Recognizing Very Small Face Images Using Convolution Neural Networks[*]

Shi-Jinn Horng, Julian Supardi, Wanlei Zhou, Chin-Teng Lin, and Bin Jiang

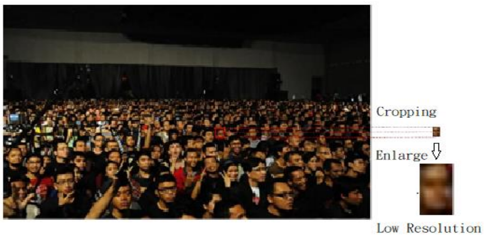
Abstract

Face recognition can be installed in a surveillance system so that it can be used for monitoring, tracking and access control. An excellent, intelligent surveillance system should be sensitive to the objects far away from the camera. Unfortunately, due to the long-distance, objects like human faces captured by the camera are too small to identify. As to enhance the subtle color differences in the face image, in this paper though we first improve the resolution of the captured image using deep convolution neural networks (DCNNs). Then the efficient features are extracted and used to do classification. As for verifying the effectiveness of the proposed method, we used three databases including AR face database, Georgia Tech face database (GT) database, and Labelled Faces in the Wild (LFW) database, altogether, to conduct the training and testing. Compared to the existing approaches, experimental results show that the identification accuracy of the proposed method outperforms to any existing approaches.

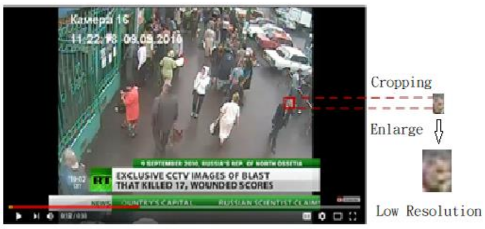
Keyword: Very low resolution; Convolution Neural Networks; Face Recognition; Low Resolution; Super-Resolution; Residual Learning

1. Introduction

For security, face recognition is one of major component of automated systems installed in the smart car. Not only it can recognize the owner of the car, but it can be integrated into the surveillance system to do self-protection from various acts of crime caused by passengers or other people. Besides, facial recognition systems have been successfully implemented in many applications in the real world, such as at immigration check-in counter, security systems, and attendance recording machines. Moreover, some methods of face recognition systems have been implemented successfully, such as Principle Component Analysis (PCA) [1][2][3], Linear Discriminant Analysis (LDA)[4], Singular Value Decomposition (SVD) [5], Canonical Correlation Analysis (CCA) [6], Artificial Neural Network (ANN) [7], and Deep Learning [8].



(a)



(b)

Fig. 1. Small sizes of face images taken from the Internet: (a) Photo taken from many people; (b) Photo taken from a long distance camera.

However, a surveillance camera usually produces a small size of face image as shown in Fig. 1(a) and 1(b), respectively. It can be seen that the face area got is so small, and it is really hard to apply the ordinary face recognition system directly, because the face area that is still good for recognition is about 32x32 pixels [9], and 64x54 pixels [10]. Meanwhile, for a face image smaller than 32 × 32 pixels, the recognition accuracy was degraded seriously as mentioned in [11]. To solve this problem, two stages are carried out by many researchers. The first stage is trying to enlarge the size of the face image and the second stage is doing the enhancement of image quality [12]. The output from the first stage is usually the blurry image so that it has fewer details and less subtle differences in the facial image. Therefore, the second stage is to aim at improving the quality of the image produced from the first stage. Using this approach, the input image to face recognition is no longer the original small face image, but the face image resulted from the reconstruction. The image accordingly is known as super-resolution images (SR) [13] and is popularly known as facial hallucinations [14][15]. In some cases, the facial recognition accuracy for low-resolution (LR) images can be improved using hallucinated face images [10]. However, in other cases, the authors [16] revealed that although visual image hallucinations have much better detail and sharpness, it could reduce the recognition accuracy compared to the LR version (from 96.35% to 96.30%). In [17], Li et al. showed that the performance decreases with decreasing resolution for datasets using super-resolution techniques. Therefore, how to develop the right model greatly influences the accuracy of hallucination facial recognition.

As for improving the LR image to become the SR image, several approaches have been successfully developed. In [18], Yang et al. proposed a method using the Spare Coding, then Dong, et al. [19] implemented this method to produce the Peak Signal-to-Noise ratio (PSNR)/accuracy ranged from 26.54 dB to 31.42 dB for the

[*]This work was supported in part by the Ministry of Science and Technology under contract number 108-2218-E-011-006-, and it was also partially supported by the “Center for Cyber-physical System Innovation” from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan.

Shi-Jinn Horng is with the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, 10607, Taiwan, ROC, is with School of Computer Science in University of Technology Sydney (UTS), and is also with the Center for Cyber-physical System Innovation, National Taiwan University of Science and Technology, Taipei, Taiwan (corresponding author, tel.: 886-2-27376700, e-mail: horngsj@yahoo.com.tw).

J. Supardi is with the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, 10607, Taiwan, ROC (is also with the Department of Informatics Engineering, Sriwijaya University, Indonesia) (email: julian@unsri.ac.id and d10415802@mail.ntust.edu.tw).

Wanlei Zhou is currently the Head of School of Computer Science in University of Technology Sydney (UTS) (email: Wanlei.Zhou@uts.edu.au).

Chin-Teng Lin is the Director, Computational Intelligence and Brain Computer Interface Centre (CIBCI), FEIT, UTS (email: Chin-Teng.Lin@uts.edu.au).

Bin Jiang is with the Department of Information Engineering, College of Computer Science and Electronic Engineering, Hunan University, China (email: jiangbinedu@gmail.com).

magnification scale x3. In [20], the authors developed the Bayesian method. The proposed method consists of a two-steps approach to statistical modeling; namely, a global parametric model and a local nonparametric model. The PSNR obtained by this method is ranged from 22.85dB to 32.87 dB. Furthermore, in [14], the proposed method used a prior gradient and obtained the RMS pixel errors ranging from 23.7 to 33.3.

One method to make SR based on learning is using Convolutional Neural Networks (CNNs). Usually, Convolutional Neural Networks with multiple hidden layers are named as Deep Convolutional Neural Networks (DCNNs). Typically, CNNs or DCNNs are used to solve classification problems. For example, AlexNet (2012) [21], Clarifai (2013) [22], GoogLeNet (2014) [23], ResNet 2015 [24] were the winners of ILSVRC in the past few years. Despite the classification, we can also use CNNs to improve image resolution, as stated in [19][25][26].

In [19], Dong et al. proposed a method named Super-Resolution Convolutional Neural Network (SRCNN), which can successfully increase the resolution of input images to become SR images. In this method, the model is composed of three layers: convolution extraction/ representation, non-linear mapping, and reconstruction. The filter sizes are 9×9, 1×1, and 5×5, respectively. Compared to the previous methods, the advantage of SRCNN is simple, but it can improve the PSNR to 32.57 dB for upscaling factor 3. However, it resulted from natural images. Unfortunately, the experiments to enhance the image resolution from very low-resolution of the face image is not satisfactory (see Fig. 2, for example). Therefore, we still require a special approach to improve the quality of the face image so that the facial features can be extracted properly. In this research, we propose the Deep Convolution Neural Networks (DCNNs) to solve the very low-resolution face recognition problem. The proposed method is named as very low face recognition using Deep Convolution Neural Network (VL-FRCNN). Here, VL-FRCNN is composed of three blocks: the first and second blocks aim to improve the resolution of face images and the third block is used for feature extraction and classification.

In general, the major differences between our method and other methods are frameworks and loss functions used in the built system. In detail, the novelty and contribution of each block are summarized in the following.

(1) For the first block: Using SRCNN [19] directly to a very low-resolution face image will produce a poor image (see Fig. 2, for example). As for improving the PSNR of the face image, we did extend two extra convolution layers for the non-linear mapping. Then applied a new loss function. Finally, Adam optimizer was applied for computing gradients.

(2) For the second block: As for improving the image resolution further, instead of using the deep convolutions layers as proposed by Pouliot et al. [27], Kim et al. [12] [26], and Zangeneh et al. [28], we used residual networks (ResNet) and arranged a new path skip connection for the deep convolution layers and a new loss function was derived.

(3) For the third block: As for extracting efficient features and doing better classification, we applied unshared weight on Fully Connected (FC) layers.

The proposed algorithms were fully analyzed and the operations proposed in these three blocks were implemented from scratch.

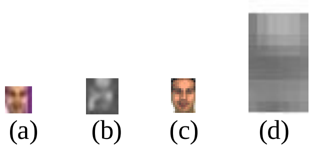

(a) (b) (c) (d)

Fig. 2. Some examples of small size images and the results after using SRCNN: (a) The original image consists of 13x13 pixels. (b) The result becomes 18x18 pixels, after using SRCNN. (c) The original image consists of 12x18 pixels. (d) The result becomes 28x54 pixels, after using SRCNN.

The remaining of this paper is organized in the following. The motivation is introduced in Section 1. The related works are discussed in Section 2. The proposed method is described in detail in Section 3. Experiments are shown in Section 4, and the concluding remarks are summarized in the last section.

2. Related works

2.1. Image super-resolution

To implement SR in real applications, researchers have tried to obtain images for SR through hardware controls [29]. However, increasing resolution is still needed, especially in areas such as video surveillance, medical diagnosis, and remote sensing applications [29].

In general, a low-resolution image can be expressed by Eq. (1):

$$Y = DHX, \qquad (1)$$

where Y is the low-resolution image, X is a high-resolution image, D and H are downsampling operators and low pass filters, respectively.

Manipulation of Eq. (1) will produce images with a higher resolution (HR). Furthermore, the high-resolution image produced from the LR image often called the super-resolution image (SR). One metric to determine the visuality of the reconstructed image is using PSNR. In the case, the higher the PSNR is, the visuality of the image reconstructed is better. Fig. 3 shows an example.

In [30], Wang et al. stated the approaches to produce the SR images which can be grouped into two categories: the reconstruction-based approach and the learning-based approach. The former has been carried out by estimating HR images from the sequence of LR images. Meanwhile, the latter explored the relationship of mapping between pairs of high-resolution and low-resolution images [31].

For the reconstruction-based approach, several methods were developed. Elad et.al [32] used two iterative algorithms to generate the desired image sequence. Huang, et al. [33] used sparse representation and improved objective function by autoregressive (AR) regularization and non-local (NL) similarity regularization. Zhang et al. [34] proposed a single-image SR algorithm based on the rational fractal interpolation model. and Yu et al. [35] made a unified learning framework using a two-stage optimization. The last three methods [33], [34], [35] produced the maximum PSNR of 35.25 dB, 36.757 dB, and 34.14 dB, respectively.

Furthermore, some methods including Kim et al. [12] [26], and Lin et al. [36] were also developed based on the learning-based approach. To get SR images, Kim et al. proposed two deep convolutional network architectures each with 20 convolution layers; one is using Very Deep Super Resolution (VDSR) to produce PSNR of 37.06 dB [12], and the other is using Deeply Recursive Convolutional Network to produce PSNR of 37.63dB [26]. Here, to overcome the vanishing/exploding gradient problem during the training phase, both methods of Kim et al. [12] [26], use skip connections as done by Svoboda et al. [37], Dong et al. [38] and Shi et al. [39], respectively. By the way, Lin et al. [36] proposed the rules to analyze the limits of learning-based SR algorithms.

Even though many methods have been developed from both approaches, the methods developed based on a learning-based approach tend to produce better super-resolution images [40].

On the other hand, face hallucination is a high-resolution face image reconstructed from a corresponding low-resolution face image using super-resolution techniques as mentioned previously. A number of methods had been proposed for face hallucination. Sun et al. [41] used Bayesian approach to enhance the primal sketch priors like edges, ridges, and corners. Xiong et al. [42] used feature enhancement method through a combination of interpolation with prefiltering and non-blind sparse prior deblurring. Li et al. [43] proposed the Space-based Local-pixel structure method and produced PSNR of 28.901 dB. An and Bhanu [44] proposed a Two-Dimensional Canonical Correlation Analysis (2D CCA) method and generated PSNR of 32.45-34.89 dB. Further, Yang et al. [45] proposed a Local Image Structure-Based method and obtained PSNR of 30.18-34.64 dB. Huang and Liu [46] used CNN and the Iterative Back Projection method and produced PSNR of 29.56-30.90 dB. Next, Rahiman and Jiji [47] used Eigen Transformation and produced PSNR of 22.404 -33.424 dB.

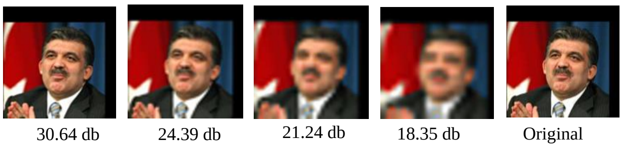


Fig. 3. The images at different levels of PSNR.

2.2. Low-resolution face recognition

Increasing the use of Video Surveillance for public security has made a research on improving the image quality of low-resolution face images because images obtained from surveillance cameras generally have a low-resolution, while the image used as a reference or training is a high resolution. Typically, there were some methods proposed to solve low-resolution face recognition problems. In [5], Jian et al. used singular value decomposition for performing both face hallucination and recognition simultaneously. By combining LFW, GT, and AR database together, the method proposed by Jian et al. [5] gets PSNR of 22.83 dB and the recognition accuracy is 72.15 % for face images each of size 18x16 pixels using closed-set scenario. Meanwhile, Shakeel et al. [48] developed a low-rank matrix and sparse error matrix from local feature extraction and used a sparse coding algorithm to learning a projection matrix, and then used linear regression to recognizing face image using the combined databases from Extended Yale-B, Multi-PIE, FERET, LFW, and Remote face databases. Shakeel et al. [48] used open-set scenario for experiments and obtained the recognition accuracy of 95.41 % for face images each of size 20x20 pixels. Lu et al. [49] used a double coupled ResNet for low-resolution face recognition using the combined databases from LFW and SCFace. The experiments are based on open-set scenario and the recognition accuracy of this method is 97.3 % for face images each of size 20x20 pixels. Gie et al. [50] used CNN and developed a pair of models between teacher stream to recognize high-resolution face image and student stream to recognize low-resolution face image. Both models are trained simultaneously and the problem of limited computational resources is handled via the selective feature approximation. Using the combined databases from UMDFace, LFW, UCCS, and SCFace, the experiments are based on open-set scenario and the recognition accuracies of this method are 95.03% and 89.72% for face images each of size 96x96 pixels and 32x32 pixels, respectively. Farrugia and Guillemot [51] proposed a method using Linear Models of Coupled Sparse Support. The method contains two dictionaries, i.e. HR dictionary and LR dictionary and then learns linear models based on the local geometrical structure. Wu et al. [16] proposed a deep model composed of two parts, i.e. face hallucination for increasing image resolution and recognition layers for face recognition, respectively. Using open-set scenario, the method gets the recognition accuracy of 97.95 % for x4 downsampling images from the LFW database. Horng et al. [52] proposed a method combining RBM for feature extraction and DCNN for recognition and the recognition accuracy for open-set scenario is 78.33% for LFW, AT, and AR databases with face images each of size 18x16 pixels. Supardi and Horng [53] proposed a method based on both SRCNN and CNN, where SRCNN is used to increase resolution and CNN is used to do recognition. Using open-set scenario in the experimental setting, the recognition accuracy is 99.00% for 50 persons out of the AR database and the size of each face image is 6x8 pixels. For low-resolution face identification, Li et al. [17] used a neural network with seven layers with center loss for experiments using two datasets. In SCFace dataset, they [17] used the HR images as gallery images, and those captured at the three standoffs as probe images and got 31.71% rank-1 accuracy; on the other hand, they [17] chose face images from 1m standoff as gallery images and images from 2.6m and 4.2m standoff as probe images and got 69.60% rank-1 accuracy. However, both accuracies are improved, compared to the state-of-the-art under the same protocol. In UCCSface dataset, for closed-set evaluation, their method beats the UCCS baseline by nearly 20% on rank-1 accuracy; for open-set evaluation, they achieve 73.6% accuracy when compared to the UCCS face baseline result.

3. The Proposed Method

Many methods related to super-resolution images and hallucination face recognition have been proposed. However, to develop a new model has still needed to meet the practical environment.

The framework of the proposed method consists of two main parts. The first part aims to increase the resolution of the face images, and the second part is to extract the efficient features and use them to do the classification. Both parts are composed of three blocks. The first block was inspired by the SRCNN model proposed by Dong et al. [19]. This model is relatively simple and has proven to provide good results to natural images, but in this research, the input of the system is the very low-resolution face images, so that besides visualization, we also have to pay attention to the texture of the images. Using SRCNN directly to increase resolution from a very low-resolution face image will produce a poor image (see Fig. 2, for example). For that, in the method proposed, we did extend the SRCNN architecture by adding two convolution layers, 1x1x48 and 1x1x32, in the non-linear mapping layer in SRCNN.

The first block has succeeded in improving image quality compared to the output of SRCNN, but it is not enough for face recognition because the maximum PSNR achieved is less than 27 dB. For this reason, we added a second block to improve the resolution of the output images of the first block. The second block consists of deep convolution networks. The architecture inspired by the method proposed by Pouliot et al. [27], Kim et al. [12] [26], and Zangeneh et al. [28]. They did succeed in developing a very deep convolutional neural network to enhance image resolution. For both blocks 1 and 2, to broaden the previous architecture, besides adding a convolutional layer, we also implemented a new loss function. In the previous methods, the loss function used is MSE [78]. It is a standard loss function and one of popular loss functions used in artificial neural networks; however, in our architecture, the MSE does not provide the best results.

The third block is the last block, and it contains several convolution layers and a pooling layer. In each block, we use the Rectified Linear Unit (ReLU) function $\varphi = \mathrm{Max}\,(0, x)$ as the activation function.

Fig. 4. shows the framework of the proposed method, and the detailed information about each block is summarized as follows:

- The first block consists of five convolution layers each of kernel size 9x9x128, 1x1x64, 1x1x48, 1x1x32, and 5x5x3, respectively.
- The second block contains twenty-two convolution layers: the convolution layer of kernel size 7x7x96 is one layer, the convolution layers each of kernel size 3x3x64 are nineteen layers, the convolution layer of kernel size 3x3x3 is one layer, and the convolution layer of kernel size 1x1x3 is one layer. It is to further enhance the resolution of images produced previously.
- The third block contains eleven convolution layers: convolution layers each of kernel size 3x3x64 are two layers, the convolution layers each of kernel size 3x3x128 are three layers, the convolution layers each of kernel size 3x3x256 are three layers, and the convolution layers each of kernel size 3x3x512 is three layers. Besides, this block also contains four layers of pooling and three layers in fully connected. This block aims to feature extraction and classification.

Furthermore, we follow [54][55][56], to do mathematical calculations on each layer in each block.

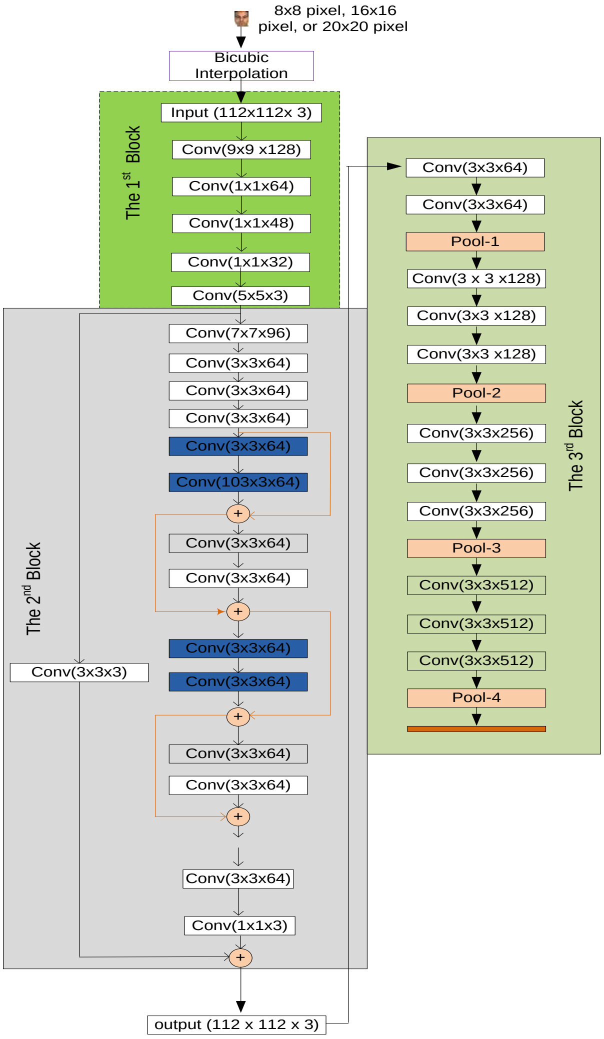


Fig. 4. The framework of the proposed method.

3.1. Convolution Layer

Let I be the input of the convolution layer in the l^{th} layer, K be the kernel, and B be the bias. The convolution layer calculated by Eqs. (2) and (3) is listed as follows:

$$Y_{r,s}^{(l)} = B^{(l)} + \sum_{u=-H_1}^{H_1} \sum_{v=-H_2}^{H_2} \sum_{d=0}^{D} K_{u,v}^{(l)} * I_{r+u,s+v}^{(l)} \qquad (2)$$

$$I_{r,s}^{(l+1)} = \varphi(Y_{r,s}^{(l)}). \qquad (3)$$

where H_1 and H_2 are sizes of kernel K, D is the number of Kernels K, $r=0,\ 1,\ \dots,\ m$ and $s=0,\ 1,\ \dots.,\ n.$

3.2. Pooling Layer

The pooling layer/subsampling layer is a layer that serves to decrease the feature resolution. The purpose of pooling is to make the highlights more impervious to noise and distortion. There are two different ways to do pooling: max pooling and average pooling. The initial step in two cases is the same, partitioning the pixel matrix into several two-dimensional matrices, for each partition, determine the maximum value for max-pooling (or the average value for the average pooling). In other words, max-pooling has gotten by taking the maximum estimation of every region, while average pooling takes the average estimation of every region.

3.3. Fully connected layer

In this research, there are three layers of fully connected layers. The first layer (FC-1) receives input from the final pooling layer, the second layer (FC-2) is a bridge layer connecting FC-1 and FC-3. The third layer (FC-3) is the output layer and is used for classification. The three fully connected layers are shown in Fig. 5.

The calculation of a fully connected layer is using the backpropagation algorithm [7]. Let V be the weights between FC-1 and FC-2 layers, and W be the weights between FC-2 and FC-3 layers. Then the calculation of each neuron in FC-2 and FC-3 is done by Eqs. (4) and (5), respectively:

$$Z_j = \varphi(\textstyle\sum_{i=1}^{p} M_i V_{ij}),\ j=1,2,3,\dots,(2p+1) \qquad (4)$$

$$Y_k = f(\textstyle\sum_{j=1}^{2p+1} Z_j W_{jk}),\ k=1,2,3,\dots,N, \qquad (5)$$

where M_i is the output of the last pooling layer, p is set to m*n corresponding to m rows and n columns of the last matrix pooling layer, φ is the ReLU activation function, and f is the softmax activation function, N is the number of persons, Z_j and Y_k are the outputs of neurons at FC-2 and FC-3, respectively.

Furthermore, we use Eqs. (6) and (7) to update the weights on the fully connected layers.

$$V_{ij}(new) = V_{ij}(old) - \eta \frac{\partial \mathcal{L}}{\partial V}, \qquad (6)$$

$$W_{jk}(new) = W_{jk}(old) - \eta \frac{\partial \mathcal{L}}{\partial W}, \qquad (7)$$

where η is the learning rate, $\frac{\partial \mathcal{L}}{\partial V}$ is the derivative of the loss function with respect to V, and $\frac{\partial \mathcal{L}}{\partial W}$ is the derivative of the loss function with respect to W.

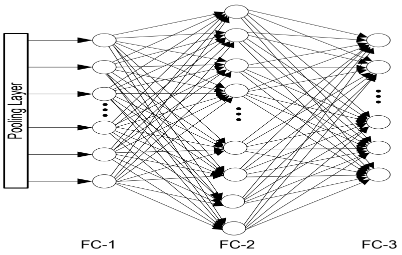


Fig. 5. Fully Connected Layer

3.4. Training Phase

In the proposed method, each block has unique characteristics and functionally independent, so that training for each block can be done separately. However, the basic training principle for each CNN in each block is the same, but the difference is the database for training, loss functions used, and parameter settings.

Furthermore, the calculation steps in each layer for the feed-forward phase follow from [54] and those for the backward phase follow from [57]. Meanwhile, the principle of the weight update rule is following the standard backpropagation algorithm [58] and applies Adam Optimizer [59]. So, mathematically to update the weights w and bias b at time *t* is using Eq. (8) and Eq. (9), respectively:

$$w(t+1) = w(t) - \alpha \frac{\widehat{m}_{t_w}}{\sqrt{v_{t_w}}+\in}, \text{ for } \in > 0 \tag{8}$$

$$b(t+1) = b(t) - \alpha \frac{\widehat{m}_{t_b}}{\sqrt{v_{t_b}+\in}}, \text{ for } \in > 0 \tag{9}$$

$$m_{t_w} = \beta_1 m_{t_w-1} + \left(1-\beta_1\right) g_{t_w}$$
$$m_{t_b} = \beta_1 m_{t_b-1} + \left(1-\beta_1\right) g_{t_b}$$
$$v_{t_w} = \beta_2 v_{t_w-1} + (1-\beta_2)\, g_{t_w}^2$$
$$v_{t_b} = \beta_2 v_{t_b-1} + (1-\beta_2) g_b^2$$
$$\widehat{m}_{t_w} = \frac{m_{t_w}}{(1-\beta_1^t)};\ \widehat{m}_{t_b} = \frac{m_{t_b}}{(1-\beta_1^t)}$$
$$\hat{v}_{t_w} = \frac{v_{t_w}}{(1-\beta_2^t)};\ \hat{v}_{t_b} = \frac{v_{t_b}}{(1-\beta_2^t)}$$

where m_{t_w} is the first moment of weight w, v_{t_w} is the second raw moment of weight w, m_{t_b} is the first moment of bias b, v_{t_b} is the second raw moment of bias b, $\widehat{m}_{t_w}$ is the weight-corrected first moment, $\hat{v}_{t_w}$ is the weight-corrected second raw moment, $\widehat{m}_{t_b}$ is the bias-corrected first moment, $\hat{v}_{t_b}$ is the bias-corrected second raw moment, α is learning rate, β_1 *and* β_2 are hyperparameters, $g_{t_w} = \frac{\partial \mathcal{L}}{\partial w}$ is the partial derivative of the loss function with respect to *w*, and $g_{t_b} = \frac{\partial \mathcal{L}}{\partial b}$ is the partial derivative of the loss function with respect to *b*.

In the training, we initialized the weights using random numbers that are generated by a Gaussian distribution with zero means, and the standard deviation is 0.001, and 0 for biases for every block.

The complete flowcharts for training the first, second, and third blocks are shown in Figs. 6~8. We conduct the joint training between the developed networks and the pre-trained VGG-19 model for the first two blocks. This idea was not used in face recognition before.

3.4.1. Training the first block

Given a low-resolution I_{LR} image, a network tries to predict the image as similar to I_{HR} as possible. Suppose $\mathbb{H}$ is a CNN network with parameter ω, training $\mathbb{H}$ aims to minimize the loss function ($\mathcal{L}$) between the output of $\mathbb{H}$ and I_{HR}, as shown mathematically in Eq. (10):

$$\omega = \underset{\omega}{\arg\min}\, \mathcal{L}(\mathbb{H}(I_{LR}), I_{HR}) \tag{10}$$

Now, we define $\xi_{16}(\mathrm{x})$as the feature maps produced by the last (16th) convolutional layer of the pre-trained VGG-19 model with input image x [60], and two-loss functions $\mathcal{L}_{MSE}$ and $\mathcal{L}_c$ as shown in Eqs. (11) and (12), respectively.

The total loss function is shown in Eq. (13) [60].

$$\mathcal{L}_{MSE} = \frac{1}{n}\sum_{i=1}^{n} \left\| I_{HR_i} - (\mathbb{H}\left(I_{LR_i}\right)) \right\|_2^2 \tag{11}$$

$$\mathcal{L}_c = \frac{1}{n}\sum_{i=1}^{n} \left\| \xi_{16}(I_{HR_i}) - \xi_{16}(\mathbb{H}\left(I_{LR_i}\right)) \right\|_2^2 \tag{12}$$

$$\mathcal{L} = 0.5\,(\gamma_c\, \mathcal{L}_c + \gamma_m \mathcal{L}_{MSE}) \tag{13}$$

Here, γ_c and γ_m are trade-off parameters of $\mathcal{L}_c$ and $\mathcal{L}_{MSE}$, respectively.

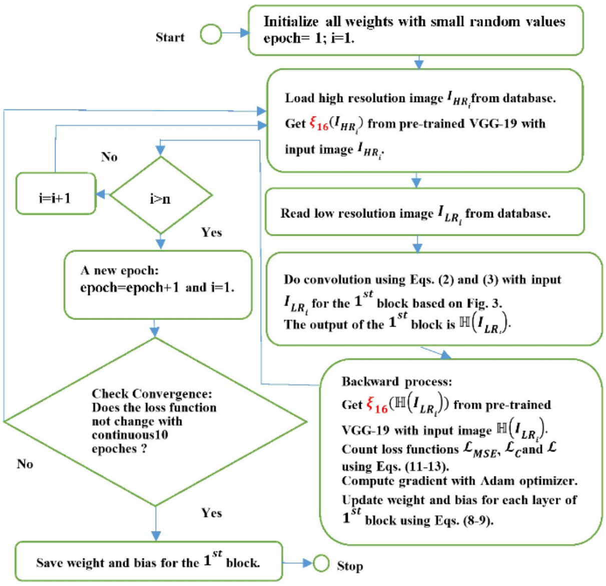

Start
Initialize all weights with small random values epoch= 1; i=1.
Load high resolution image $I_{HR_i}$from database.
Get $\xi_{16}(I_{HR_i})$ from pre-trained VGG-19 with input image I_{HR_i}.
No
i=i+1
i>n
Yes
Read low resolution image I_{LR_i} from database.
A new epoch: epoch=epoch+1 and i=1.
Do convolution using Eqs. (2) and (3) with input I_{LR_i} for the 1st block based on Fig. 3. The output of the 1st block is $\mathbb{H}(I_{LR_i})$.
Check Convergence: Does the loss function not change with continuous10 epoches ?
No
Yes
Backward process: Get $\xi_{16}(\mathbb{H}(I_{LR_i}))$ from pre-trained VGG-19 with input image $\mathbb{H}(I_{LR_i})$. Count loss functions $\mathcal{L}_{MSE}$, $\mathcal{L}_c$and $\mathcal{L}$ using Eqs. (11-13). Compute gradient with Adam optimizer. Update weight and bias for each layer of 1st block using Eqs. (8-9).
Save weight and bias for the 1st block.
Stop

Fig. 6. Flowchart for training the first block

Furthermore, as mentioned in Section 3.1, the first block is extended from SRCNN, but in the training phase, we made changes to the learning rate settings. As we know, the SRCNN [19] has three convolution layers: the learning rate for the first two layers are set to 10^{-4}, and the last layer is set to10^{-5}. Meanwhile, we used Adam optimizer [59] and obtained a suitable learning rate 10^{-5} for all convolution layers in the first block. Nevertheless, the database used as training data for the first block is the ImageNet database of ILSVRC in 2013 which contains 395909 images. It is the same as used in SRCNN. Here, if the loss function after ten epochs did not change, then the learning process was stopped.

The main steps of training the first block using n pairs of images are described in the following.

Forward Process: A pair of high-resolution/ low-resolution images ($\mathrm{I_{HR_i}}, \mathrm{I_{LR_i}}$) were used to train the convolution blocks. The output of the last convolution block was set to $\mathbb{H}(\mathrm{I_{LR_i}})$.

Backward Process: Counted the loss functions based on $\mathrm{I_{HR_i}}$, $\mathbb{H}(\mathrm{I_{LR_i}})$, $\xi_{16}(\mathrm{I_{HR_i}})$, and $\xi_{16}(\mathbb{H}(\mathrm{I_{LR_i}}))$ using Eqs. (11)~(13). The weight and bias of each layer were then updated. An epoch was to repeat these two processes from i=1 to i=n. If the loss function after ten epochs did not change, then the training process was stopped. The detailed flow chart can be seen in Fig. 6.

Training the second block

In the second block, we used a database from Yang et al. [18] and BSD200 [61]. During the training process, each output of the ReLU layer was applied dropout [62] with p = 0.9. Furthermore, we used Adam optimizer to compute the gradient and determined the learning rate. Here, the learning rate obtained was 10-3. Furthermore, if the loss function after five epochs did not change, then the learning process was stopped.

The training process of the second block follows from [12]. However, we used different loss function, because when using VDSR directly, the PSNR was obtained around 19-21 dB [63] for images each of size 16x16 pixels. Besides, we also used one additional convolution layer and redesigned the shortcut path.

Now, we define Xi and Xi+1 are input and output of the convolution layer; also the output of the last layer with 64 channels is condensed to 3 channels.

Residual Learning

In [12], Kim et al. using a basic loss function $\frac{1}{2}\|\mathrm{y}-\mathrm{f}(\mathrm{x})\|^2$ where x is an interpolated low-resolution image and y is a high resolution image. The loss function for residual learning is $\frac{1}{2}\|\mathrm{r}-\mathrm{f}(\mathrm{x})\|^2$, where f (x) is a network prediction and r = y − x is a residual image.

In the second block, the network receives four inputs: interpolated low-resolution image, feature maps from the pre-trained VGG-19 model, residual, and ground truth image.

Now, given the image $\hat{\mathrm{I}}_{\mathrm{LR}}$ as an interpolated low-resolution image from each image in the database, and the image $\mathrm{I_{HR}}$ as a high-resolution image, then we can rewrite the loss functions defined in Eqs. (11), (12), and (13) as follows:

$$\mathcal{L}_{\mathrm{rMSE}} = \frac{1}{\mathrm{n}}\Sigma_{\mathrm{i}=1}^{\mathrm{n}}\left\|\mathrm{I_{HR_i}} - \mathbb{H}(\hat{\mathrm{I}}_{\mathrm{LR_i}})\right\|_2^2 \tag{14}$$

$$\mathcal{L}_{\mathrm{v}} = \frac{1}{\mathrm{n}}\Sigma_{\mathrm{i}=1}^{\mathrm{n}}\left\|\xi(\mathrm{I_{HR_i}}) - \xi(\mathbb{H}(\hat{\mathrm{I}}_{\mathrm{LR_i}}))\right\|_2^2 \tag{15}$$

$$\mathcal{L} = 0.5\ (\gamma_{\mathrm{v}}\,\mathcal{L}_{\mathrm{v}} + \gamma_{\mathrm{m}}\mathcal{L}_{\mathrm{rMSE}}\qquad) \tag{16}$$

Furthermore, we define $\mathrm{r} = (\mathrm{I_{HR_i}} - \hat{\mathrm{I}}_{\mathrm{LR_i}})$ as a residual image, and then the loss functions for the residual networks are listed in the following.

$$\mathcal{L}_{rMSE} = \frac{1}{n}\Sigma_{i=1}^{n}\left\|\mathrm{r} - \mathbb{H}(\hat{I}_{LR_i})\right\|_2^2 \tag{17}$$

$$\mathcal{L}_{rv} = \frac{1}{n}\Sigma_{i=1}^{n}\left\|\xi(r) - \xi(\mathbb{H}(\hat{I}_{LR_i}))\right\|_2^2 \tag{18}$$

$$\mathcal{L}_r = 0.5\ (\gamma_{rv}\,\mathcal{L}_{rv} + \gamma_m\mathcal{L}_{rMSE}) \tag{19}$$

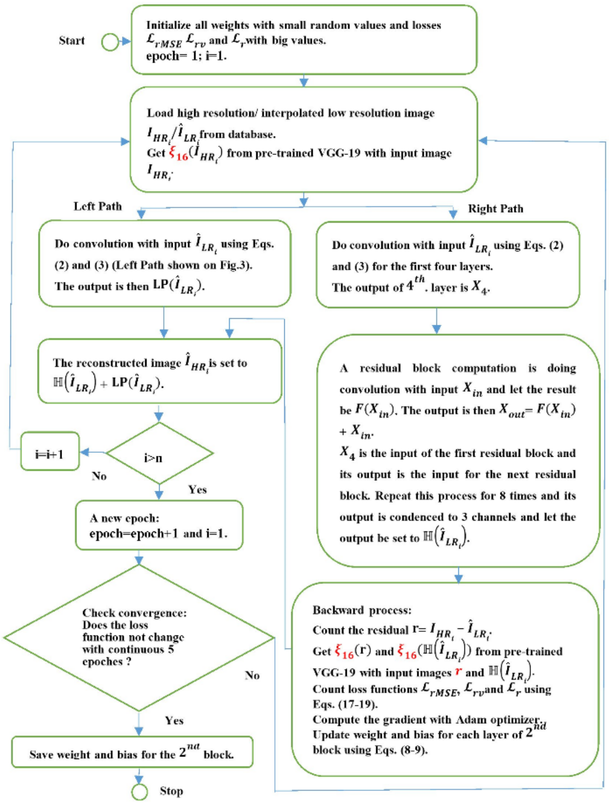

Start
Initialize all weights with small random values and losses $\mathcal{L}_{rMSE}$ $\mathcal{L}_{rv}$ and $\mathcal{L}_r$ with big values. epoch= 1; i=1.
Load high resolution/ interpolated low resolution image $I_{HR_i}/\hat{I}_{LR_i}$ from database. Get $\xi_{16}(I_{HR_i})$ from pre-trained VGG-19 with input image I_{HR_i}.
Left Path
Right Path
Do convolution with input $\hat{I}_{LR_i}$ using Eqs. (2) and (3) (Left Path shown on Fig.3). The output is then $\mathrm{LP}(\hat{I}_{LR_i})$.
Do convolution with input $\hat{I}_{LR_i}$ using Eqs. (2) and (3) for the first four layers. The output of 4^{th}. layer is X_4.
The reconstructed image $\hat{I}_{HR_i}$ is set to $\mathbb{H}(\hat{I}_{LR_i}) + \mathrm{LP}(\hat{I}_{LR_i})$.
A residual block computation is doing convolution with input X_{in} and let the result be $F(X_{in})$. The output is then $X_{out} = F(X_{in}) + X_{in}$. X_4 is the input of the first residual block and its output is the input for the next residual block. Repeat this process for 8 times and its output is condenced to 3 channels and let the output be set to $\mathbb{H}(\hat{I}_{LR_i})$.
i=i+1
i>n
No
Yes
A new epoch: epoch=epoch+1 and i=1.
Check convergence: Does the loss function not change with continuous 5 epoches ?
No
Yes
Backward process: Count the residual $\mathrm{r} = I_{HR_i} - \hat{I}_{LR_i}$. Get $\xi_{16}(\mathrm{r})$ and $\xi_{16}(\mathbb{H}(\hat{I}_{LR_i}))$ from pre-trained VGG-19 with input images r and $\mathbb{H}(\hat{I}_{LR_i})$. Count loss functions $\mathcal{L}_{rMSE}$, $\mathcal{L}_{rv}$ and $\mathcal{L}_r$ using Eqs. (17-19). Compute the gradient with Adam optimizer. Update weight and bias for each layer of 2^{nd} block using Eqs. (8-9).
Save weight and bias for the 2^{nd} block.
Stop

Fig. 7. Flowchart for training the second block

Here, the goal of residual learning is to minimize the loss function $\mathcal{L}_{\mathrm{r}}$. After the loss is minimized during training, the reconstructed image $\hat{\mathrm{I}}_{\mathrm{HR_i}}$ is then set to $\mathbb{H}(\hat{\mathrm{I}}_{\mathrm{LR_i}}) + \hat{\mathrm{I}}_{\mathrm{LR_i}}$, which is close to $\mathrm{I_{HR_i}}$.

The main steps of training the second block with n pairs of images are described in the following.

Forward Process: A pair of high-resolution/interpolated low-resolution images $(\mathrm{I_{HR_i}}, \hat{\mathrm{I}}_{\mathrm{LR_i}})$ were used to train the residual blocks. The output of the last residual block was condensed to 3 channels and its output was set to $\mathbb{H}(\hat{\mathrm{I}}_{\mathrm{LR_i}})$.

Backward Process: Set $\mathrm{r} = (\mathrm{I_{HR_i}} - \hat{\mathrm{I}}_{\mathrm{LR_i}})$ as a residual image, and then the loss functions for the residual networks based on r, $\mathbb{H}(\hat{\mathrm{I}}_{\mathrm{LR_i}})$, $\xi(\mathrm{I_{HR_i}})$, and $\xi(\mathbb{H}(\hat{\mathrm{I}}_{\mathrm{LR_i}}))$ were computed using Eqs. (17)~(19). The weight and bias of each layer were then updated. An epoch was to repeat these two processes from i=1 to i=n. If the loss function after five epochs did not change, then the training process was stopped. The detailed flow chart can be seen in Fig. 7.

3.4.2. Training the third block

Three databases, namely AR [64], GT [65], and LFW [66][67] were used for training the third block. We have taken five pictures of faces with different expressions for each person to become the training data. Since the sizes of the ground truth images from the database are different, we cropped and normalized the images to form the ground truth images each of size 112 x 112 pixels. Furthermore, we arrange a target (label) for each person during the training. The output of a neuron corresponding to a labeling person should be closed to 1 as possible since it is a real number in the range [0,1] for each person. The loss function ($\mathcal{L}$) which is used to the third block is the Large Margin Cosine Loss as stated in Eq. (20) [68]:

$$\mathcal{L} = \frac{1}{S}\sum_i -log \frac{e^{\rho(\cos(\alpha_{t_i},i)-c)}}{e^{\rho(\cos(\alpha_{t_i},i)-c)} + \sum_{k\neq t_i} e^{\rho\cos(\alpha_{k,i})}} \qquad (20)$$

With:

$$W = \frac{W^*}{\|W^*\|} \;;\; x = \frac{x^*}{\|x^*\|} \;\text{; and }\; \cos(\alpha_k, i) = W_k^T\, x_i$$

where x_i is the i^{th} feature vector corresponding to a ground-truth class label t_i, $\rho = \|x\|$, S is the number of training samples, W_k is the weight vector of the kth class, α_k is the angel between weight W_k and x_i , $c \geq 0$ is a fixed parameter introduced to control the magnitude of the cosine margin.

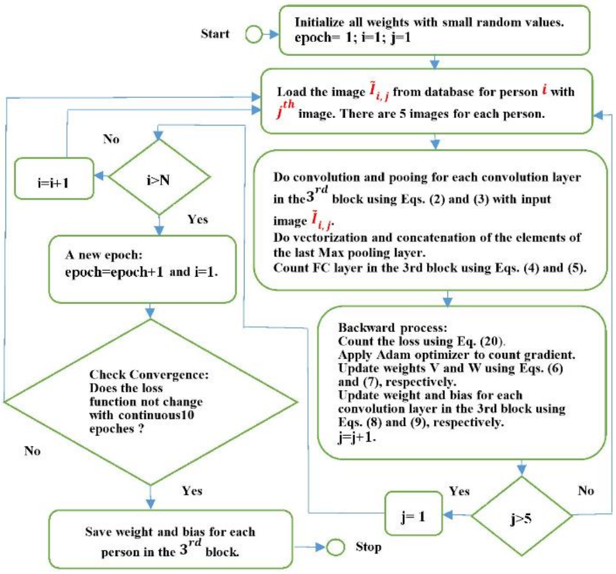

Start
Initialize all weights with small random values.
epoch= 1; i=1; j=1
Load the image $\tilde{I}_{i,j}$ from database for person i with j^{th} image. There are 5 images for each person.
No
i>N
i=i+1
Yes
A new epoch:
epoch=epoch+1 and i=1.
Do convolution and pooing for each convolution layer in the3^{rd} block using Eqs. (2) and (3) with input image $\tilde{I}_{i,j}$.
Do vectorization and concatenation of the elements of the last Max pooling layer.
Count FC layer in the 3rd block using Eqs. (4) and (5).
Backward process:
Count the loss using Eq. (20).
Apply Adam optimizer to count gradient.
Update weights V and W using Eqs. (6) and (7), respectively.
Update weight and bias for each convolution layer in the 3rd block using Eqs. (8) and (9), respectively.
j=j+1.
Check Convergence:
Does the loss function not change with continuous10 epoches ?
No
Yes
Save weight and bias for each person in the 3^{rd} block.
Stop
j= 1
Yes
j>5
No

Fig. 8. Flowchart to training the third block

Furthermore, the learning rate that was suitable for the third block was 10^{-3}. Also, if the loss function did not change again after ten epochs, the training process was then stopped. After the training was converged, the weights on the FC network were stored separately for each person.

Let N be the number of persons and set S=5N in Eq. (20) as each person has 5 images. The main steps of training the third block are described in the following.

Forward Process: For the jth image of person i, did convolution and pooling for each convolution layer using Eqs. (2) and (3). Then did vectorization of the last Max pooling layer and did computations on the fully connected layer using Eqs. (4) and (5).

Backward Process: Counted the loss functions using Eq. (20). The weight and bias of each layer were then updated for each layer. An epoch was to repeat these two processes from i=1 to i=N. If the loss function after ten epochs did not change, then the training process was stopped. The detailed flow chart can be seen in Fig. 8.

3.5. Testing Phase

The testing phase is a step to the classification of the input face image. Here, all models obtained from the training phase are combined into one unit. The weight (W) and the bias (B) for each Kernel in each convolution layer, and the weight for the fully connected layer for each person $\{W_{P_i}\}_{i=1,2,..,N}$ are loaded from the training phase. The low-resolution face image (I_{LR}) flows into the input of the first block, then the second block until the third block. The first and the second blocks do increase the image resolution, while the third block does recognize the face based on the low-resolution face image of the input.

Furthermore, we construct the following three steps for the network to recognize the input face image. For the **first step**, suppose there are N persons, let P= $\{P_1, P_2, \ldots, P_N\}$ be the person set and we define:

(1) $T = \{t^1, t^2, \ldots, t^N\}$ is the target set (label) corresponding to each person in P.

(2) $O = \{O^1, O^2, \ldots, O^N\}$ is the output set of FC-3 as shown in Fig. 5.

(3) $X = \{x_1, x_2, \ldots, x_m\}$, the input vector of FC-1 as shown in Fig. 5, which is obtained from the final Maxpooling in the previous layer.

(4) $Z = \{Z^1, Z^2, \ldots, Z^N\}$ is the output set of FC-2 as shown in Fig. 5.

(5) $f(.)$ is a softmax activation function.

For the second step, we check the similarity between the output O^i and persons P_j using Eq. (21):

$$d_{ij} = \|O^i - t^j\| \qquad (21)$$

with:

$$O^i = f(\sum_i Z^i W_{P_i} + B)$$

where Z is obtained using Eq. (4).

For the third step, we make Rule (1) for identification:

Rule (1):

If $d_{ij} = min\{d_{ij}\}_{j=1,2,\ldots,N}^{i=1,2,\ldots,N}$ and $i=j$ ***then***
    if $d_{ij} <= \delta$ ***then*** the person is P_i
    else the person is unknown.

Here, δ is the threshold estimated from training process, which is used for excluding the person who was not registered in the system before.

The complete algorithm for the testing phase is shown in Algorithm 1.

Algorithm 1: Testing Phase

```
(1) Load weights and biases saved in all blocks from the training phase to set all weights and biases (W&B) for each Kernel Convolution layer and weights and biases ($W_{P_i}$&B) for each FC.
(2) Process Convolution in the 1st block using Eq. (2) and Eq. (3)
(3) Process Convolution in the 2nd block using Eq. (2) and Eq. (3)
(4) Process Convolution and Pooling in the 3rd block using Eq. (2) and Eq. (3)
(5) Count the FC layer in the 3rd block using Eq. (4) and Eq. (5)
(6) Check the similarity using Eq. (21)
(7) Classify the image using Rule (1)
```

4. Experiments

4.1. Datasets

In this research, to verify the proposed method, we used three databases, namely AR [64], GT [65], and LFW [66][67]. The database consists of 100 people, 50 people, and 460 people, respectively. So, the total is 610 people in the database. Each person has five different expressions, and the number of images is 3050 images. Data sizes vary for each original database, but we did cropping and normalization to form the ground truth images each of size 112 x 112 pixels. The samples of images in the AR database are shown in Fig. 9. Furthermore, to obtain a small image, we did subsampling with a factor of s= 14, and got an 8x8 image. See Fig. 10 for example.

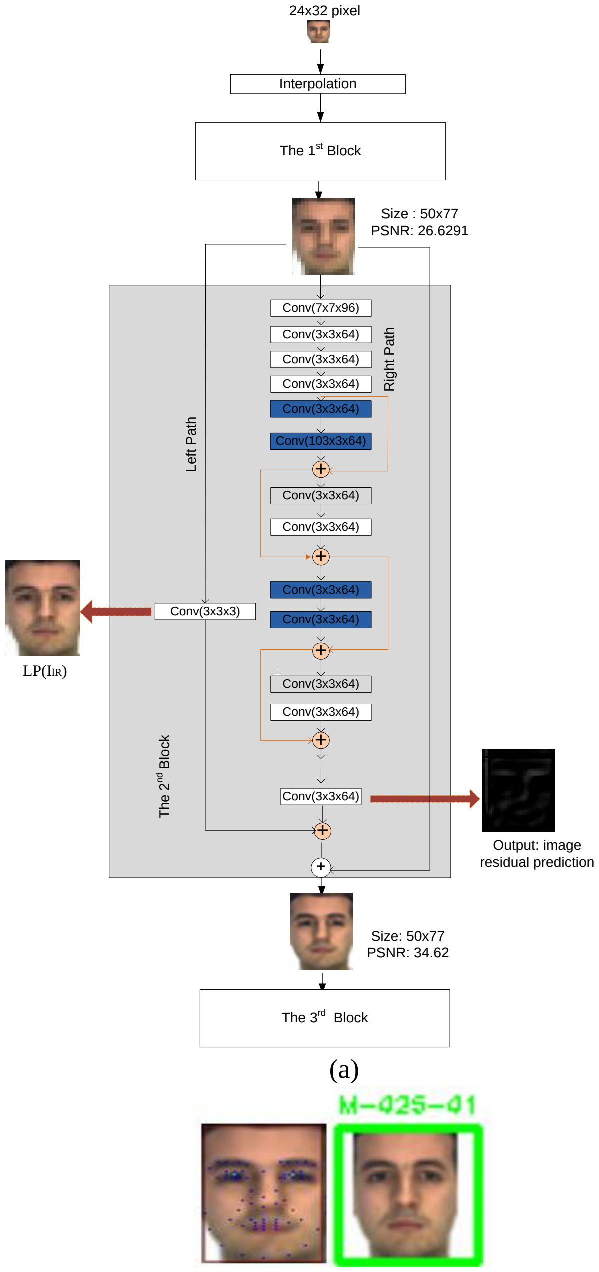
Fig. 9. Samples of ground truth images cropped from the AR database.

Fig. 10. Samples from the results of subsampling. The image resolution is 8x8 pixels.

4.2. Testing

In the training phase mentioned in Section 3, the input of the network receives the high-resolution face images, whereas, in the testing phase, the input is a low-resolution face image. From existing datasets, we have tried various image sizes to be used as test data, such as 6x8 pixels, 8x6 pixels, 8x8 pixels, 16x16 pixels, 20 x20 pixels, and 24x32 pixels, respectively.

As stated before, the purpose of the 1st and 2nd blocks is used to improve the PSNR of the low-resolution face image. For example, suppose the input is a low-resolution face image of size 24x32 pixels, the illustrations of the process on the 1st and 2nd blocks are shown in Fig. 11. Here, the PSNR obtained from the output of the first block is 26.6291 dB and the second block successfully increases the PSNR up to 34.62 dB.



(a)

(b)

Fig. 11. (a) The illustration process on the first and second blocks with the corresponding outputs. (b) Feature detection and face identification of output image.

Obviously, the visualization of the output of the 2nd block is better than that of the 1st block. As an implication, the system can detect the important components of the face, such as the mouth, nose, cheeks, and eyebrows, which are shown in Fig. 11(b). By using the GT database, other samples of the output from the 1st and 2nd blocks are summarized and shown in Fig. 12. Compared to SRCNN, we did improve the PSNR quite a lot.

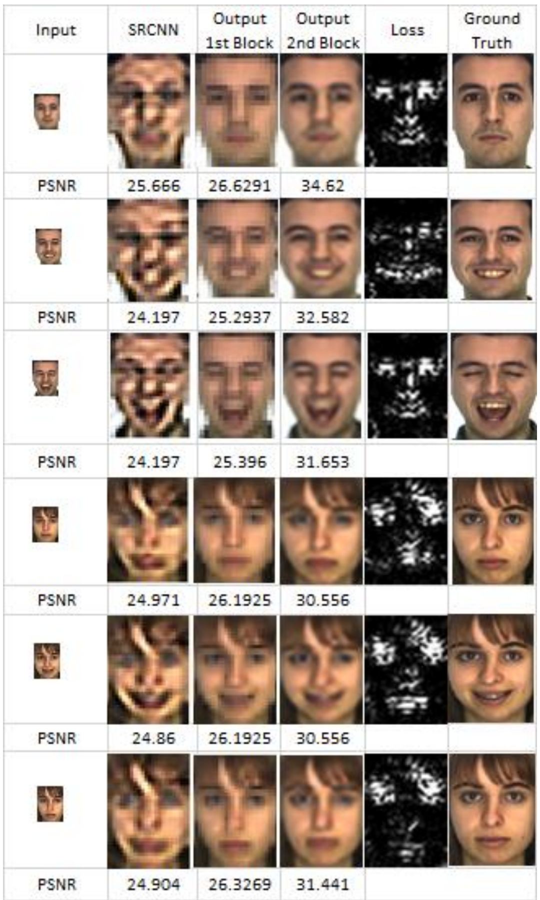

Input
SRCNN
Output 1st Block
Output 2nd Block
Loss
Ground Truth
PSNR 25.666 26.6291 34.62
PSNR 24.197 25.2937 32.582
PSNR 24.197 25.396 31.653
PSNR 24.971 26.1925 30.556
PSNR 24.86 26.1925 30.556
PSNR 24.904 26.3269 31.441

Fig. 12. Some samples from the GT database. Compared to SRCNN, the PSNR was improved after the first and second blocks.

4.3. Recognition Rate

The main objective of this research is to develop an architecture of CNNs which can improve the identification accuracy of the very small size of face images. Here, the accuracy is the rank-1 recognition as described in [69] [70].

To verify the recognition rate, we conducted two experimental settings using open-set scenario for the third block. First, we conducted training using high-resolution (HR) images and then testing using low-resolution (LR) images. Second, we conducted training using LR images and testing using the LR image. The size of the watchlist for both experiments is 610. As listed in the literatures [5, 10, 16, 20, 28, 41, 63, 71, 72, 73], the largest size of LR images is 36 x36 pixels. Hence, for simplicity, any face image whose size is less than 36 x36 pixels is classified into an LR image in this paper.

A. Training using high-resolution (HR) images and then testing using low-resolution (LR) images

In this experiment, the HR image that we used for training has 112x112 pixels. We obtained this image using crop operations on the face area of the HR images from the datasets, and then we did the scaling process to get the face image of size 112x112 pixels. That is, as mentioned in Section 4.1, the databases used were AR, GT, and LFW together. Originally, the dimensions of images in these three databases are 576x768 pixels, 640x480 pixels, and 250x250, respectively. Based on these datasets, as for getting the ground truth face image of size 112x112 pixels, we need to use cropping and subsampling operations. For simplicity, we use the LFW dataset for example. Others can be done similarly. After getting the face area of an image of LFW by OpenCV then cropping it to get the face image of size 182x182 pixels, it then becomes the ground truth face image of size 112x112 pixels after scaling down by a factor of 1.6. Fig. 13 shows an illustration.

Table 4.1. Comparison of testing results for the images of category c1 under Non-ResNet and ResNet.

No	Networks	Training Data on the 3rd Block	Size of Testing Data (pixels)	Identification Accuracy (%)			
				AR DB	GT DB	LFW DB	ALL
1	Non-ResNet	HR	6 x 8	87.5	86	88.8	88.36
2	ResNet	HR	6 x 8	90.5	89	89.46	89.59
3	Non-ResNet	HR	11 x16	92	89	90.87	90.9
4	ResNet	HR	11 x16	95	91	95.33	94.92
5	Non-ResNet	HR	24 x 32	94.5	92	95.11	94.75
6	ResNet	HR	24 x 32	99	95	99.24	98.85

Table 4.2. Comparison of testing results for the images of category c2 between the proposed method and other methods based on deep learning.

Name of Method	Accuracy (%)		
	Size of Image		
	8x8	16x16	20x20
LightCNN [74] in [49]	67.7	86.9	92.7
LightCNN-FT [49]	70.3	88.9	92.9
Coupled-LightCNN [49]	80.0	90.2	93.5
VGGFACE [75] in [49]	75	89.3	93.4
VGGFace-FT [49]	82.3	92.7	94.8
Coupled-VGGFace [49]	83.7	93.1	95.2
ResNet [76] in [49]	72.7	92.3	95.4
ResNet-FT [49]	88.9	95.9	96.8
Trunk network [49]	92.2	95.5	96.8
DCR (Coupled-trunk) [49]	93.6	**96.6**	97.3
Two-Branch DCNN [28]	80.8	**96.7**	98.8
FR-SKD [50]	-	**85.87**	-
VLFRCNN-Non-ResNet(Ours)	90.00	90.90	96.89
VLFRCNN-ResNet(Ours)	93.77	97.05	**98.93**

Furthermore, for the first experiment, the LR images that we used for testing were grouped into c1 and c2 categories, where in c1, the images are 6x8 pixels, 11x16 pixels, and 24x32 pixels, respectively; in c2, the images are 8x8 pixels, 16x16 pixels, and 20 x20 pixels, respectively. Both categorized images were used to test either on the network architecture with residual networks or on network architecture without residual networks. Next, the recapitulation of the testing results for the images of category c1 under network architecture with residual networks (abbrev. to ResNet) and network architecture without residual networks (abbrev. to Non-ResNet) are summarized in Table 4.1, whereas those of category c2 are given in Table 4.2.

B. Training using LR images and testing using LR image

The LR images which we used for training consist of t1 and t2 types. In t1, the size of each face image is 24x24 pixels and in t2, the size is enlarged to 32x32 pixels. Each image of t1 was obtained by subsampling the HR image, then upsampling the image obtained previously and finally using crop operations to obtain the face image. Like Sec. 4.3.A, as for getting the ground truth face images, we only take LFW for example. Others can be obtained similarly. In other words, the original image from the LFW dataset is an HR image of size 250x250 pixels. An image of size 25x25 is obtained after scaling down by a factor of 10 from the previous HR image. Then an image of size 50x50 is obtained by scaling up by a factor of 2 of the previous image. Finally, an LR face image of size 24x24 pixels (an image of t1) is obtained using crop operations on the previous image and based on the bicubic interpolation, it then becomes the ground truth face image of size 112x112 pixels. The images of t2 were obtained similarly. Fig. 13 shows an illustration.

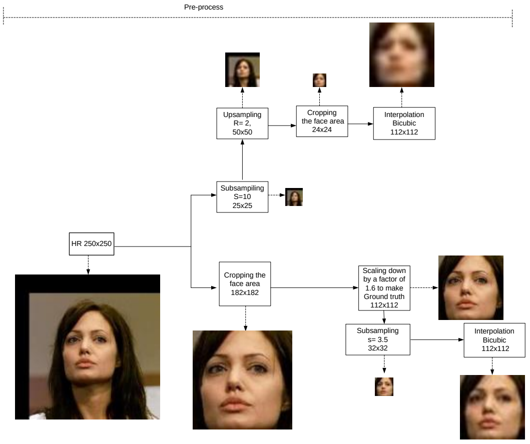

Pre-process
Upsampling R= 2, 50x50
Cropping the face area 24x24
Interpolation Bicubic 112x112
Subsampiling S=10 25x25
HR 250x250
Cropping the face area 182x182
Scaling down by a factor of 1.6 to make Ground truth 112x112
Subsampling s= 3.5 32x32
Interpolation Bicubic 112x112

Fig. 13. The preprocess of obtaining HR/LR images as the input for training the third block.

Furthermore, for the second experiment, the LR images that we used for testing were grouped into c3 and c4 categories, wherein c3, the images are 6x8 pixels, 11x16 pixels, and 24x32 pixels, respectively; in c4, the images are 8x6 pixels. Both categorized images were used to test either on ResNet or on Non-ResNet.

After training the 3rd block using t1 LR images (each of size 24x24 pixels), it was then testing using c3 category LR images. On the contrary, for comparison, training the 3rd block using t2 LR images (each of size 32x32 pixels) was testing using c4 category LR images. Both testing results are summarized in Tables 4.3 and 4.4, respectively.

Table 4.3. Comparison of testing results for the training using t1 images and testing using c3 images under Non-ResNet and ResNet.

No	Networks	Training Data On Block 3	Size of Testing Data (pixels)	Identification Accuracy (%)			
				AR DB	GT DB	LFW DB	ALL
1	Non-ResNet	LR	6 x8	86	76	86.96	85.9
2	ResNet	LR	6 x8	88	83	87.93	87.54
3	Non-ResNet	LR	11 x16	87.5	83	89.46	88.61
4	ResNet	LR	11 x16	89.5	87	90.54	90.08
5	Non-ResNet	LR	24 x 32	90	85	93.15	91.97
6	ResNet	LR	24 x 32	93.5	91	97.28	96.15

Table 4.4. Comparison of testing results for the training using t2 images and testing using c4 images under Non-ResNet and ResNet between the proposed method and the image reconstruction method.

Size Image Training (pixels)	Size Image Testing (Pixels)	Method	Accuracy (%)
48x40	8x6	MDS [71]	52
48x40	8x6	VLRFRH [77]	95.18
32x32	8x6	VLFRCNN-Non-ResNet (Ours)	90.41
32x32	8x6	VLFRCNN-ResNet (Ours)	**95.49**

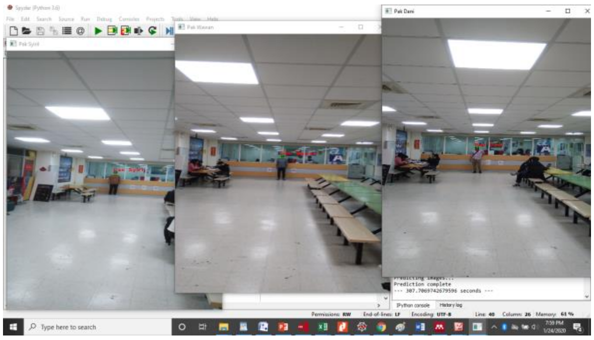
The results of the experiments are analyzed in the following. Tables 4.1 and 4.3 show the results of training using HR images provide a higher level of accuracy, compared to those of training using LR images. This result is contradicting to the results obtained by Wu et al. [16] but is in accordance with the results obtained by Wang et al. [10]. This is because the PSNR of the image produced by [16] has around 30 dB, whereas our method is more than 32 dB. In other words, the image quality produced by our method is better than that produced by [16]. This has the implication that the features got from the image reconstructed from our method are closer to those obtained from the HR image (original). Whereas, the features of the image produced by [16] are closer to those of the LR image. Hence, during testing, the closer to the features of the HR image, the better the recognition accuracy. Besides, Tables 4.1, 4.2, 4.3, and 4.4 also provide information that the residual network (ResNet) provides higher accuracy results, compared to those obtained from the network without residuals (Non-ResNet). Other results that we can also see from Table 4.2 and Table 4.4 are the methods that we propose, especially on those using residual networks having a higher level of accuracy compared to those obtained from the existing methods based on deep learning and image reconstruction. Note that while the image sizes are set to 6×6, 12×12, 24×24, and 36×36, the accuracies got in [71], [72], and [73] are 57.3%, 87.4%, 90.2%, 92.2%; 60.3%, 84.4%, 88.4%, 91.1%; and 69.4%, 88.5%, 90% and 93%, respectively. Our results outperform those of these existing methods.

C. Testing Using Primary Data Low-Resolution Face

Besides testing using datasets, we also verify the proposed method using primary data. We have taken data from the real environment using a real camera in the mobile phone NOKIA 6. To training, the distance of the HR images taken is around 2-3 m, while to testing, the LR images are taken from 8 m, 10 m, and 15 m away.

For images obtained from real cameras, we apply OpenCV by setting it to detect facial areas up to 16x16 pixels. If a face is found, the face area will be cropped and processed in our proposed framework; otherwise, the image got is enlarged using bicubic operation and using OpenCV to detect the face area again. Repeat this process until a face area is detected.

Total images to train the third block (see the architecture in Fig. 4) are 115 images and for testing, there are 30 images taken from 10 persons. Fig. 14. shows some successful examples of testing results.



```
Predicting images...
Prediction complete
--- 307.7069742679596 seconds ---
```

Fig. 14. The face images were taken 15 m away and were recognized successfully.

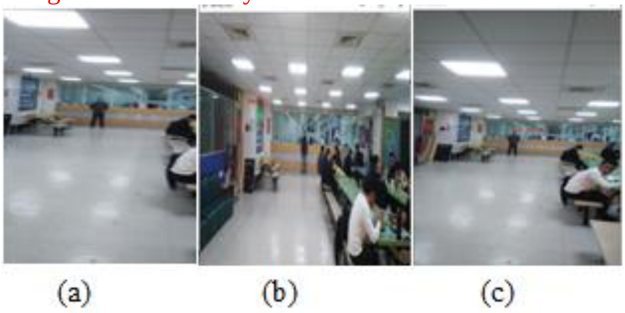


Fig. 15. Some failed examples of recognition: (a) OpenCV failed to detect face area; (b) camera was not focused, and (c) false recognition

From 30 test images, we found 3 images that were not recognized. The failure was caused by (1) OpenCV failed to detect face area (Fig. 15.(a)); (2) camera was not focused (Fig. 15.(b)); and (3) the identification error (Fig. 15.(c)). Fig. 15 shows fail examples of recognition.

4.4. Time Consuming

In this study, the specifications of the computer used to testing were CPU Intel Core i5-6200U, 2.88Hz, and 12GB Memory. The time needed to complete the processing of each image is almost the same for each size, but each block has different times to complete the process. The average time is 3.007 seconds, 1.331242 seconds, and 1.783875 seconds for block 1, block2, and block 3, respectively. Thus, the total average time needed to execute one image is 6.122117 seconds.

5. Conclusions

This research has succeeded to develop a new architecture of Convolution Neural Networks to improve the identification accuracy of face recognition with very low-resolution images. The architecture developed has been classified as deeply-CNN models, and it has more than 30 convolution layers. Furthermore, the proposed method outperforms any exiting methods based on identification accuracy.

Acknowledgments. This research was partially supported by the "Center for Artificial Intelligence, University of Technology Sydney, Australia."

References

[1] M. Turk and A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, vol. 3, No. 1, MIT Press, 1991. pp. 71-86.

[2] F. Jalled, "Face recognition machine vision system using Eigenfaces," 2017, pp. 1-7. Retrieved from https: //arxiv.org/abs/1705.02782.

[3] S. Thakur, J. K. Sing, D. K. Basu, M. Nasipuri, and M. Kundu, "Face recognition using principal component analysis and RBF neural networks," *IJSSST*, Vol. 10, No. 5, pp. 7-15. ISSN: 1473 7 804x online, 1473-8031.

[4] J. Lu, K. N. Plataniotis, A. N. Venetsanopoulos, "Face recognition using LDA based algorithms," *IEEE Trans. Neural Net.*, vol.14, no.1, 2003 pp. 195-200.

[5] M. Jian and K. M. Lam, "Simultaneous hallucination and recognition of low-resolution faces based on singular value decomposition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 11, 2015 pp. 1761-1772.

[6] W. Yang, D. Yi, Z. Lei, J. Sang, and S. Z. Li, "2D-3D face matching using CCA," *IEEE Int. Conf. Autom. Face Gesture Recognition*, 2008, pp. 0-5.

[7] J. Supardi and A. S. Utami, "Development of artificial neural network architecture for face recognition in real time," *Int. J. Mach. Learn. Comput.*, vol. 4, no. 1, 2014, pp. 1-4.

[8] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural network approach," *IEEE Trans. Neural Networks*, vol. 8, no. 1, 1997, pp. 98-113.

[9] Y. M. Lui, D. Bolme, B. A. Draper, J. R. Beveridge, G. Givens, and P. J. Phillips, "A meta-analysis of face recognition covariates," *IEEE 3rd Int. Conf. Biometrics Theory, Appl. Syst. BTAS 2009*, 2009.

[10] Z. Wang, S. Chang, Y. Yang, D. Liu, and T. S. Huang, "Studying very low resolution recognition using deep Networks," *IEEE CVPR*, 2016, pp. 4792-4800.

[11] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for

nonparametric object and scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 11, 2008, pp. 1958-1970.

[12] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," *IEEE CVPR*, 2016, pp. 1646-1645.

[13] P. Autee, S. Mehta, S. Desai, V. Sawant, and A. Nagare, "A review of various approaches to face hallucination," *Procedia Comput. Sci.*, vol. 45, 2015, pp. 361-369.

[14] S. Baker and T. Kanade, "Hallucinating faces," *4th IEEE Int. Conf. Autom. Face Gesture Recognition*, 2000, pp. 83-88.

[15] R. Yang, Y. Wang, D. Yang, T. Xu, J. Zhou. "Face Hallucination via Using the Graph-optimal Locality Preserving Projections". *10th IEEE/ACIS International Conference on Computer and Information Science*, pp. 189-193, 2011.

[16] J. Wu, S. Ding, W. Xu, and H. Chao, "Deep joint face hallucination and recognition," 2016. retrieved in https://arxiv.org/abs/1611.08091.

[17] P. Li, L. Prieto, D. Mery and P. J. Flynn, "On low-resolution face recognition in the wild: Comparisons and new techniques," *IEEE Trans on Information Forensics and Security*, vol. 14, no. 8,.2019, pp. 2000-2012.

[18] J. Yang, S. Member, J. Wright, and T. S. Huang, "Image super-resolution via sparse representation," *IEEE Trans. Image Proces.*, vol. 19, no. 11, 2010, pp. 2861-2873.

[19] C. Dong, C. Loy, K. He, & X. Tang., "Image super-resolution using deep convolutional networks," 2015, pp.1-14. https://arxiv.org/pdf/1501.00092.pdf.

[20] C. Liu, H. Y. Shum, and W. T. Freeman, "Face hallucination: Theory and practice," *Int. J. Comput. Vis.*, vol. 75, no. 1, 2007, pp. 115-134.

[21] A. Krizhevsky, Sutskever, I., & G.E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, 2012, pp. 1-9.

[22] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," *Computer Vision–ECCV 2014, 8689*, 2014, pp. 818-833.

[23] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, A. Rabinovich "Going deeper with convolutions," *IEEE CVPR*, 2015, pp. 1-9.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. Retrive from https://arxiv.org/pdf/1512.03385.pdf.

[25] S. Zhu, S. Liu, C. C. Loy, and X. Tang, "Deep cascaded bi-network for face hallucination," *Computer Vision* – ECCV 2016, pp. 614-630.

[26] J. Kim, J. K. Lee, and K. M. Lee, "Deeply-recursive convolutional network for image super-resolution," *IEEE CVPR*, 2016, pp. 1637-1645.

[27] D. Pouliot, R. Latifovic, J. Pasher, and J. Duffe, "Landsat super-resolution enhancement using convolution neural networks and Sentinel-2 for training," *Remote Sens.*, vol. 10, no. 3, 2018, pp. 1-18.

[28] E. Zangeneh, M. Rahmati, and Y. Mohsenzadeh, "Low resolution face recognition using a two-branch deep convolutional neural network architecture," 2017, pp. 1-11. https://arxiv.org/pdf/1706.06247.pdf.

[29] L. Yue, H. Shen, J. Li, Q. Yuan, H. Zhang, and L. Zhang, "Image super-resolution: The techniques, applications, and future," *Signal Processing*, vol. 128, 2016, pp. 389-408.

[30] N. Wang, D. Tao, X. Gao, X. Li, and J. Li, "A comprehensive survey to face hallucination," *Int. J. Comput. Vis.*, vol. 106, no. 1, 2014, pp. 9-30.

[31] S. Baker and T. Kanade, "Limits on super resolution and how to break them," *IEEE Trans. on Pattern Anal. and Machine. Intel.*, vol. 24, No. 9, 2002, pp. 1167-1183.

[32] M. Elad and A. Feuer, "Super-resolution reconstruction of image sequences," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 9, 1999, pp. 817-834.

[33] D. Huang, W. Huang, P. Gu, P. Liu, and Y. Luo, "Image super-resolution reconstruction based on regularization technique and guided filter," *Infrared Phys. Technol.*, vol. 83, 2017, pp. 103-113.

[34] Y. Zhang, Q. Fan, F. Bao, Y. Liu, and C. Zhang, "Single-image super-resolution based on rational fractal interpolation," *IEEE Trans. Image Process.*, vol. 27, no. 8, 2018. pp. 3782-3797.

[35] J. Yu, X. Gao, D. Tao, X. Li, and K. Zhang, "A unified learning framework for single image super-resolution," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 25, no. 4, 2014, pp. 780-792.

[36] Z. Lin, J. He, X. Tang, and C. K. Tang, "Limits of learning-based superresolution algorithms," *Int. J. Comput. Vis.*, vol. 80, no. 3, 2008, pp. 406-420.

[37] P. Svoboda, M. Hradis, D. Barina, and P. Zemcik, "Compression artifacts removal using convolutional neural networks," 2016. Retrieved from https://arxiv.org/abs/1605.00366.

[38] J. F. Dong, Y. Z. Gan, X. J. Mao, Y. B. Yang and C. Shen, "Learning deep representations using convolutional auto-encoders with symmetric skip connections," *ICASSP*, 2018, pp. 3006-3010.

[39] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," 2016 *IEEE CVPR*, pp. 1874-1883.

[40] W. Yang, X. Zhang, Y. Tian, W. Wang, J. H. Xue, Q. Liao, "Deep learning for single image super-resolution: A brief review," IEEE Trans. on Multimedia, vol. 21, no. 12, 2019, pp 3106-3121.

[41] J. Sun, N. N. Zheng, H. Tao, and H.Y. Shum, "Image hallucination with primal sketch priors," *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, no. c, pp. II-729–36, 2003.

[42] Z. Xiong, X. Sun, and F. Wu, "Image hallucination with feature enhancement," *IEEE CVPR 2009*, pp. 2074-2081.

[43] Y. Li, C. Cai, G. Qiu, and K. M. Lam, "Face hallucination based on sparse local-pixel structure,"

Pattern Recognit., vol. 47, no. 3, 2014, pp. 1261-1270.

[44] L. An and B. Bhanu, “Face image super-resolution using 2D CCA,” *Signal Processing*, vol. 103, 2014, pp. 184-194.

[45] C. Y. Yang, S. Liu, and M. H. Yang, “Structured face hallucination,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 1099-1106.

[46] D. Huang and H. Liu, “Biometric recognition,” *Proc. of 11th Chinese Conf. CCBR*, 2016, pp. 167-175.

[47] Abdu Rahiman V., Jiji, C. V, “Face hallucination using eigen transformation in transform domain,” *Int. J. Image Process.*, vol. 3, no. 6, 2010, pp. 265-282.

[48] M. S. Shakeel, K. M. Lam, S. C. Lai, “Learning sparse discriminant low-rank features for low-resolution face recognition,” *J. Vis. Commun. Image Represent.*, vol. 63, 2019, pp. 1-17.

[49] Z. Lu and X. Jiang, “Deep coupled ResNet for low-resolution face recognition,” *IEEE Signal Processing Letters*, vol. 25, no. 4, 2018, pp. 526-530.

[50] S. Ge, S. Member, S. Zhao, C. Li, J. Li, and S. Member, “Low-resolution face recognition in the wild via selective knowledge distillation,” *IEEE Trans. Image Proces.*, vol. 24, no. 8, 2019. pp. 2051-2062.

[51] R. A. Farrugia and C. Guillemot, “Face hallucination using linear models of coupled sparse support,” *IEEE Trans. Image Proces.*, vol. 26, no. 9, 2017, pp. 4562-4577.

[52] S. J. Horng, J. Supardi, and T. Li, “Appling both hybrid restricted Boltzmann machine and deep convolution neural networks to low-resolution face image recognition,” *J. Comput. Eng. Inf. Technol.*, vol. 07, 2018, p. 9307.

[53] J. Supardi and S. Horng, “Very small image face recognition using deep convolution neural networks,” *J. Phys.: Conf. Ser.* 1196 012020, 2019.

[54] J. Wu, “Introduction to convolutional neural networks,” 2017, pp. 1-31. Retrieve from https://pdfs.semanticscholar.org/450c/a19932fcef1ca6d0442cbf52fec38fb9d1e5.pdf.

[55] Y. LeCun, K. Kavukcuoglu, and C.Farabet, “Convolutional networks and applications in vision,” *IEEE International Symposium onCircuits and Systems: Nano-Bio Circuit Fabrics and Systems*, 2010, pp. 253-256.

[56] J. Bouvrie, “Notes on convolutional neural networks,” In Practice, 2006, pp. 47-60.

[57] Z. Zhang, “Derivation of backpropagation in convolutional neural network,” 2016. Retrieved from https://pdfs.semanticscholar.org/5d79/11c93dd.cb34cac088d99bd0cae9124e5dcd1.pdf.

[58] Y. LeCunn, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no, 11, Nov. 1998, pp. 2278-2324.

[59] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” *ICLR*, 2015. pp. 1-15.

[60] T. Vu, C. V. Nguyen, T. X. Pham, T. M. Luu, and C. D. Yoo, “Fast and efficient image quality enhancement via desubpixel convolutional neural networks.” *Computer Vision – ECCV 2018 Workshops*, pp 243-259.

[61] M. Maire, C. Fowlkes, and J. Malik, “Contour detection and hierarchical image segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, 2011, 2011, pp. 898-916.

[62] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” 2012, pp. 1-18. Retrieve from https://arxiv.org/pdf/1207.0580.pdf.

[63] Yu, X., Fernando, B., Hartley, R., Porikli, F., “Super-resolving very low-resolution face images with supplementary attributes,” *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 908-917.

[64] A. R. Martinez and R. Benavente, “The AR face database,” *CVC*, Barcelona, Spain, Tech. Rep. 24, 1998. Retrieved from https://www2.ece.ohiostate.edu/~aleix/ARdatabase.html.

[65] Georgia Inst. Technol, “GT face database,” Atlanta. http://www.anefian.com/research/face_reco.htm.

[66] G. B. Huang, V. Jain, and E. Learned-Miller, “Unsupervised joint alignment of complex images,” *IEEE Int. Conf. ComputerVision*, 2007.

[67] G. B. Huang, M. A. Mattar, H. Lee, and E. Learned-Miller, “Learning to Align from Scratch,” *Proc. Neural Inf. Processing Systems*, 2012, pp. 1-9.

[68] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, and J. Zhou, “CosFace: Large margin cosine loss for deep face recognition,” 2018. Retrieve from https://arxiv.org/pdf/1801.09414.pdf”

[69] P. J. Phillips, H. Moon, S. Rizvi, and P. Rauss. “The FERET evaluation methodology for face-recognition algorithms,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 22, No. 10. 2000, pp. 1090-1104.

[70] P. J. Phillips, D. Blackburn, P. Grother, E. Newton, J. M. Bone. Methods for Assessing Progress in Face Recognition. In: J. Wayman, A. Jain, D. Maltoni, D. Maio (eds). Biometric Systems. Springer, London. 2005. pp. 207-240.

[71] S. Biswas, K. W. Bowyer, P. J. Flynn, and S. Member, “Multidimensional scaling for matching low-resolution face images,” *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 34, no. 10, 2012, pp. 2019-2030.

[72] H. Huang and H. He, "Super-resolution method for face recognition using nonlinear mappings on coherent features," *IEEE Transactions on Neural Networks*, vol. 22, no. 1, 2011, pp. 121-130.

[73] W. W. Zou and P. C. Yuen, "Very low resolution face recognition problem," *IEEE Trans. Image Process.*, vol. 21, no. 1, 2012.pp. 327-340.

[74] X. Wu, R. He, S. Member, Z. Sun, and T. Tan, "A light CNN for deep face representation with noisy labels," *IEEE Trans. Info. Forensics and Sec.*, vol. 13, no. 11, 2018, pp. 2884-2896.

[75] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," *Proc. of the British Machine Vision Conference (BMVC)*, 2015, pp. 41.1-41.12.

[76] L. Shen, Z. Lin, and Q. Huang, "Relay backpropagation for effective learning of deep convolutional neural networks," *Computer Vision* – ECCV 2016, pp. 467-482.

[77] M. C. Yang, C. P. Wei, Y. R. Yeh, and Y. C. F. Wang, "Recognition at a long distance: Very low resolution face recognition and hallucination," *Proc. 2015 Int. Conf. Biometrics, ICB 2015*, pp. 237-242.

[78] https://towardsdatascience.com/understanding-different-loss-functions-for-neural-networks-dd1ed0274718

Shi-Jinn Horng received the Ph.D. degree in computer science from National Tsing Hua University, in 1989. He is currently a Chair Professor in the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology. He has published more than 200 research papers and received many awards. Especially, the Distinguished Research Award got from the National Science Council in Taiwan in 2004. His research interests include deep learning, biometric recognition, image processing, and information security.

Julian Supardi He is a lecturer in Universitas Sriwijaya, Palembang, South Sumatera, Indonesia. He is currently a Ph.D. candidate and looking forward to his degree in computer science and information engineering at the National Taiwan University of Science and Technology. His research interests include image processing and deep learning.

Professor **Wanlei Zhou** is currently the Head of School of Computer Science at the University of Technology Sydney (UTS). He received the Ph.D. degree from The Australian National University, Canberra, Australia, in 1991. He also received a DSc degree (a higher Doctorate degree) from Deakin University in 2002. His research interests include security and privacy, bioinformatics, and e-learning. Professor Zhou has received over 10 ARC grants in the last 10 years and has published more than 400 papers in refereed international journals and refereed conferences proceedings. Prof Zhou is a Senior Member of the IEEE.

Dr. **Chin-Teng Lin** received the Ph.D. degree in electrical engineering from Purdue University, USA in 1992. He is currently the Distinguished Professor of Faculty of Engineering and Information Technology, and Co-Director of Center for Artificial Intelligence, University of Technology Sydney, Australia. Dr. Lin was elevated to be an IEEE Fellow in 2005, and was elevated International Fuzzy Systems Association (IFSA) Fellow in 2012. Dr. Lin received the IEEE Fuzzy Systems Pioneer Awards in 2017. He served as the Editor-in-chief of IEEE Transactions on Fuzzy Systems from 2011 to 2016. He served as the Deputy Editor-in-Chief of IEEE Transactions on Circuits and Systems-II in 2006-2008. He has published more than 300 journal papers in the areas of neural networks, fuzzy systems, brain computer interface, multimedia information processing, and cognitive neuro-engineering.

JIANG, Bin, Associate Professor. He is the dean of Department of Information Engineering, College of Computer Science and Electronic Engineering, Hunan University, China. He received the Doctor of Engineering Degree from Tokyo Institute of Technology, Japan. His research interests include Big Data Technology, Artificial Intelligence, Machine Vision, Machine Learning, Data Mining, Intelligent Computing, Recommender System, Social Computing and etc. He is a member of CCF, CAAI, IEEE, and ACM.