNet-with-Residual-512, the accuracy was increased by 0.35% over NU-LiteNet-with-Residual-512. However, both of the NUF-Net models used 0.35 million fewer parameters than the NU-LiteNet models.

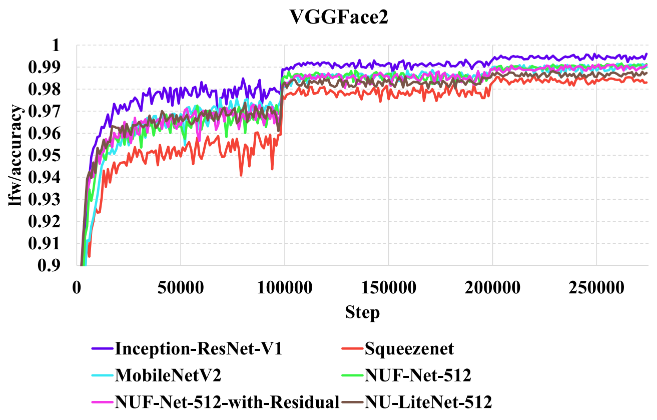

Fig. 10. All the models were trained on VGGFace2. The curves denote values of LFW Accuracy (cropped from the LFW Accuracy value of 0.9) associated with the number of steps. The training process was terminated at the step of 275,000. Learning rates were mapped to specific ranges of steps: for 0–99,999 steps, the learning rate was 0.05, for 100,000–199,999 steps, the learning range was 0.005, and for 200,000–275,000 steps, was 0.0005.

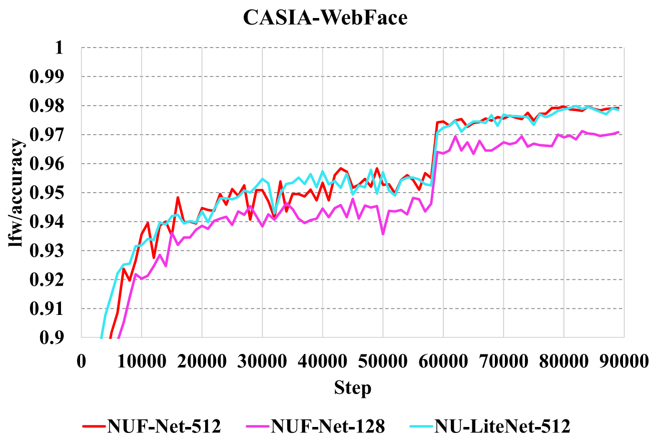

Fig. 11. Shows the training of NUF-Net-512, NUF-Net-128, and NU-LiteNet-512 on CASIA-WebFace. The curves denote values of LFW Accuracy (cropped from the LFW Accuracy value of 0.9) associated with the number of steps.

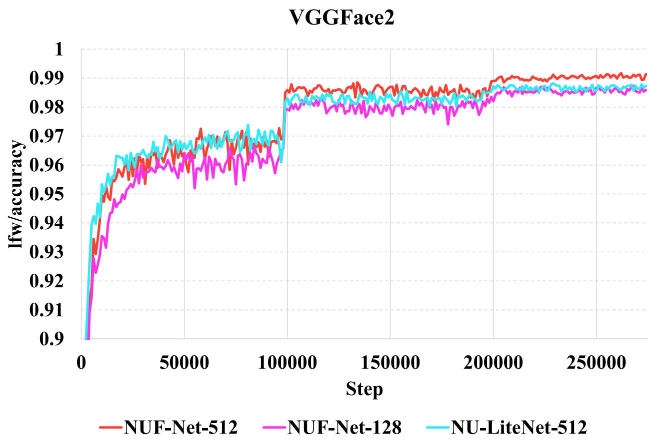

Fig. 12. Shows the training the NUF-Net-512, NUF-Net-128, and NU-LiteNet-512 on VGGFace2. The curves denote values of LFW Accuracy (cropped from the LFW Accuracy value of 0.9) associated with the number of steps.

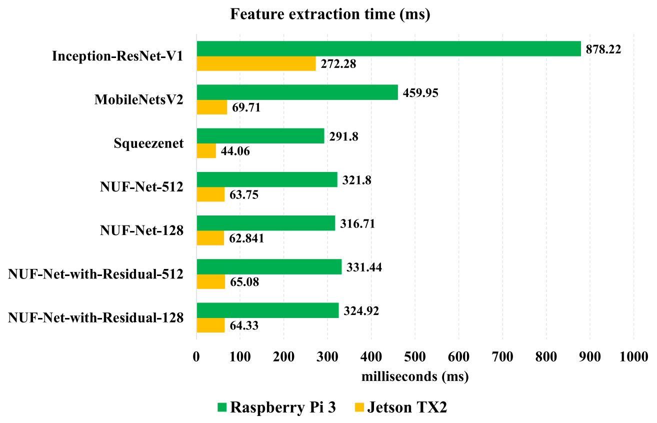

Fig. 13. Comparison of time taken for the Feature Extraction (milliseconds).

The LFW dataset was used for measuring the accuracy of the training of each model. Fig. 9 shows the accuracy of the training using the CASIA-WebFace dataset, and Fig. 10, shows the accuracy of the training using the VGGFace2 dataset. As shown in both Figs. 9 and 10, the accuracy of the Inception-ResNet-V1 model was higher than all other models for the whole span of the training phase. The SqueezeNet model demonstrated the lowest accuracy due to the internal structure of the model not being very complex. The NUF-Net provided better accuracy than MobileNetV1, MobileNetV2, and NU-LiteNet.

NUF-Net 512 features vs. 128 features

Both the NUF-Net and NUF-Net-with Residual with 512 features were reduced to 128 features to help reduce the number of parameters. The reduction of the NUF-Net features to 128 features resulted in a decrease of accuracy by 0.5% for the CASIA-WebFace dataset and 0.25% for the VGGFace2 dataset, while the number of parameters in NUF-Net-512 decreased to 45% of the previous number of parameters. Further, the NUF-Net with Residual showed an 0.63% decrease in accuracy for the CASIA-WebFace dataset, and 0.5% decrease for the VGGFace2 dataset. However, the number of parameters required by the NUF-Net with Residual was 43% of that required by the NUF-Net-with Residual with 512 features.

The reduction of features from 512 to 128 resulted in a decrease in training accuracy when using CASIA-WebFace dataset, for the entire training phase (see Fig. 11). The accuracy of the NUF-Net-128 is about 1% lower than for NUF-Net-512 and the accuracy of both NUF-Net-512 and NU-LiteNet-512 is very similar due to both models having the same model architecture but with different kernel sizes: NU-LiteNet models use 1 × 1, 3 × 3, 5 × 5 and 7 × 7 kernels while the NUF-Net-128 and NUF-Net-512 use 1 × 1, 3 × 3, 3 × 3 kernels of 2 layers and 3 × 3 of 3 layers.

From Fig. 12, the results of training with VGGFace2 can be seen to be different from training with the CASIA-WebFace dataset. When evaluating the accuracy of NUF-Net-512 against NU-LiteNet-512, the accuracy of NU-LiteNet-512 was found to be higher in the step range between 1–100,000 (learning rate = 0.5) while NUF-Net-512 had a higher accuracy for the step ranges 100,001–200,000 (learning rate = 0.05) and 200,001–275,000 (learning rate = 0.005), and the accuracy of NUF-Net-512 remained 0.4% higher than NU-LiteNet-512 until the end of the training phase.

The CASIA-WebFace and VGGFace2 dataset showed that NUF-Net-with Residual technique is better than NUF-Net. By reducing feature dimensionality from 512 features to 128 features, it incidentally reduced the number of parameters by half in NUF-Net-128. The number of parameters in NUF-Net-128 was 0.38 million less than in SqueezeNet. NUF-Net-128 also provided 0.05% higher accuracy for the CASIA-WebFace dataset and 0.58% higher accuracy for the VGGFace2 dataset. This demonstrated that NUF-Net-128 could be deployed on a device with a small processor.

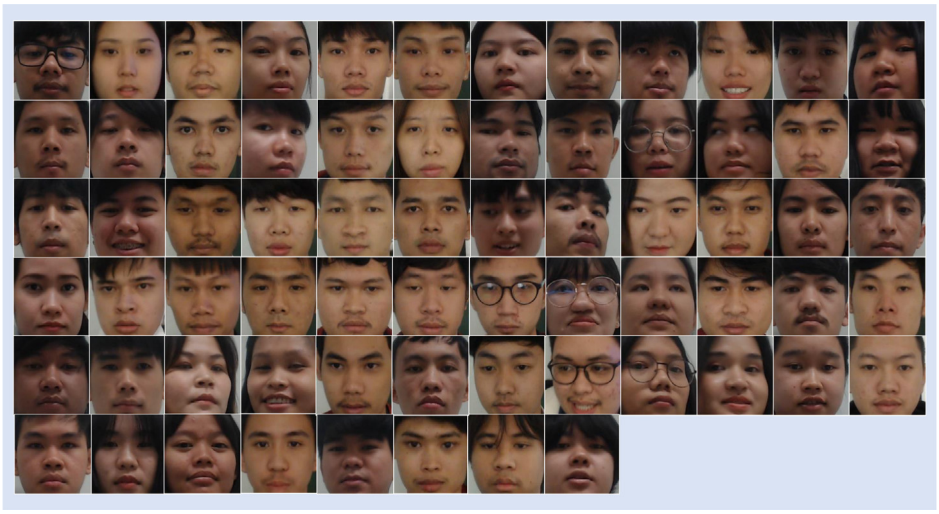

Fig. 14. Examples of images from the facial test dataset for 68 people.

Computational Cost

Fig. 13 shows the time spent in the feature extraction phase with 100 faces on the Raspberry Pi 3 (green bars) and NVIDIA Jetson TX2 (yellow bars) devices with limited resources by NUF-Net-512, NUF-Net-128, NUF-Net with Residual-128, SqueezeNet, MobileNetsV2, and Inception-ResNet-V1 model. On the Raspberry Pi 3, the NUF-Net-512 and NUF-Net with Residual-512 models processed each image in 321.8 and 331.44 ms respectively. At 128 dimensions, both NUF-Net-512 and NUF-Net with Residual-512 models took 316.71 and 324.92 ms per image, respectively, whereas the SqueezeNet model spent 291.8 ms per image. NUF-Net also has shorter processing time than the MobileNetsV2 and Inception-ResNet-V1 models and the processing time of each model was similar on both the NVIDIA Jetson TX2 device and the Raspberry Pi 3.

4.2. Application on raspberry Pi 3

Given these results, the Raspberry Pi 3 was chosen to process facial images. It was connected to the door control device via the General-purpose input/output (GPIO) port for sending a signal to open or close the door.

A sample group of 68 students participated in the experiments. Their images are shown in Fig. 14. The video recorded two days of activity of the student group, entering and exiting the classroom. The vector data of each student's face was created from the images captured in the first-day recordings, while the second-day recordings were used for testing and comparing the face vectors from the first set of data. The test data were divided into two sets: the first set containing 594 images (with 1 frame per second) and the second dataset contained 15,079 images (30 frames per second).

The multiply-accumulate operation (+×) (Ma et al., 2018) was used for analysis of the computational cost of all the models with 512 features in the experiment.

Fig. 15 compares the results of each of the face verification models, i.e. Inception-ResNet-V1, SqueezeNet, MobileNets-V2, NUF-Net, and NUF-Net-with Residual. The results of each model are shown in Fig. 15, which shows 12 different faces for each model, with the green frame (solid) indicating "accept" and the red frame (dashed) indicating "reject".

As illustrated in Table 3, the accuracy of NUF-Net was 84.17% for one frame per second (fps) and 83.55% for 30 fps and the model uses 12.03 million floating-point operations per second (FLOPs). However, when the residual technique was applied to NUF-Net, the accuracy increased to 92.59% for 1 fps and 91.18% for 30 fps with the complexity reduced to 10.12 million FLOPs.

Table 3
Comparison of accuracy and multiply-accumulate operations of the experimental system to use on the Raspberry Pi 3.

Model	+× (M FLOPs)	Accuracy (%)	
		594 images (1 fps)	15,079 images (30 fps)
Inception-ResNet-V1	46.86	92.76	91.51
SqueezeNet	3.51	80.97	79.87
MobileNets-V2	32.66	79.12	78.91
NUF-Net	12.03	84.17	83.55
NUF-Net-with-Residual	10.12	92.59	91.18

The NUF-Net-with-Residual achieved an accuracy close to the performance of a state-of-the-art model, the Inception-ResNet-V1 model, and the processing speed and complexity of the model is about 4 times better than the Inception-ResNet-V1. However, NUF-Net-with-Residual has higher accuracy than MobileNets-V2 model with complexity level being 33% of the complexity shown from MobileNets-V2.

5. Conclusion

This paper presents a face detection and identification system through the development of the NUF-Net model derived from NU-LiteNet. The model was developed to require fewer parameters and low processing time to allow deployment of the model on systems with less processing power, particularly the Raspberry Pi 3 device. Better performance was achieved by applying an increasing number of convolutional layers with different kernels to provide a variety of feature maps and adding Identity mapping techniques from ResNet model to retain data from the previous layer and to create more depth model.

Experimental results demonstrate that this was successfully achieved. Using the Raspberry Pi 3 device, the enhanced NUF-Net with Residuals technique provided an effective identification of the person in the image as close to Inception-ResNet-V1. NUF-Net-with-Residual uses a multiply-accumulate operation lower than 33% the number of times required by Inception-ResNet-V1, with only 25% of the number of parameters required by Inception-ResNet-V1. As well, the NUF-Net with Residual model demonstrated greater accuracy and less computational cost, with a lower number of parameters, than MobileNets-V2.

Acknowledgments

The authors would like to acknowledge the financial support received from the Thailand Research Fund through the Royal Golden

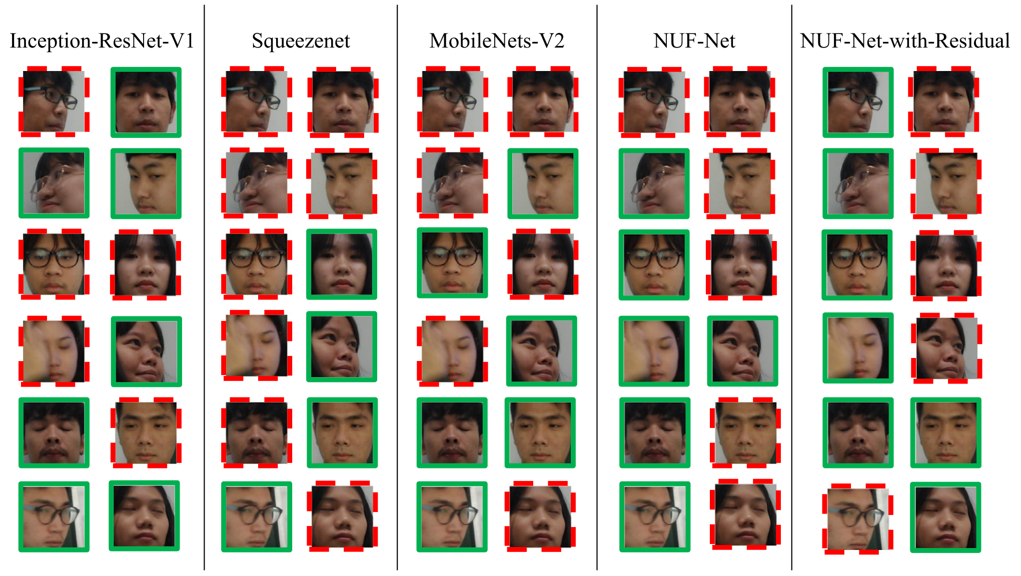

Fig. 15. Comparison of each face verification in each model. (green frame denotes "accept" and the red frame denotes "reject").

Jubilee Ph.D. Program (Grant No. PHD/0101/2559) and further, we would like to extend our appreciation to Mr. Roy I. Morien of the Naresuan University Graduate School for his assistance in editing the English grammar and expression in the paper. Finally, we would like to thank Ms. Suparat Yongjoh, Mr. Peerapol Kompunt, Mr. Pisit Rueang-wichaiwat and undergraduate students of 305438 Multimedia course (2/2561) for their unwavering support without which the research would not have been successful.

References

Cao, Q., Shen, L., Xie, W., Parkhi, O.M., Zisserman, A., 2018. Vggface2: A dataset for recognising faces across pose and age. In: 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). IEEE, pp. 67–74.

Chen, D., Cao, X., Wang, L., Wen, F., Sun, J., 2012. Bayesian face revisited: A joint formulation. In: European Conference on Computer Vision. Springer, pp. 566–579.

Chen, B.-C., Chen, C.-S., Hsu, W.H., 2014. Cross-age reference coding for age-invariant face recognition and retrieval. In: European Conference on Computer Vision. Springer, pp. 768–783.

Chen, Y.-H., Krishna, T., Emer, J.S., Sze, V., 2017. Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. IEEE J. Solid-State Circuits 52 (1), 127–138.

Cheng, G., Han, J., Zhou, P., Xu, D., 2019. Learning rotation-invariant and fisher discriminative convolutional neural networks for object detection. IEEE Trans. Image Process. 28 (1), 265–278.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Han, J., Chen, H., Liu, N., Yan, C., Li, X., 2017. CNNs-based RGB-D saliency detection via cross-view transfer and multiview fusion. IEEE Trans. Cybern. (99), 1–13.

Han, J., Cheng, G., Li, Z., Zhang, D., 2018. A unified metric learning-based framework for co-saliency detection. IEEE Trans. Circuits Syst. Video Technol. 28 (10), 2473–2483.

He, K., Zhang, X., Ren, S., Sun, J., Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778, http://dx.doi.org/10.1109/CVPR.2016.90.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

Hu, J., Shen, L., Sun, G., Squeeze-and-excitation networks, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7132–7141, http://dx.doi.org/10.1109/CVPR.2018.00745.

Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K., 2016. SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and <0.5 MB model size. arXiv preprint arXiv:1602.07360.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.

Krizhevsky, A., Hinton, G., 2009. Learning multiple layers of features from tiny images. Tech. rep., Citeseer.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems. pp. 1097–1105.

Learned-Miller, E., Huang, G.B., RoyChowdhury, A., Li, H., Hua, G., 2016. Labeled faces in the wild: A survey. In: Advances in Face Detection and Facial Image Analysis. Springer, pp. 189–248.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. nature 521 (7553), 436.

Ma, Y., Cao, Y., Vrudhula, S., Seo, J., 2018. Optimizing the convolution operation to accelerate deep neural networks on FPGA. IEEE Trans. Very Large Scale Integr. Syst. (ISSN: 1063-8210) 26 (7), 1354–1367. http://dx.doi.org/10.1109/TVLSI.2018.2815603.

Macaluso, S., Shih, D., 2018. Pulling out all the tops with computer vision and deep learning. J. High Energy Phys. 2018 (10), 121.

Nie, S., Zheng, M., Ji, Q., 2018. The deep regression bayesian network and its applications: probabilistic deep learning for computer vision. IEEE Signal Process. Mag. 35 (1), 101–111.

Nogueira, K., Penatti, O.A., dos Santos, J.A., 2017. Towards better exploiting convolutional neural networks for remote sensing scene classification. Pattern Recognit. 61, 539–556.

Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural Netw. 61, 85–117.

Schroff, F., Kalenichenko, D., Philbin, J., FaceNet: A unified embedding for face recognition and clustering, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815–823, http://dx.doi.org/10.1109/CVPR.2015.7298682.

Sun, Y., Chen, Y., Wang, X., Tang, X., 2014. Deep learning face representation by joint identification-verification. In: Advances in Neural Information Processing Systems. pp. 1988–1996.

Szegedy, C., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1–9, http://dx.doi.org/10.1109/CVPR.2015.7298594.

Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A., Inception-v4, inception-resnet and the impact of residual connections on learning, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., Rethinking the inception architecture for computer vision, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2818–2826, http://dx.doi.org/10.1109/CVPR.2016.308.

Taigman, Y., Yang, M., Ranzato, M., Wolf, L., Deepface: Closing the gap to human-level performance in face verification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1701–1708.

Termritthikun, C., Kanprachar, S., 2017. Accuracy improvement of thai food image recognition using deep convolutional neural networks. In: 2017 International Electrical Engineering Congress (IEECON). IEEE, pp. 1–4.

Termritthikun, C., Kanprachar, S., 2018. NU-ResNet: Deep residual networks for thai food image recognition. J. Telecommun. Electr. Comput. Eng. 10 (1–4), 29–33.

Termritthikun, C., Kanprachar, S., Muneesawang, P., 2018. NU-LiteNet: Mobile landmark recognition using convolutional neural networks. arXiv:1810.01074.

Termritthikun, C., Muneesawang, P., Kanprachar, S., 2017. NU-InNet: Thai food image recognition using convolutional neural networks on smartphone. J. Telecommun. Electr. Comput. Eng. 9 (2–6), 63–67.

Viola, P., Jones, M., Rapid object detection using a boosted cascade of simple features, in: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, vol. 1, 2001, pp. I–I, http://dx.doi.org/10.1109/CVPR.2001.990517.

Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E., 2018. Deep learning for computer vision: a brief review. Comput. Intell. Neurosci. 2018.

Yi, D., Lei, Z., Liao, S., Li, S.Z., 2014. Learning face representation from scratch. arXiv preprint arXiv:1411.7923.

Zhang, D., Han, J., Zhao, L., Meng, D., 2018. Leveraging prior-knowledge for weakly supervised object detection under a collaborative self-paced curriculum learning framework. Int. J. Comput. Vis. 1–18.

Zhang, K., Zhang, Z., Li, Z., Qiao, Y., 2016. Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Process. Lett. (ISSN: 1070-9908) 23 (10), 1499–1503. http://dx.doi.org/10.1109/LSP.2016.2603342.