Reconstruction Low- Resolution Image Face Using Restricted Boltzmann Machine

Julian Supardi [a,1*]

[a] Departement of Informatics Engineering, Univerisity Sriwijaya, Palembang, Sumatera Selatan
[1] julian@unsri.ac.id

* corresponding author

ARTICLE INFO

Article history
Received Jan 4, 2023
Revised Jan 16, 2023
Accepted Feb 26, 2023

Keywords
Low-Resolution
Deep learning
Restricted Boltzmann Machine

ABSTRACT

Low-resolution (LR) face images are one of the most challenging problems in face recognition (FR) systems. Due to the difficulty of finding the specific features of faces, the accuracy of face recognition is low. To solve this problem, some researchers are using an image reconstruction approach to improve the resolution of their images. In this research, we are trying to use the restricted Boltzmann machine (RBM) to solve the problem. Furthermore, a labelled face in the wild (lfw) database has been used to validate the proposed method. The results of the experiment show that the PSNR and SSIM of the image result are 34.05 dB and 96.8%, respectively.

1. Introduction

The sciences of image processing and computer vision are both faced with the crucial and difficult challenge of FR [1]. Until now, FR is still in its infancy and is currently the subject of research, despite having been researched for more than 30 years [2]. Face recognition systems still need to be improved, even if some approaches have achieved outstanding levels of accuracy. This is because numerous situations can result in erroneous detection because of problems like poor lighting, incorrect orientation, twisted expressions, etc.

One of the major issues with image processing and pattern recognition is low resolution (LR)[3]. Due to the small difference in pixel values in the face area and the difficulty in identifying the features of the face image, it is difficult to locate the face in FR. A LR image is typically obtained from a digital scan, an expanded image, a blurry or out-of-focus image, video surveillance, etc. Figure 1 displays a LR face image as an illustration. Several techniques have been suggested in the literature to address this issue [4-9]. The principle of the currently used approaches is, in general, to reconstruct the LR image into a super-resolution image. This is a straight forward solution to the issue. However, each of these approaches has certain drawbacks, and additional study is still required to solve some of the issues mentioned in [10-11].

Figure 1. A low resolution face image. A person is sitting far away from the surveillance camera (the image taken from internet).

A Restricted Boltzmann machine (RBM) is a stochastic graph model that can learn a probability distribution over its set with n visible unit inputs and m hidden feature units [12-13]. Generally, the training mechanism of RBM is unsupervised learning. In this case, the learning of RBM does not require the target output as a thing to be achieved in the learning process. The termination of the learning process is based on the number of repetitions for each of the training data. RBMs have successfully been used to solve many problems, such as dimension reduction [14], classification [15], collaborative filtering [16], feature learning [17-18], and modeling [19]. A detailed knowledge of RBM and deep architecture can be found in [20-21]. RBMs can be used to improve image quality, as shown in [22].

We propose a method for solving the LR face problem using the Restricted Boltzmann Machine (RBM) in this study. The method proposed employs unsupervised learning. In this instance, RBM is utilised to reconstruct low-resolution (LR) visage images into high-resolution (HR) images.

The remainder of this paper is structured as follows: in Section 1's introduction, we discuss existing work and motivation. The second section provides a concise overview of the Boltzmann machine, the restricted Boltzmann machine, the proposed procedure, the results and discussion, and the conclusion.

2. Preliminaries

A Review of Boltzmann machine (BM) and restricted boltzmann machine (RBM) resulting from literatures [12][21][22-26]] is given here..

A. A brief BM

A symmetrically connected network is referred to as a BM [24]. There are no gaps in the network's connectivity, and every link connecting two neurons is symmetrical. Therefore, for each pair, the impact of one neuron on another is symmetrical. Figure 2 (a) depicts the Boltzmann Machine's architectural layout.

For a particular state $x = [x_1, x_2, \dots, x_d]^T$ of the network, the probability as defined by the energy of a BM is postulated as:

$$E((x|\theta) = -\sum_i \sum_{j>i} w_{ij} x_i x_j - \sum_i b_i x_i, \quad (1)$$

where θ denotes parameters of the network consisting of a weight matrix $W = [w_{ij}]$ and bias vector $b = [b_i]$. w_{ij} is the weight of the symmetric connections between neurons i and j.

We assume $w_{ij} = 0$ and $w_{ij} = w_{ji}$. Furthermore, the probability of state x is then

$$P(x|\theta) = \frac{1}{Z(\theta)} exp[-E(x|\theta)], \quad (2)$$

where $Z(\theta) = \sum_x exp[-E(x|\theta)]$

is the normalizing constant.

It follows from Equation (2) that the conditional probability of a single neuron being either 0 or 1 given the states of the other neurons can be written in the following way [25]:

$$P(x_i = 1|x_{\setminus i}, W) = \frac{1}{1+exp(-\sum_{j\neq i} w_{ij}x_j - b_i)}, \quad (3)$$

where x_i denotes a vector $[x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_d]^T$

The process of training the parameters of a BM uses standard likelihood estimation. If given data $\{v^t\}_{t=1}^N$, the log-likelihood of parameters of a BM is:

$$\mathcal{L}(\theta) = \sum_{t=1}^{N} \log P(v^t|\theta) \quad (4)$$

$$= \sum_{t=1}^{N} log \sum_h P(v^t, h|\theta),$$

where the samples $v^t$are assumed to be independent from each other, and the states h of the hidden neurons have to be marginalized out.

Now, we use the partial derivative of (4) with respect to parameters w_{ij} to determine the gradient of the log-likelihood:

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = \frac{N}{2}\left[\langle v_i h_j\rangle_{data} - \langle v_i h_j\rangle_{reconstruction}\right], \quad (5)$$

where the shorthand notation $\langle . \rangle_{P(.)}$ denotes the expectation computed over the probability distribution $P(.)$. Additionally, data $d$and reconstruction $m$are used for denoting two probability distribution $P\left(h|\{v^{(t)}\}, \theta\right)andP(x|\theta)$, respectively.

The overall update formula for a parameterw_{ij} is

$$\Delta w_{ij} = \epsilon\left(\left(v_i h_j\right)_{data} - \left(v_i h_j\right)_{reconstruction}\right) \quad (6)$$

$$w_{ij}^{(k+1)} = w_{ij}^{(k)} + \eta^{(k)} \Delta w_{ij}, \quad (7)$$

where η denotes the learning rate.

Let b be a vector of the biases b_i of the visible neurons exclusively from this point on, and let c be a vector of the biases of the hidden neurons, respectively. The update rules are then analogous to the update rule for the weights and are used to separate the biases of visible and hidden neurons.

$$b_i = b_i + \eta[\langle v_i\rangle_{data} - \langle v_i\rangle_{recontruction}] \quad (8)$$

And

$$c_j = b_i + \eta\left[\langle h_j\rangle_{data} - \langle h_j\rangle_{reconstruction}\right], \quad (9)$$

where v_i, h_j ,b_i , and $c_j$are the $i-th$ visible neuron, the $j-th$ hidden neuron, the $i-th$ visible bias, and the $j-th$ hidden bias, respectively.

B. A brief Review about RBM

Figure 2(b) illustrates the structure of the RBM used in this study, which is in accordance with [12]. There are two layers to it. a surface layer and an interior layer. The input of the RBM is connected to the visible layer.

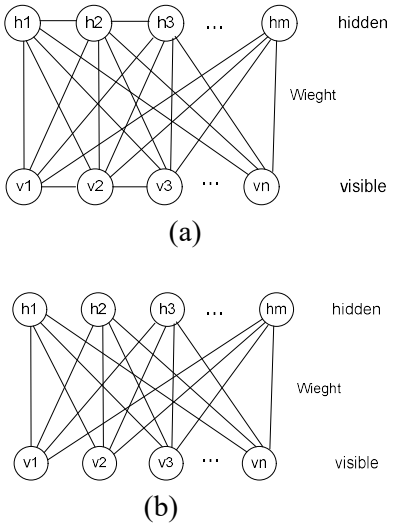


Figure 2. The BM and RBM graphs (a), each having n visible layers and m hidden layers. In BM, it has links between variables in the same layer as well as between the layer of hidden and visible variables. (b) Only links between the layers of hidden and transparent variables are present in RBM.

The energy function of an RBM is defined as:

$$E(v,h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j v_i w_{i,j} h_j \quad (10)$$

Or

$$E(v,h) = -b'v - c'h - h'Wv, \quad (11)$$

where $W = (w_{i,j})$ is the weight connecting hidden and visible units, and b' and c' are the offsets of the visible and hidden layers, respectively.

We can write two independent probability variables since the variables in both levels are independent. [27]:

$$p(h|v) = \prod_{i=1}^{n} p(h_i|v) \quad (12)$$

$$p(v|h) = \prod_{i=1}^{m} p(v_i|h) \quad (13)$$

It is simple to calculate the marginal distribution because the hidden and visible variables are independent [28].

$$p(v) = \frac{1}{Z}\sum_{h} p(v,h) = \frac{1}{Z}\sum_{h} e^{-E(v,h)}$$

$$= \frac{1}{Z}\sum_{h_1}\sum_{h_2}\dots\sum_{h_m} e^{\sum_{j=1}^{m} b_j v_j} \prod_{i=1}^{n} e^{h_i\ (\ c_i\ +\sum_{j=1}^{m} w_{ij} v_j)}$$

$$= \frac{1}{Z} e^{\sum_{j=1}^{m} b_j v_j} \sum_{h_1} e^{h_1\ (\ c_1\ +\sum_{j=1}^{m} w_{1j} v_j)} \sum_{h_2} e^{h_2\ (\ c_2\ +\sum_{j=1}^{m} w_{2j} v_j)} \dots$$

$$\dots \sum_{h_n} e^{h_n\ (\ c_n\ +\sum_{j=1}^{m} w_{nj} v_j)}$$

$$= \frac{1}{Z} e^{\sum_{j=1}^{m} b_j v_j} \prod_{i=1}^{n}\sum_{h_i} e^{h_i\ (\ c_i\ +\sum_{j=1}^{m} w_{ij} v_j)}$$

$$= \frac{1}{Z}\prod_{j=1}^{m} e^{b_j v_j} \prod_{i=1}^{n}(1 + e^{c_i\ +\sum_{j=1}^{m} w_{ij} v_j}), \qquad \textbf{(14)}$$

where $z = \sum_x e^{-E(x)}$ is a normalization constant and E(x) is an energy function [14].

This equation shows why an (marginalized) RBM can be regarded as a product of experts model [26-27], in which a number of "experts" for individual components of the observations are combined multiplicatively.

Any distribution on $\{0,1\}^m$ can be modeled arbitrarily well by an RBM with m visible and $k + 1$ hidden units, where k denotes the cardinality of the support set of the target distribution, that is, the number of input elements from $\{0,1\}^m$ that have a non-zero probability of being observed [28]. It has been shown recently that even less units can be sufficient depending on the patterns in the support set [29].

In [28] it is explained that the RBM can be interpreted as a stochastic neural network, where nodes and edges correspond to neurons and synaptic connections, respectively. The conditional probability of a single variable being 1 can be interpreted as the firing rate of a (stochastic) neuron with sigmoid activation function $\sigma(x) = \frac{1}{(1+e^{-x})}$, because it holds:

$$p(H_i|v) = \sigma(\sum_{j=1}^{m} w_{ij} v_j + c_i) \qquad \textbf{(15)}$$

$$p(V_j|h) = \sigma(\sum_{i=1}^{n} w_{ij} h_i + b_j) \qquad \textbf{(16)}$$

To see this, let v_{-l} denote the state of all visible units except the $l-$th one and let us define [28]:

$$\alpha_l(h) = -\sum_{i=1}^{n} w_{li}\, h_i - b_l \quad (17)$$

$$\beta(v_{-1}, h) = -\sum_{i=1}^{n}\sum_{j=1, j\neq l}^{m} w_{ij} h_i\, v_j - \sum_{j=1, j\neq l}^{m} b_i v_i - \sum_{i=1}^{n} c_i h_i \quad (18)$$

Now, we can write Equation (10) as

$$E(v_{-l}, h) = \beta(v_{-l}, h) + v_l \alpha_l(h) \quad (19)$$

where $v_l \alpha_l(h)$ collects all terms involving v_1 and we can write [27]:

$$p(V_l = 1|h) = p(V_l = 1|v_{-1}, h) = \frac{p(V_l = 1, v_{-1}, h)}{p(v_{-1}, h)}$$

$$= \frac{e^{-E(v_l=1, v_{-l}, h)}}{e^{-E(v_l=1, v_{-l}, h} + e^{-E(v_l=0, v_{-l}, h}} = \frac{e^{-\beta(, v_{-l}, h) - 1.\alpha_l(h)}}{e^{-\beta(, v_{-l}, h) - 1.\alpha_l(h)} + e^{-\beta(, v_{-l}, h) - 0.\alpha_l(h)}}$$

$$= \frac{e^{-\beta(, v_{-l}, h)}.e^{-\alpha_l(h)}}{e^{-\beta(, v_{-l}, h)}.e^{-\alpha_l(h)} + e^{-\beta(, v_{-l}, h))}} = \frac{e^{-\beta(, v_{-l}, h)}.e^{-\alpha_l(h)}}{e^{-\beta(, v_{-l}, h)}(e^{-\alpha_l(h)} + 1)}$$

$$= \frac{e^{-\alpha_l(h)}}{e^{-\alpha_l(h)} + 1} = \frac{\frac{1}{e^{\alpha_l(h)}}}{\frac{1}{e^{\alpha_l(h)}} + 1} = \frac{1}{1 + e^{\alpha_l(v_{-1}, h)}}$$

$$= \sigma(-\alpha_l(h) = \sigma(\sum_{i=0}^{n} w_{il} h_i + b_j) \quad (20)$$

We follow [30]Samples of $P(x)$ can be obtained by running a Markov chain to convergence, using Gibbs sampling as the transition operator. Detailed description about Gibbs Sampling on RBM can be seen in [31-32]. The illustration of Gibbs sampling is shown in Figure 5 and the mechanism for counting follows from Algorithm 1.

Gibbs sampling of the collection of N random variables $S = \{S_1, S_2, \ldots, S_N\}$ is done through a sequence of$N$ sampling sub-steps of the form $S_i \sim P(S_i|S_{-i})$ where S_{-i} contains the $N-1$ other random variables in S excludingS_i .

Since S for RBMs is conditionally independent and consists of the set of visible and hidden units, block Gibbs sampling is possible. Here, the hidden units' fixed values are sampled concurrently with the visible units' samples, and vice versa. Thus, the following action is conducted in the Markov chain [30]:

$$h^{(n+1)} \sim sigm(W' v^{(n)} + c) \quad (21)$$

$$v^{(n+1)} \sim sigm(W'h^{(n)} + b) \quad \textbf{22}$$

where $h^{(n)}$ refers to the set of all hidden units at the $n-th$ step of the Markov chain. What it means is that, for example, $h_i^{(n+1)}$is randomly chosen to be $\mathbf{1}$ (versus 0) with probability $sigm(W_i' v^{(n)} + c_i)$, and similarly,$v_j^{(n+1)}$ is randomly chosen to be 1 (versus 0) with probability $sigm(W_j' h^{(n+1)} + b_j)$.

3. Methodology

In this study, to solve low-resolution face image problems generally, the architecture of RBM is shown in figure 3. This method consists of two phases, namely a learning stage and a testing phase. In the learning phase, the focus is on determining the weights on the RBM network to perform single image reconstruction, while in the testing phase, RBM is used as a medium to transform LR image input into an image with high resolution.

As an illustration of the proposed method shown in Figure 3.

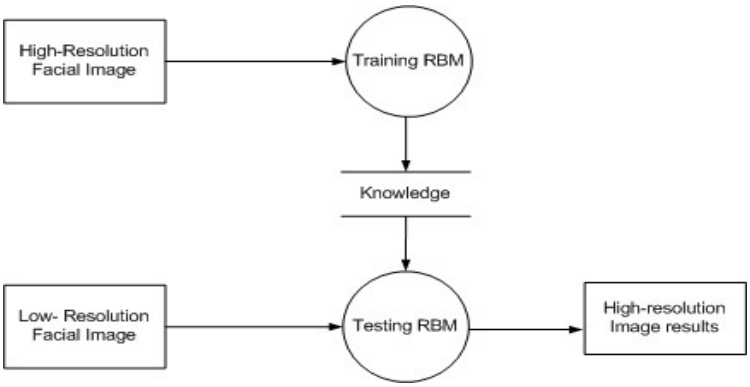


Figure 3. The scheme of the proposed method.

The following is a summary of how the suggested approach operates: Learning the RBM Net is the first step. Unsupervised learning is used to do this. The weight and the outcome of the picture reconstruction are the two sorts of output for this RBM. The second phase is the rebuilt image using RBM when the learning of the RBM is finished. The specific justification is as follows:

A. Training RBM

Suppose $I$is a low resolution face image of size $m\ x\ n$. Let $I_r, I_g,$ and I_B, each of size $m\ x\ n$ be the three color channels (red, green, and blue) of I. Each of them is represented as:

$$I_{r(m,n)} = \begin{vmatrix} I_{r(11)} & \cdots & I_{r(1n)} \\ \cdots & \cdots & \cdots \\ I_{r(m1)} & \cdots & I_{r(mn)} \end{vmatrix}$$

$$I_{g(m,n)} = \begin{vmatrix} I_{g(11)} & \cdots & I_{g(1n)} \\ \cdots & \cdots & \cdots \\ I_{g(m1)} & \cdots & I_{g(mn)} \end{vmatrix}$$

$$I_{b(m,n)} = \begin{vmatrix} I_{b(11)} & \cdots & I_{b(1n)} \\ \cdots & \cdots & \cdots \\ I_{b(m1)} & \cdots & I_{b(mn)} \end{vmatrix}$$

Let I'be the image that is reconstructed from image I and it is shown in Figure 4. Let $I'_r, I'_g,$ and I'_b , each of size $m\ x\ n$, be the three color channels of the reconstructed images. Each of them is represented as:

$$I'_{r(m,n)} = \begin{vmatrix} I'_{r(11)} & \cdots & I'_{r(1n)} \\ \cdots & \cdots & \cdots \\ I'_{r(m1)} & \cdots & I'_{r(mn)} \end{vmatrix}$$

$$I'_{g(m,n)} = \begin{vmatrix} I'_{g(11)} & \cdots & I'_{g(1n)} \\ \cdots & \cdots & \cdots \\ I'_{g(m1)} & \cdots & I'_{g(mn)} \end{vmatrix}$$

$$I'_{b(m,n)} = \begin{vmatrix} I'_{b(11)} & \cdots & I'_{b(1n)} \\ \cdots & \cdots & \cdots \\ I'_{b(m1)} & \cdots & I'_{b(mn)} \end{vmatrix}$$

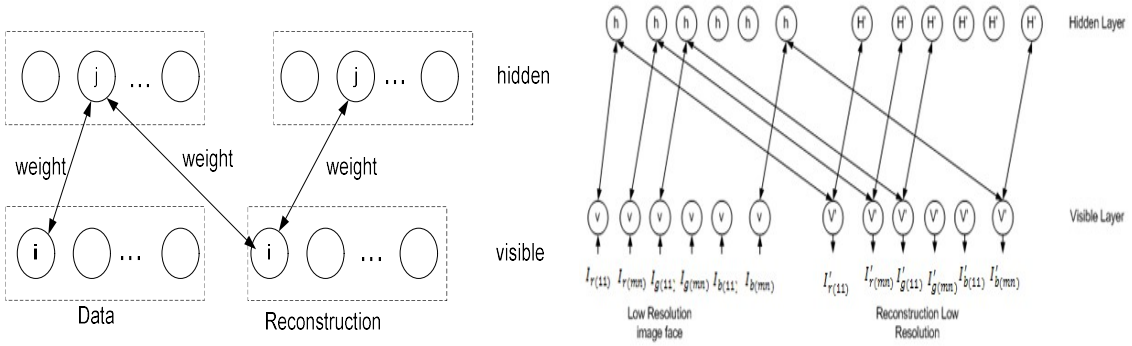


Figure 4. The ilustration of process of reconstructing a LR face image using RBM.

The training rules of RBM is as follows:

$$w_{ij}^{(k+1)} = w_{ij}^{(k)} + \eta^{(k)} \epsilon((v_i h_j)_{data} - (v_i h_j)_{reconstructon} \quad \textbf{(21)}$$

$$b_i = b_i + \eta[\langle v_i \rangle_{data} - \langle v_i \rangle_{reconstruction}] \quad \textbf{(22)}$$

$$c_j = c_j + \eta[\langle h_j \rangle_{data} - \langle h_j \rangle_{reconstruction}] \quad \textbf{(23)}$$

Furthermore, step by step, training of the RBM described in [39] is summarized as follows:

Step 0: Set weight $W = \{w_{ij}\}_{i=0,1,..,m\ ,j=0,1,..,n}$ with small random value and set N as the number of repeats for each data. Initial Count =0.

Step 1: Create a training vector using the units that are visible. The initial layer in RBM is the one that is visible.An RGB image can be fed into the layer.

Step 2: The visible unit, which is the pixel value of the RGB input image and is a sigmoid function, is updated in tandem with all the hidden units.

$$p(h_j = 1) = \sigma(b_j + \sum_i v_i w_{ij}) \quad (24)$$

Step 3: To obtain a reconstruction, update all the discernible units concurrently.

$$p(v_i = 1) = \sigma(c_i + \sum_j h_j w_{ij}) \quad (25)$$

Step 4: Update weight (Update weight until the stop condition is met, which is typically the number of repetitions for each data point).

$$\Delta w_{ij} = \epsilon((v_i h_j)_{data} - (v_i h_j)_{reconstruction} \quad (26)$$

$$w_{ij}^{(k+1)} = w_{ij}^{(k)} + \eta^{(k)} \Delta w_{ij} \quad (27)$$

Step 5: Go to Step 1 until Count $\leq$N.

Step 6: Save W and the visible unit's value.

We can use block Gibbs sampling since each visible unit and concealed unit are independent. One Markov Chain Monte Carlo approach that can be used to extract samples from posterior distributions is Gibbs sampling. By tracing each variable (or variable block) for a sample of its conditional distribution and keeping the remaining fixed variable at its actual value, Gibbs sampling aims to create a posterior sample. Figure 5 provides an illustration of Gibbs sampling.

See figure 5, In general, the Gibbs sampling has two steps: first, sampling a new state h for the hidden neurons on $P((h|v)$; then, it is sampling a state v for the visible layer based on $P((v|h)$. For each new state we can use Equations (29) and (30) respectively. Furthermore, the formula is implemented in Algorithm 1 to determine sample.

$$h^{(k+1)} \sim P(h^{(k+1)}|v^{(k)}) \quad (28)$$

$$v^{(k+1)} \sim P(v^{(k+1)}|h^{(k+1)}) \quad (29)$$

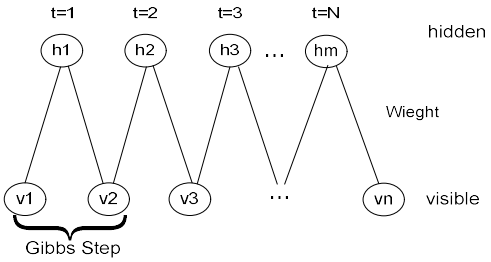


Figure 5. Gibbs sampling.

Table 1. Gibbs sampling steps for RBM

Algorithm 1: Gibbs sampling steps for general

RBM

Draw $h^{(0)}$ uniformly from the state space

Draw $v^{(0)}$ uniformly from the state space

For iteration k=1… d do

Sample$h^{(k)}$using Equation (29)

End for

For iteration k=1 … d do

Sample $v^{(k)}$ using Equation (30)

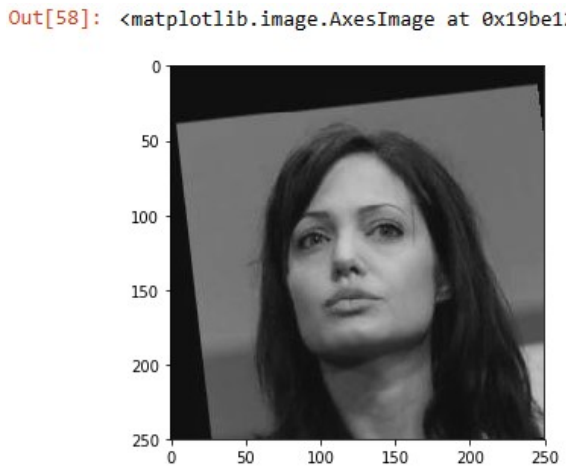
End for

4. Results and Discussion

This study utilised the LFW database [33-34] to evaluate the efficacy of the proposed method. The original image's dimensions are 250 by 250 pixels. Next is the procedure of downsampling to obtain an LR image. In this study, bicubic interpolation is used to conduct downsampling, and the resulting LR image is 32x32 pixels in size. Figures 6(a) and 6(b) depict the image from the data set and the corresponding LR image, respectively.

Figure 6. Example of a data set image: (a) The original dataset; (b) The corresponding LR dataset derived from bicubic downsampling and used as an input to the RBM. The information is derived from the LFW database. Original image dimensions are 250x250 pixels, while LR dimensions are 42x42 pixels.

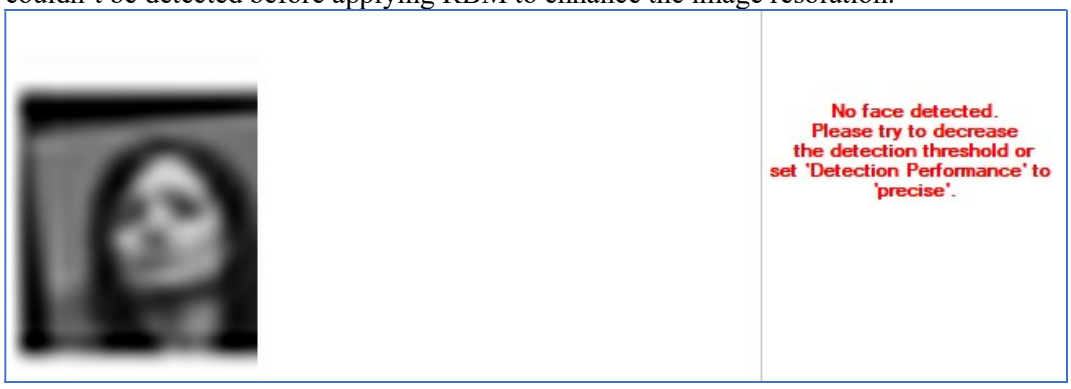
Using the proposed procedure for reconstruction, the low-resolution (LR) image was transformed into a high-resolution (HR) image with a size of 250 x 250 pixels. Figure 7 depicts the result of the procedure. The resulting image reconstruction is of high quality. The PSNR and SSIM values of the image produced by the experiment are 34.05 dB and 96.8%, respectively.



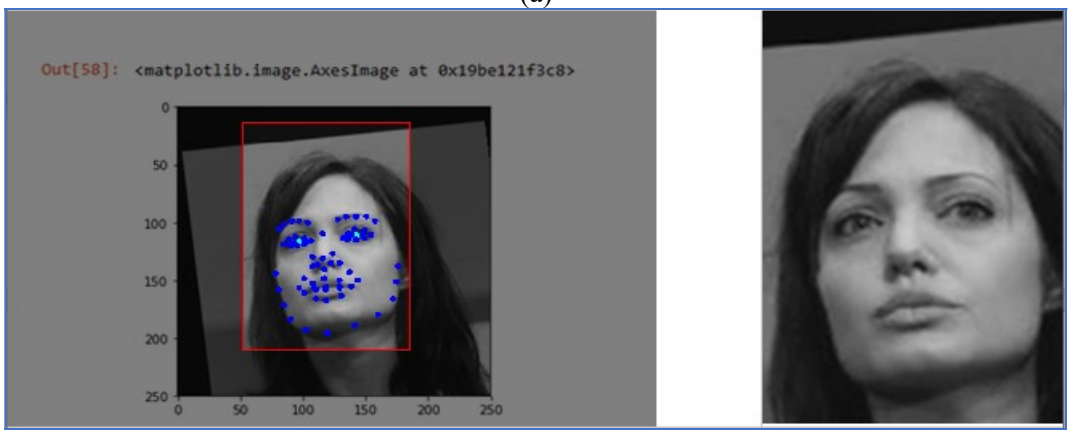
Gambar 7. The Output upsampling scaling x6 (PSNR/ SSIM: 34.05 db/ 96.8)

A. Experiments for face feature extaction detection

Furthermore, to measure the effectiveness of the proposed method, we have tried to detect the feature and reconstruct the facial image. We can compile images. Figure 7.b shows the feature face image can be detected after the reconstruction process using RBM, while in Figure 7.a, the feature image couldn't be detected before applying RBM to enhance the image resolution.



(a)



(b)

Figure 7.(a) detection feature LR face image; (b) detection feature after enhance image resolution using RBM

5. Conclusion

In this research, we propose a new approach to the reconstruction of LR images into HR images using RBM. The impact of the enhancement process on the image feature can be detected.

References

[1] Jian, M., &Lam, K. M. (2015). Simultaneous hallucination and recognition of low-resolution faces based on singular value decomposition. *IEEE Transactions on Circuits and Systems for Video Technology*, *25*(11), 1761–1772.
[2] Li, J., Zhao, B., &Zhang, H. (2009). Face recognition based on PCA and LDA combination feature extraction. *2009 First International Conference on Information Science and Engineering*, (20042013), 1240–1243.
[3] Zou, W. W. W., &Yuen, P. C. (2012). Very low resolution face recognition problem. *IEEE Transactions on Image Processing*, *21*(1), 327–340.
[4] Yang, R., Wang, Y., Yang, D., Xu, T., &Zhou, J. (2011). Face hallucination via using the graph-optimal locality preserving projections. *2011 10th IEEE/ACIS International Conference on Computer and Information Science*, 189–193.
[5] Yang, C. Y., Liu, S., &Yang, M. H. (2013). Structured face hallucination. *Proc. of the IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, 1099–1106.
[6] An, L., & Bhanu, B. (2014). Face image super-resolution using 2D CCA. *Signal Processing*, *103*, 184–194.
[7] Biswas, S., Aggarwal, G., Flynn, P. J., &Bowyer, K. W. (2013). Pose-robust recognition of low-resolution face images, *35*(12), 3037–3049.
[8] Park, J. S., &Lee, S. W. (2008). An example-based face hallucination method for single-frame, low-resolution facial images. *IEEE Transactions on Image Processing*, *17*(10), 1806–1816.
[9] Ren, C. X., Dai, D. Q., &Yan, H. (2012). Coupled kernel embedding for low-resolution face image recognition. *IEEE Transactions on Image Processing*, *21*(8), 3770–3783.
[10] Zhao, W., Chellappa, R., Phillips, P. J., &Rosenfeld, a. (2003). Face recognition: A literature survey. *Acm Computing Surveys*, *35*(4), 399–458.
[11] Wang, Z., Miao, Z., Jonathan Wu, Q. M., Wan, Y., &Tang, Z. (2014). Low-resolution face recognition: A review. *Visual Computer*, *30*(4), 359–386.
[12] Fischer, A., &Igel, C. (2014). Training restricted Boltzmann machines: An introduction. *Pattern Recognition*, *47*(1), 25–39.
[13] Fischer, A., &Igel, C. (2012). An introduction to restricted Boltzmann machines. *Lecture Notes in Computer Science: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, *7441*, 14–36.
[14] Hinton, G. E., Salakhutdinov, R. R.. (2006) . Reducing the dimensionality of data with neural networks. *SCIENCE*, Vol. 313, Issue 5786, 2006, 504-507.
[15] Larochelle, H., &Bengio, Y. (2008). Classification using discriminative restricted Boltzmann machines. *Icml*, 536–543.
[16] Salakhutdinov, R., Mnih, A., &Hinton, G. (2007). Restricted Boltzmann machines for collaborative filtering. *Proceedings of the 24th International Conference on Machine Learning - ICML '07*, 791–798.
[17] Zheng, X., Wu, Z., Meng, H., Li, W., &Cai, L. (2013). Feature learning with Gaussian restricted Boltzmann machine for robust speech recognition. arXiv:1309.6176 [cs.CL]
[18] Tran, S. N., Wolff, D., Weyde, T., &Garcez, A. (2014). Feature preprocessing with RBMs for music similarity learning. *AES 53th*, 1–8.
[19] Ranzato, M. A., &Hinton, G. E. (2010). Factored 3-way restricted Boltzmann machines for modeling natural images. *Artificial Intelligence*, *9*, 621–628.
[20] Salakhutdinov, R., &Hinton, G. (2009). Deep Boltzmann machines. *Aistats*, *1*(3), 448–455.
[21] Bengio. Y.,(2009). Learning deep architectures for AI.*Foundations and Trends®in Machine Learning*. Vol. 2, No. 1 (2009) 1–127
[22] Sahasrabudhe, M., &Namboodiri, A. M. (2014). Fingerprint enhancement using unsupervised hierarchical feature learning. *Proc. of the 2014 Indian Conference on Computer Vision Graphics and Image Processing - ICVGIP '14*, 1–8.
[23] Hinton, G. (2014). Boltzmann machines. *Encyclopedia of Machine Learning and Data Mining*, (1), 1–7.
[24] Fischer, A., &Igel, C. (2014). Training restricted Boltzmann machines. *Pattern Recogn.*, *47*(1), 25–39.

[25] Cho, K. H. (2011). Improved learning algorithms for restricted Boltzmann machines. *Master's thesis, Aalto University School of Science*.
[26] Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, *14*(8), 1771–1800.
[27] Welling, M. Product of experts. Scholarpedia 2(10), 3879 (2007)
[28] LeRoux, N., &Bengio, Y. (2008). Representational power of restricted Boltzmann machines and deep belief networks. *Neural Computation*, *20*(6), 1631–1649.
[29] Montufar, G., &Ay, N. (2010). Refinements of universal approximation results for deep belief networks and restricted Boltzmann machines, (2010), 1–12.
[30] Restricted Boltzmann Machines (RBM) — DeepLearning 0.1 documentation
[31] Yildirim, I. (2012). Bayesian inference: Gibbs sampling, *14627*, 1–6.
[32] Hinton, G. (2010). A practical guide to training restricted Boltzmann machines. *Computer*, *9*(3), 1.
[33] Huang, G. B., Jain, V., &Learned-Miller, E. (2007). Unsupervised joint alignment of complex images. *Proceedings of the IEEE International Conference on Computer Vision*.
[34] Huang, G. B., Mattar, M. A., Lee, H., &Learned-Miller, E. (2012). Learning to align from scratch. *Proc. Neural Information Processing Systems*, 1–9.