

Finger-vein biometric identification using convolutional neural network

Syafeeza AHMAD RADZI^{1,*}, Mohamed KHALIL-HANI², Rabia BAKHTERI²

¹Faculty of Electronics and Computer Engineering, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, Durian Tunggal, Melaka, Malaysia

²VeCAD Research Laboratory, Faculty of Electrical Engineering, Universiti Teknologi Malaysia, Skudai, Johor, Malaysia

Received: 06.11.2013

Accepted/Published Online: 01.07.2014

Final Version: 23.03.2016

Abstract: A novel approach using a convolutional neural network (CNN) for finger-vein biometric identification is presented in this paper. Unlike existing biometric techniques such as fingerprint and face, vein patterns are inside the body, making them virtually impossible to replicate. This also makes finger-vein biometrics a more secure alternative without being susceptible to forgery, damage, or change with time. In conventional finger-vein recognition methods, complex image processing is required to remove noise and extract and enhance the features before the image classification can be performed in order to achieve high performance accuracy. In this regard, a significant advantage of the CNN over conventional approaches is its ability to simultaneously extract features, reduce data dimensionality, and classify in one network structure. In addition, the method requires only minimal image preprocessing since the CNN is robust to noise and small misalignments of the acquired images. In this paper, a reduced-complexity four-layer CNN with fused convolutional-subsampling architecture is proposed for finger-vein recognition. For network training, we have modified and applied the stochastic diagonal Levenberg–Marquardt algorithm, which results in a faster convergence time. The proposed CNN is tested on a finger-vein database developed in-house that contains 50 subjects with 10 samples from each finger. An identification rate of 100.00% is achieved, with an 80/20 percent ratio for separation of training and test samples, respectively. An additional number of subjects have also been tested, in which for 81 subjects an accuracy of 99.38% is achieved.

Key words: Finger vein, convolutional neural network, biometric identification

1. Introduction

The need for security of personal belongings and information is becoming increasingly important. Conventional biometric personal identification systems, which are based on physiological characteristics and behavioral patterns such as faces, irises, or fingerprints, have several drawbacks [1]. The features can be forged due to physical appearance to the human eye. Hence, the features are susceptible to damage or change with time. An example of this type of biometric trait is a fingerprint. Iris recognition is considered as less user-friendly since it can cause discomfort due to the brightness of the light during the biometric capture process [1]. In addition, face recognition accuracy is sensitive to variance in occlusions, poses, facial expressions, and illumination [2]. Recently, a biometric recognition method based on vascular patterns such as hand veins and especially finger veins has gained popularity [3]. Table 1 summarizes the characteristics of the aforementioned methods [1].

*Correspondence: syafeeza@utem.edu.my

Table 1. Summary of characteristics of existing biometric systems [1].

Biometric trait	Main advantage	Defect	Security level	Sensor	Cost
Voice	Natural and convenient	Noise	Normal	Noncontact	Low
Face	Remote capture	Lighting conditions	Normal	Noncontact	Low
Fingerprint	Widely applied	Skin	Good	Contact	Low
Iris	High precision	Glasses	Excellent	Noncontact	High
Finger vein	High security level	Few	Excellent	Noncontact	Low

A finger-vein biometric system captures images of the vein patterns inside a finger, and, like other biometric patterns, these patterns are unique. However, unlike existing biometric techniques such as fingerprint and face, vein patterns are inside the body, hence making them virtually impossible to replicate. Finger-vein features exhibit several other excellent advantages and these include: 1) each person has a unique finger-vein pattern, including identical twins, and vein patterns provide good distinction between individuals; 2) vein patterns are permanent and do not change with time; 3) unlike fingerprint recognition, finger-vein patterns are not easily forged, observed, damaged, or obscured; 4) it is gaining wide acceptance because the capturing device is comparatively small and recognition accuracy is relatively very high compared to other biometric identification methods; 5) finger-vein pattern acquisition is user-friendly, since the images can be captured noninvasively and with contactless sensors, ensuring both convenience and cleanliness for the user; and 6) with ten fingers available on a person, if something unexpected happens to one finger, other fingers can also be used for authentication [3–5].

Despite the advantages mentioned above, there are challenges still outstanding that need to be overcome in order to achieve higher performance required in real-world deployments of finger-vein biometric identification systems. First, the quality of the image is greatly affected by the image acquisition device itself. In a finger-vein acquisition process, the position of the back of the finger and the camera is very close, which causes optical blurring on the captured image [6]. Poor lighting at the capturing device may cause the image to appear extremely dark or bright [7], and if the finger position is not guided, misalignment of the finger position may decrease the recognition rate [8]. Misalignment of finger position can easily cause the matching process to fail. In addition, each individual has different thickness of bones and skin, and since the skin layer is not transparent and uniform, light scattering may occur [5]. The resulting noise has to be removed as much as possible, and conventional finger-vein recognition methods employ complex image processing algorithms to overcome the issues mentioned [9].

Recent research has shown that the above-mentioned problems can be tackled by computational intelligence (CI) methods, which include neural networks, fuzzy logic, and evolutionary computing. Inspired by the human brain, a neural network (NN), typically known as a multilayer perceptron (MLP), works as a powerful classifier. The NN is formed by several processing elements that provide the ability to parallelize processing, and it has the ability to handle noisy images and learn from experience.

A convolutional neural network (CNN) is a NN variant that consists of a number of convolutional layers alternating with subsampling layers and ending with one or more fully connected layers in the standard MLP. A CNN combines image segmentation, feature extraction, and a classifier in one trainable module. It accepts a two-dimensional (2-D) raw image with minimal preprocessing and retains the 2-D topology throughout its training. Classification is performed during training, and at the end of training, the final weights obtained behave as a feature extractor to classify the query input sample. CNNs have been applied to various case

studies such as face recognition [10], handwriting recognition [11], license plate recognition [12], and several other applications.

We now review some previous work applying CI methods in finger-vein biometric identification. In 2006, Zhang et al. [13] proposed a finger-vein identification solution using a MLP. A multiscale feature extraction method based on curvelets was proposed and an accuracy of 99.87% was obtained on a database of 400 subjects. In 2009, Wu and Ye [1] proposed a finger-vein driver identification system using a probabilistic NN. Vein region segmentation and gamma enhancement algorithms were applied in preprocessing, and the feature extraction method was Radon transform. An accuracy of over 99% was obtained on a database of 25 subjects with 20 samples. In Wu and Liu's work [2], a finger-vein pattern identification using a SVM was proposed, in which the feature extraction method was principal component analysis (PCA) and linear discriminant analysis (LDA) algorithms. The database has 10 people with 10 samples for each finger, and classification using the SVM resulted in an accuracy of 98% and processing time was 0.015 s. In 2011, Wu and Liu [14] proposed a finger-vein identification using PCA as the feature extraction and also a neuro-fuzzy inference system (ANFIS) as the classifier. With 10 people and 10 samples for each finger, the accuracy obtained was 99% with 45 s of response time.

In the finger-vein recognition methods mentioned, much processing effort is required to remove the noise and extract and enhance the features before the recognition process can be performed. A significant advantage of the CNN over conventional approaches in pattern recognition is due to its ability to simultaneously extract features, reduce data dimensionality, and classify in one network structure. The CNN is robust to noise and small misalignments, and therefore the design of CNN-based recognizers requires minimal image preprocessing and the feature extraction and recognition tasks can be performed in a minimum number of network layers. This paper proposes a novel method based on a CNN to perform finger-vein biometric identification. To the best of our knowledge, this is the first attempt to apply a CNN for finger-vein recognition. The remainder of this paper is organized as follows: the second section describes the theoretical background of the proposed approach. The third section discusses in detail the proposed approach, followed by the experimental work and results. The final section concludes the paper.

2. Theoretical background

A well-known CNN architecture called LeNet-5 was proposed by LeCun et al. [11]. The LeNet-5 CNN, which was first applied in a handwriting recognition problem, consisted of seven layers performing convolution alternated with subsampling operations, as illustrated in Figure 1. The first convolutional layer, which convolves the input with a convolution kernel, essentially acts an edge detector that extracts salient features of the input samples. The kernel, which is of size 5×5 in this case, consists of weighting coefficients that create blurring (low-pass filter), sharpening (high-pass filter), or edge enhancement effects. The convolution process is performed by moving a flipped kernel through the images and the resulting output is placed as a new pixel of a feature map at the succeeding layer. The second layer performs subsampling, that is, a local averaging on a nonoverlapping small window size of 2×2 in this case. This operation reduces the resolution of the feature maps from the previous layer, essentially adding robustness against small distortions in translation, rotation, and scaling. The final two layers of this architecture are MLPs that act as classifiers.

The CNN architecture proposed in this paper is based on the work by Simard et al. [15], in which convolution and subsampling layers were fused into one layer. As shown in Figure 2, this fusion significantly reduces the total number of layers in the CNN from seven to four. Simard et al.'s CNN was also applied in

handwriting recognition, and a higher accuracy than that of LeNet-5 was achieved. The fused convolution and subsampling layer can be described by Eq. (1):

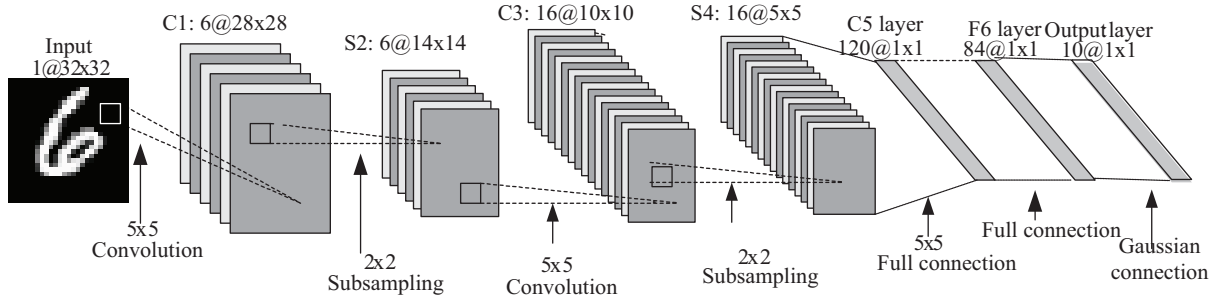


Figure 1. LeNet-5 CNN applied in handwriting recognition [11].

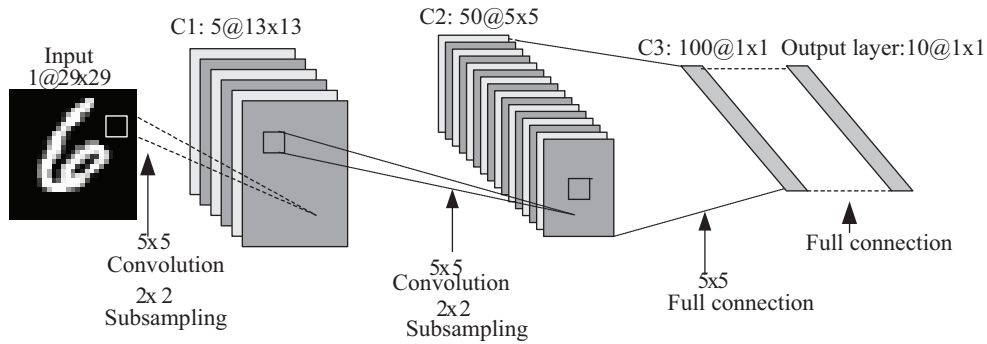


Figure 2. CNN with fused convolution/subsampling proposed by Simard et al. [15].

$$y_{pj}^{(l)} = f \left(\sum_{i \in M_j^{(l-1)}} \sum_{(u,v) \in K^{(l)}} w_{ji}^{(l)}(u,v) \circ x_{pi}^{(l-1)}(c+u, r+v) + b_j^{(l)} \right) \quad (1)$$

$$c \bmod S_x = 0, r \bmod S_y = 0,$$

where $K = \{(u, v) \in N^2 | 0 \leq u < k_x; 0 \leq v < k_y\}$, k_x and k_y are the width and the height of the convolution kernels $w_{ji}^{(l)}$ of layer (l) and $b_j^{(l)}$ is the bias of feature map n in layer (l) , c and r respectively refer to the column and row of a current pixel, and p refers to the particular training sample. The set $M_j^{(l-1)}$ contains the feature maps in the preceding layer $(l-1)$ that are connected to feature map n in layer (l) . The notation f is the activation function of layer (l) . The variables u and v describe the horizontal and vertical step sizes of the convolution kernel in layer (l) .

3. The proposed approach

The development of the proposed finger-vein biometric identification system is discussed in two parts: the preprocessing stage and CNN design. The image acquisition part of the design is not discussed as it is out of

the scope of this paper. For development and testing, the finger-vein image database used in this work was developed in our own laboratory and has been reported in [16]. We describe this image database in more detail in Section 4.1. Typical image processing designs require image preprocessing and segmentation to be performed before any further image processing and feature extraction process can be conducted. Preprocessing involves algorithms that include, among others, region of interest (ROI) extraction, image enhancement (denoising), and image normalization (resizing). Our approach does not require image enhancement since the CNN is robust to noise. The segmentation process essentially transforms the original image into a meaningful representation that is easier to analyze.

3.1. Preprocessing stage

In the proposed system, image capture is done by placing a finger between an infrared light source and a camera. A near infrared light (NIR) with wavelength of 760–850 nm is transmitted from the back of the hand, penetrating deeply into the skin while the radiation of light is absorbed by the deoxyhemoglobin [17]. When hemoglobin absorbs light, the finger veins appear as a pattern of shadows. These vein patterns are enhanced in the preprocessing stage of the design.

The original image captured is of size 240×320 pixels. We crop the image into 70×130 size, setting the origin at the center pixel of the finger image (the center pixel is fixed for all samples despite small finger misalignments among the samples). This size covers almost the entire finger. We perform this image cropping in order to reduce the time consumed by the process of local dynamic thresholding, which will be subsequently applied in the segmentation phase. The image is then resized to 55×67 pixels, reducing some information content so as to ease subsequent NN training. This image size has significant impact on the size of the convolution kernel to be selected, and this aspect is further discussed in the following section. The robustness of the proposed system is evaluated through two cases, namely the case in which preprocessing is performed with segmentation (using local dynamic thresholding; from here on referred to as *Case A*), and the case in which no segmentation is performed (referred to as *Case B*). Figure 3 illustrates the preprocessing applied in *Case A*, while *Case B* is shown in Figure 4.

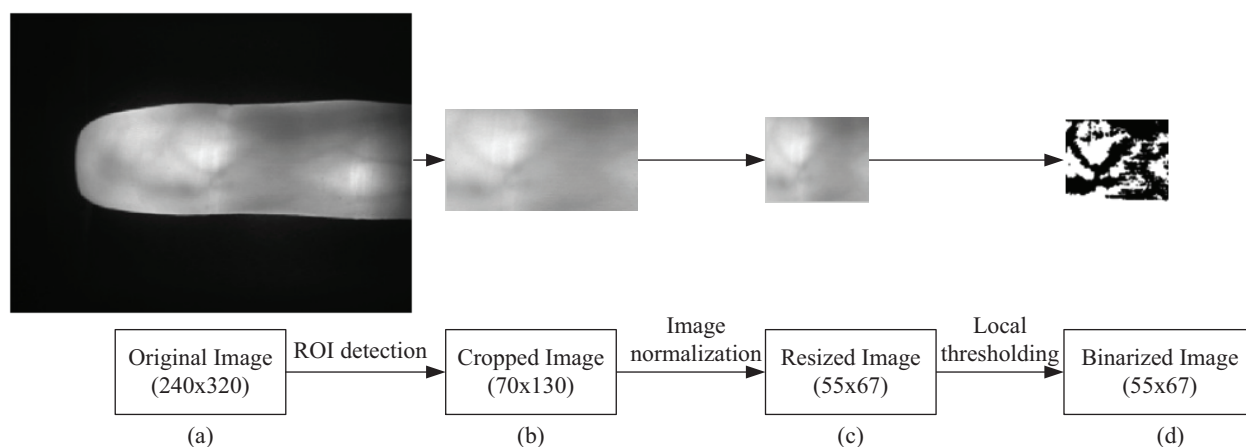


Figure 3. *Case A* - preprocessing that includes segmentation.

For the segmentation process (applied in *Case A*), we implement local dynamic thresholding, an algorithm that converts a gray-scale image into a bilevel representation of black and white pixels, resulting in the

binarization of the image. In this work, the black and white pixels are set to -1 and $+1$, respectively. Local dynamic thresholding allows adapting the threshold to the pixel within the window size. Let $g(x, y) \in [-1, 1]$ be the intensity of a pixel at location (x, y) in a gray-scale image. Threshold $t(x, y)$ is calculated by finding the mean pixel value within a window size shown by Eq. (2). For every pixel at location (x, y) , $t(x, y)$ is its local threshold and $m(x, y)$ is the mean of the pixel intensities in a $w \times w$ sized window centered on the pixel (x, y) as shown in Eq. (3).

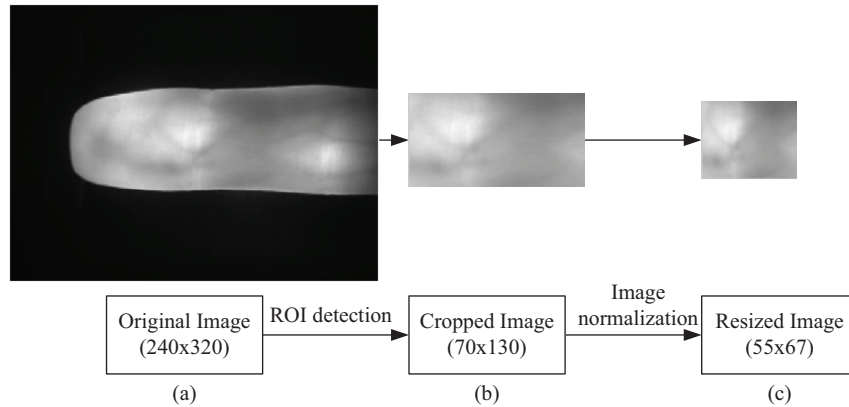


Figure 4. Case B - preprocessing that does not include segmentation.

$$t(x, y) = m(x, y) \tag{2}$$

$$out(x, y) \begin{cases} -1 & \text{if } g(x, y) < t(x, y) \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

After thresholding, each pixel at location (x, y) is replaced with $out(x, y)$. We selected a 19×19 window size for the local dynamic thresholding. This method of segmentation is simpler than *NiBlack* segmentation [18] that requires the mean and variance to be computed [19]. Thus, the preprocessing applied in this work has lower computation complexity than the traditional approach. Examples of preprocessed images for 5 subjects are shown in Figure 5, with *Case A* in Figure 5a and *Case B* in Figure 5b. In Figure 5b, the vein images are unclear due to different effects of illumination. These images are used to evaluate the robustness of the developed system.

3.2. Proposed CNN architecture

As mentioned earlier, the proposed CNN architecture is based on a design that consists of fused convolutional and subsampling layers as first proposed by Simard et al. in [15]. The method dispenses does away with complex operations, such as momentum, weight decay, structure-dependent learning rates, averaging layers, extra padding on the input, tangent prop, and even the fine tuning of the architecture, which are otherwise needed in conventional CNN solutions [15]. Figure 6 shows the proposed CNN architecture for finger-vein biometric recognition. The CNN consists of four layers (namely C1, C2, C3, and C4 convolutional layers), and this does not include the input layer. We refer to this architecture as the *5-13-50* model, implying that there are 5, 13, and 50 feature maps in layers C1, C2, and C3, respectively. Layer 4 (i.e. the output layer) is fixed

at 50 neurons since the target number of categories to classify is 50 subjects; hence, this information is implicit and therefore unnecessary to include in the model referencing name. The cross-validation technique, a popular empirical method for estimating the generalization accuracy of a NN [20,21], is applied to find the best model configuration, in which we determine the best number of feature maps and the best connection schemes to implement. We will discuss this best model selection process in the following section.

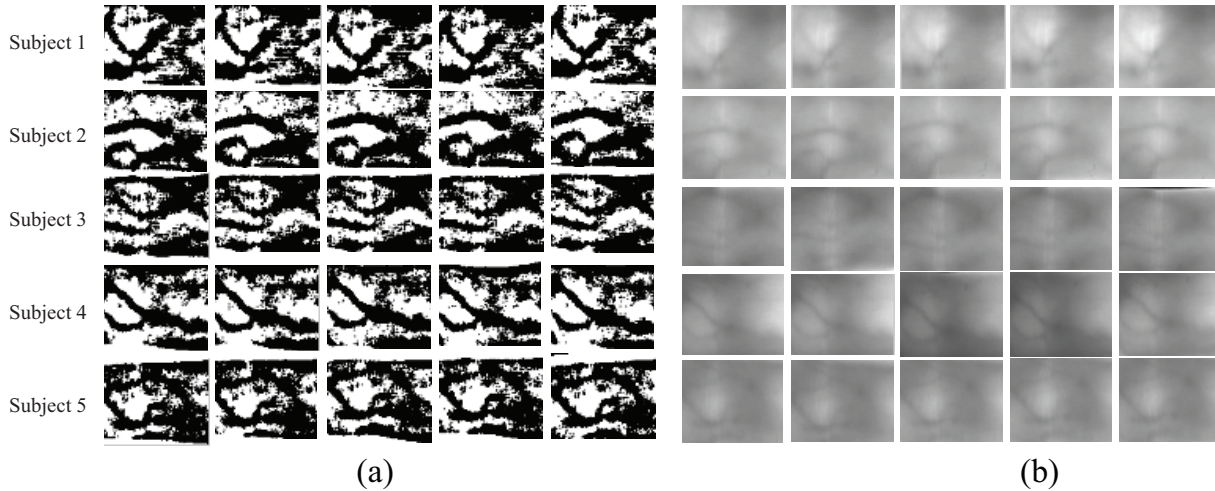


Figure 5. Image samples of 5 subjects in (a) Case A and (b) Case B.

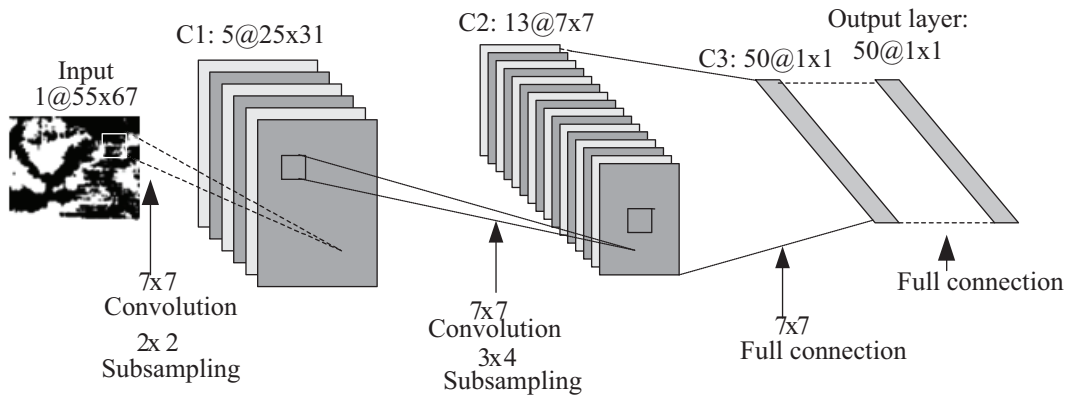


Figure 6. The proposed CNN architecture (referred to here as the 5-13-50 model).

In our CNN design, the input image size is set to 55×67 pixels and a 7×7 convolution kernel size is used. Each convolution operation uses the same kernel size. For subsampling operations, the kernel sizes are different for different layers: the kernel size in subsampling in layer C1 is 2×2 , and in C2 it is fixed at 3×4 . The selection of convolution and subsampling sizes determines the sizes of the feature maps at layer C3. We are targeting for 1×1 neurons at the final two layers of the MLP classifier. Each layer performs feature extraction tasks and dimensionality reduction processes at the same time. The size of feature maps at each layer follows the following equation:

$$M_x^{(l)} = \frac{M_x^{(l-1)} - (K_x^{(l)} - S_x^{(l)})}{S_x^{(l)}}; \quad M_y^{(l)} = \frac{M_y^{(l-1)} - (K_y^{(l)} - S_y^{(l)})}{S_y^{(l)}}, \quad (4)$$

where M_x and M_y give the sizes of the output map (M_x, M_y) in x and y directions, K_x and K_y indicate the convolution kernel size, S_x and S_y determine the subsampling kernel size, and (l) and $(l-1)$ refer to the current and previous layer.

In order to break symmetry and reduce the number of free parameters, we do not apply a full connection scheme for layer C1-C2. If full connection is applied between C1 and C2, there might be the possibility that the same resulting features are obtained in layer C2. Based on the work in [11], we connect certain feature maps from layer C1 to generate new feature maps at layer C2. Table 2 lists five connection types that we examine through the cross-validation technique to find the best connection scheme (cross-validation is a popular empirical method to estimate the best model and parameters of a NN). In cross-validation, the training set is split into two disjointed sets: one set that contains samples to train with, and another set that is used for validation. Through 10-fold cross-validation, each layer of our 5-13-50 model is tested with these connection schemes. The best model is determined by selecting the one that results in the lowest validation error. The final connection scheme applied in our CNN design is given in Table 3.

Table 2. Types of connections between C1 and C2 layers.

Connection type	conn(0)	conn(1)	conn(2)	conn(3)	conn(4)	conn(5)
1	1	0	0	1	1	0
2	1	1	0	0	1	1
3	1	1	1	1	1	1
4	0	1	1	0	1	1
5	0	0	1	1	0	1

Table 3. Proposed connection scheme between C1 and C2 layers.

C1 \ C2	C2												
	0	1	2	3	4	5	6	7	8	9	10	11	12
0	x	x	x		x		x						x
1	x	x	x		x		x	x	x				x
2	x	x	x	x	x	x	x	x	x	x	x	x	x
3	x	x			x		x		x	x	x	x	x
4					x		x	x		x	x	x	x

As recommended by LeCun et al. in [22], we use the hyperbolic tangent as the activation function, as given in Eq. (5). The notation x is the net input of a neuron. The activation function scales the net input into a specified range. The active range is $[1,-1]$ while the constant value is the asymptote of a sigmoid.

$$f(x) = 1.7159 * \tanh\left(\frac{2}{3}x\right) \tag{5}$$

In this CNN design, classification is based on the winner-takes-all rule [23]. The neuron with the value of +1 corresponds to the “winning” neuron (that corresponds to the correct class) while -1 corresponds to the incorrectly classified neurons. In the winner-takes-all approach, each output neuron represents a single predefined class of the pattern set. The output neurons compete among each other, in which only the neuron with maximum output value is determined as the winner. This indicates the class of the particular input pattern. This approach is simpler and computationally effective, since only the operation of finding the maximum output value among these output neurons is required in order to determine the class of the pattern. This approach replaces the similarity metric recognition approach that is commonly used in biometric matching processes. A true match is assigned during training and the optimal weights obtained enable direct identification of an

unknown subject in the test phase. This approach can speed up the identification process. Mean square error (E) is used as the error function given by:

$$E(k) = \frac{1}{P \times M} \sum_{j=1}^P \sum_{i=1}^M \left(y_i^j - t_i^j \right)^2, k = 1, \dots, s_k. \quad (6)$$

4. Experimental work and results

4.1. Finger-vein image database

This paper presents a new method for personal identification of finger-vein patterns using a CNN. The database used in this research was developed by the VeCAD Laboratory of Universiti Teknologi Malaysia and it consists of 50 subjects with 10 samples each for 6 different fingers, developed by [16]. In the mentioned database, the ages of participants were between 18 and 60 years and they were professors and university students. The infrared image for each subject was captured with different poses. The acquired image was also captured with adaptive illumination setting; hence, quality assessment is applied utilizing 2-D entropy. Therefore, there are a total of 500 samples.

In the NN training, the samples are separated into training and test samples that contain 400 and 100 samples, respectively. For faster convergence performance, an enhanced stochastic diagonal Levenberg–Marquardt learning algorithm is applied to train the NN. Detailed information on this learning algorithm is out of the scope of this paper. Mean square error is applied as the cost function. The system ran on a 2.5 GHz Intel i5-3210 M quad core processor, 8 GB RAM, with Ubuntu Linux OS. The algorithms are implemented in C, and MATLAB 9.0 is utilized for preprocessing.

4.2. Finding the best model: cross-validation results

This section presents the results obtained in the model selection process using the 10-fold cross-validation technique on six different CNN models that include *5-4-60*, *4-14-60*, *5-14-50*, *4-14-50*, *5-13-50*, and *5-14-40* models. Preprocessing with segmentation (*Case A*) is applied. In order to find the best model complexity, the 400 training images are divided into 10 folds, of which one is selected to be the validation dataset, and the remaining 9 folds are combined to form the training dataset. This method is repeated ten times, each time selecting a different fold as a validation set and the remaining nine to form the training set. In the cross-validation process, the error rate is computed for each validation set after training on its complement set, and it takes the average of these errors as the estimate of the out-of-sample error. Figure 7 gives the result of the 10-fold cross-validation process on the six models.

In this experiment, the Gaussian weight initialization is applied with a zero mean and a standard deviation of 0.03. Parameters μ and γ are set at 0.04 and 0.001, respectively. From Figure 7, we observe that the lowest validation error is at 15 epochs. Hence, we evaluate the validation error at the 15th epoch point, for assurance. From the figure, we can also see that the lowest number of feature maps in C3 is 50 (through the *5-14-50* model). When 40 is chosen, the validation error increases. In layer C2, the best number of feature maps is 14 and it becomes degraded when 13 feature maps are applied. A similar result occurs in layer C1 when the validation error is higher when the number of feature maps is reduced from 5 to 4.

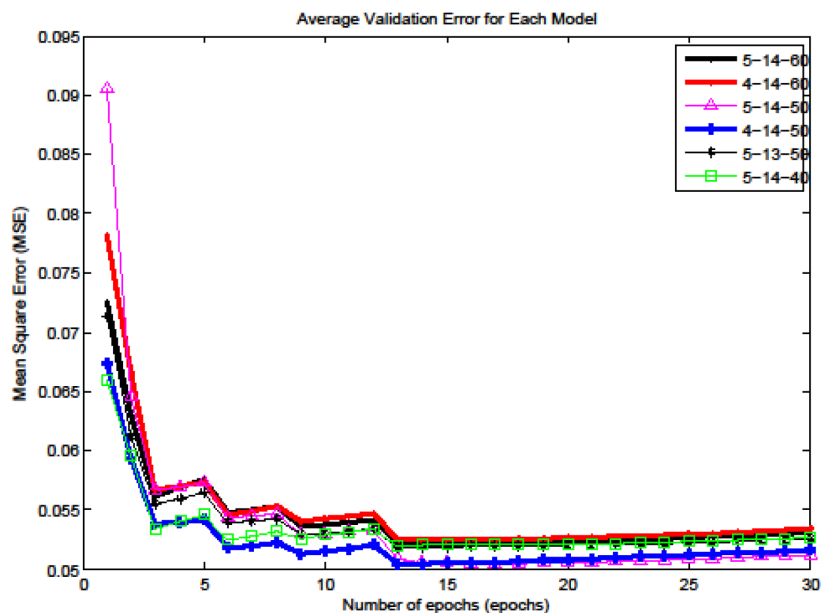


Figure 7. Error rates of the CNN models tested in cross-validation process.

The cross-validation result is only an estimation of a NN performance. The actual performance of a NN has to be evaluated on the test samples (100 samples). The performance measure of the system is based on the percentage of correct classification of the test samples. We measure the performance of the network by the accuracy of the classification process. In this work, instead of measuring the output error, we train all the samples and identify the number of misclassified samples. Misclassified samples (or samples being misrecognized) occur due to insufficient learning of the features contained in that particular sample by the NN. Correct classification is obtained by comparing the maximum value of the output vector and its corresponding class (target). If the position of the maximum value in the output vector is not equal to the target value, the number of misclassified samples is increased by 1. Then the accuracy is obtained by finding the percentage of correctly classified samples as given in the following equation:

$$\text{Accuracy} = \left(1 - \frac{\text{No. of misclassified test samples}}{\text{Total no. of test samples}} \right) \times 100\%. \tag{7}$$

Table 4 summarizes the results of the accuracy performance. The best accuracy of 100% is achieved by the 5-13-50 model.

Table 4. Performance of each model.

Model	Accuracy (%)
5-14-60	99.00
5-14-50	99.00
4-14-60	99.00
5-13-50	100.00
4-14-50	99.00
5-14-40	99.00

4.3. Optimum input size and convolution kernel size

The accuracy is also determined by the size of the input image and the convolution kernel size. The size of the convolution kernel is fixed for each layer. For fair comparison, we fixed the size of the subsampling kernel. The size of the input image is set to 55×67 with 7×7 convolution kernel size, 46×56 with 6×6 convolution kernel size, and 37×45 with 5×5 convolution kernel size. From Table 5, we can see that all of the mentioned settings produced very high accuracy with the best one shown by 55×67 input image and 7×7 convolution kernel size. Therefore, this setting is used for the next experiment in evaluating the best combination of normalization and weight initialization method. Note that the size of feature maps at each layer is calculated based on Eq. (4).

Table 5. Performance of the 5-13-50 CNN model for different input sizes and convolution kernel sizes.

Input	C1	C2	C3	Output layer	Accuracy (%)
55×67	5 @ 25×31 conv: 7×7 sub: 2×2	13 @ 7×7 conv: 7×7 sub: 3×4	50 @ 1×1 conv: 7×7	50 @ 1×1	100.00
46×56	5 @ 21×26 conv: 6×6 sub: 2×2	13 @ 6×6 conv: 6×6 sub: 3×4	50 @ 1×1 conv: 6×6	50 @ 1×1	99.00
37×45	5 @ 17×21 conv: 5×5 sub: 2×2	13 @ 5×5 conv: 5×5 sub: 3×4	50 @ 1×1 conv: 5×5	50 @ 1×1	99.00

4.4. The most suitable normalization and weight initialization methods

An experiment is conducted to evaluate the impact of the different combinations of normalization and weight initialization methods on the accuracy of the proposed CNN considering *Case A* and *Case B*. The normalization methods considered here are min-max and Z-score, and the weight initialization algorithms include uniform, Gaussian, fan-in, and Nguyen–Widrow. For min-max normalization method, let x_i and x_i' represent the current pixel value and new value, respectively. Then the normalization, x_i' can be computed by the following equation:

$$x_i' = \left(\max_t - \min_t \right) * \left[\frac{x_i - \min_v}{\max_v - \min_v} \right] + \min_t, \quad (8)$$

where \max_t and \min_t refer to the minimum and maximum range of the target, and \min_v and \max_v indicate the minimum and maximum pixel value of an image. The Z-score normalization, x_i' , is given by the following equation:

$$x_i' = \left[\frac{(x_i - \mu_i)}{\sigma_i} \right] \quad (9)$$

where μ_i and σ_i are the mean and standard deviation of the current image. This method produces data with zero mean and a unit variance.

Uniform and fan-in weight initialization differ in the range of randomization and weight distribution but have similar equations. The range of uniform weights is set to $[0.05 \text{ to } -0.05]$ while the fan-in initialization uses the range $[4.0/\text{fan-in} \text{ to } -4.0/\text{fan-in}]$. Fan-in is the number of incoming weights into a particular neuron. The Gaussian and Nguyen–Widrow methods have similar kinds of distribution. Gaussian distribution is determined

by the mean and standard deviation, whereas Nguyen–Widrow has a uniform distribution that is scaled by the number of hidden and input neurons, and a constant. The regularization parameter and γ constant are set at 0.04 and 0.001, respectively.

Training is performed with our selected model (i.e. model 5-13-50), applying the different combinations of normalization and weight initialization methods. Different parameter setting is required for each case. These parameters are obtained by executing the simulation three times and selecting the parameters that provide the best average accuracy (refer to Table 6). Table 7 presents the results of the experiment, which show that the highest accuracy (i.e. 100%) achieved in the case when preprocessing is done with segmentation (i.e. *Case A*) is when min-max normalization and Gaussian weight initialization methods are applied. With the Z-score normalization method, the best result possible is 98%. For the case when preprocessing is applied without segmentation (i.e. *Case B*), min-max and Z-score normalization has been applied, and the result shows that Z-score normalization provides the best result (100.00%). With the min-max normalization method, the best result that can be achieved is only 96.00%.

Table 6. Parameter setting and accuracy for the 5-13-50 model.

Approach	γ parameter	Regularization parameter	Gaussian standard deviation
<i>Case A</i>	0.001	0.04	0.03
<i>Case B</i> (with min-max normalization)	0.001	0.07	0.03
<i>Case B</i> (with Z-score normalization)	0.001	0.07	0.02

Note: *Case A* applies segmentation, while *Case B* does not.

Table 7. Classification accuracy of proposed 5-13-50 model for different combinations of weight initialization and normalization algorithms.

Normalization method	Weight initialization method	Accuracy (%), <i>Case A</i>	Accuracy (%), <i>Case B</i>
Min-max	Uniform	99.00	97.00
	Gaussian	100.00	96.00
	Fan-in	99.00	95.00
	Nguyen–Widrow	98.00	98.00
Z-score	Uniform	98.00	100.00
	Gaussian	98.00	100.00
	Fan-in	98.00	100.00
	Nguyen–Widrow	98.00	100.00

Results of training and test error, as well as rate of accuracy, are compared for both cases. Figures 8 and 9 depict training and test error and rate of accuracy during training for the best results obtained by *Case A* and *Case B*. The training and test error for *Case B* is much higher than that for *Case A*. In terms of accuracy, *Case B* reached optimum accuracy much slower than *Case A*, as shown in Figure 10. However, *Case B* excludes the costly local dynamic thresholding process. Reduced cost is preferred in any design; hence, *Case B* is chosen as the best approach. Figure 11 depicts the output images at each layer after training for *Case B*.

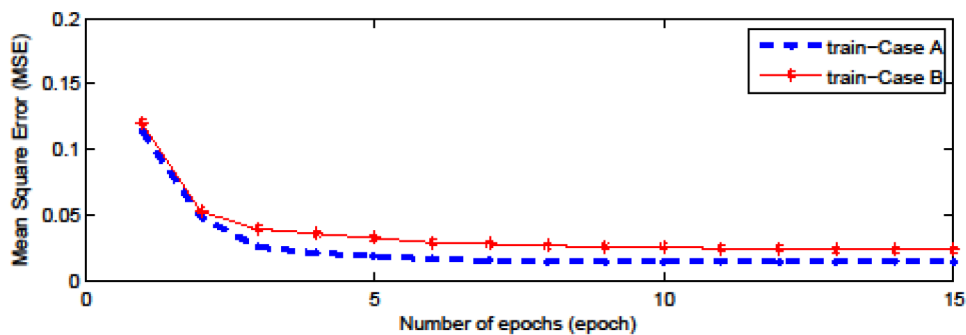


Figure 8. Error rates in training mode.

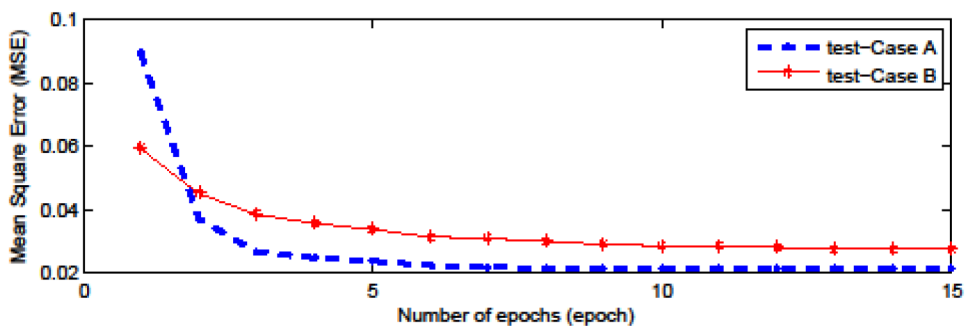


Figure 9. Error rates in testing mode.

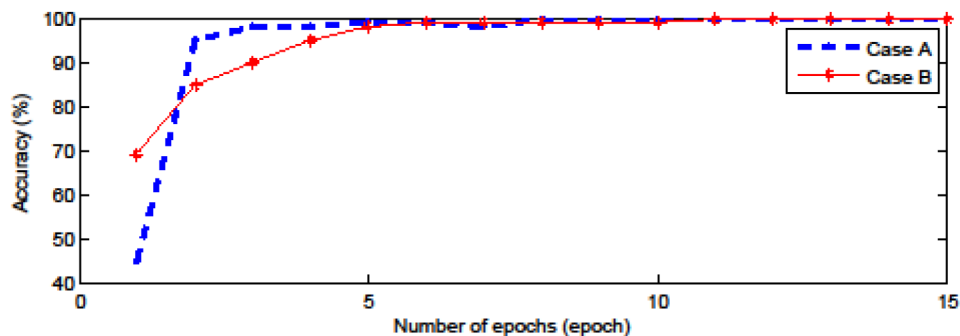


Figure 10. Classification accuracy in testing mode.

Additional testing has also been applied with an increased number of subjects. The database has been added with another 31 subjects, resulting in a total of 81 subjects altogether. Table 8 shows the results obtained with 81 subjects compared to the previous result with 50 subjects. From the table, the accuracy obtained with 81 subjects has shown that 1 test sample is misclassified. Therefore, it can be concluded that, with an increasing number of subjects, the system could still perform effectively.

4.5. Benchmarking results

The proposed CNN-based finger-vein recognition solution is benchmarked against existing NN-based methods for recognition accuracy. Table 9 provides the benchmarking results, and it shows that the proposed method outperforms existing NN solutions in recognition accuracy, except for that of Zhang et al. [13]. That work was tested on 400 subjects, which is much larger than the database used in the proposed method. However, since

this is the first successful attempt of implementing a CNN-based solution for finger-vein problem, it can be concluded that the proposed method is viable.

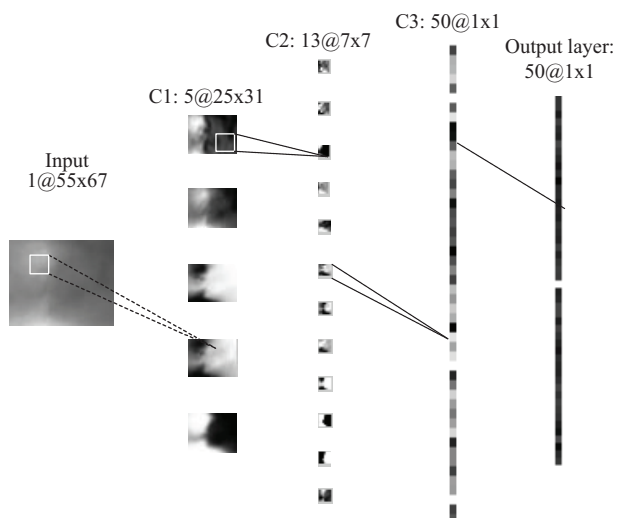


Figure 11. Output images at each layer after training for *Case B*.

Table 8. The accuracy achieved with 50 and 81 subjects, respectively.

Number of subjects	Number of training samples	Number of test samples	Accuracy (%)
50	400	100	100.00
81	648	162	99.38

Table 9. Benchmarking against existing NN-based finger-vein recognition.

Reference	Preprocessing algorithms applied	Feature extraction	No. of subjects	Accuracy (%)
Zhang et al., 2006 [13]	Curvelet decompose, orientation normalization	Multiscale based on curvelet	400	99.87
Wu and Ye, 2009 [1]	Vein region segmentation, gamma enhancement, gray-scale, and contrast enhancement	Radon transform	25	99.00
Wu and Liu, 2011 [2]	ROI extraction and image resize	PCA & LDA	10	98.00
Wu and Liu, 2011 [14]	ROI extraction and image resize	PCA	10	99.00
Proposed method	ROI extraction and image resize	CNN	50	100.00
Proposed method	ROI extraction and image resize	CNN	81	99.38

5. Conclusion

A new method of a finger-vein biometric identification system was developed using a CNN. A four-layered CNN with fusion of convolution and subsampling layers was proposed through the 5-13-50 model. An enhanced stochastic diagonal Levenberg–Marquardt algorithm was applied to ensure faster convergence. This work applies the winner-takes-all rule as the recognition method. This method replaces the similarity metric matching method that is normally applied in common biometric approaches. The advantage of this method is that a true match

is assigned to each subject during the training and the identity of an unknown subject is directly known in the test phase. On a 2.5 GHz Intel i5-3210M quad core processor, 8 GB RAM computer, the recognition time is less than 0.1574 s (including the preprocessing stage). The response time sufficiently satisfied the requirements for user convenience.

Experimental work has shown that optimum accuracy is achieved with the proposed CNN-based solution that implements preprocessing without the costly segmentation (local dynamic thresholding) process. The combination of Z-score and uniform weight was identified as the most appropriate normalization and weight initialization method. The input image of 55×67 was selected as the most optimum size. These selections led to a 100.00% and 99.38% recognition rate tested on samples from 50 and 81 subjects, respectively. From the results obtained, we can also conclude that segmentation processes and noise filtering are not required for the proposed method. For future work, the proposed CNN model will be tested with a more challenging database that contains a large number of subjects.

Acknowledgment

Authors would like to thank Universiti Teknikal Malaysia Melaka (UTeM) and Ministry of Higher Education for supporting this research under FRGS/1/2015/TK04/FKEKK/02/F00266.

References

- [1] Wu JD, Ye SH. Driver identification using finger-vein patterns with Radon transform and neural network. *Expert Syst Appl* 2009; 36: 5793-5799.
- [2] Wu JD, Liu CT. Finger-vein pattern identification using SVM and neural network technique. *Expert Syst Appl* 2011; 38: 14284-14289.
- [3] Yanagawa T, Aoki S, Ohyama T. Human finger vein images are diverse and its patterns are useful for personal identification. In: 21st Century COE Program, Development of Dynamic Mathematics with High Functionality; 2007. pp. 1-8.
- [4] Mulyono D, Jinn HS. A study of finger vein biometric for personal identification. In: International Symposium on Biometrics and Security Technologies; 2008.
- [5] Miura N, Nagasaka A, Miyatake T. Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification. *Mach Vision Appl* 2004; 15: 194-203.
- [6] Lee EC, Park KR. Image restoration of skin scattering and optical blurring for finger vein recognition. *Opt Lasers Eng* 2011; 49: 816-828.
- [7] Podgantwar UD, Raut U. Extraction of finger-vein patterns using Gabor filter in finger vein image profiles. *International Journal of Engineering Research and Technology* 2013; 2: 60861.
- [8] Yang J, Shi Y. Finger-vein ROI localization and vein ridge enhancement. *Pattern Recogn Lett* 2012; 33: 1569-1579.
- [9] Liu Z, Yin Y, Wang H, Song S, Li Q. Finger vein recognition with manifold learning. *J Netw Comput Appl* 2010; 33: 275-282.
- [10] Cheung B. Convolutional neural networks applied to human face classification. In: 11th International Conference on Machine Learning and Applications; 2012. pp. 580-583.
- [11] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. *Proc IEEE* 1998; 86: 2278-2324.
- [12] Chen YN, Han CC, Wang CT, Jeng BS, Fan KC. The application of a convolution neural network on face and license plate detection. In: 18th International Conference on Pattern Recognition; 2006.

- [13] Zhang Z, Ma S, Han X. Multiscale feature extraction of finger-vein patterns based on curvelets and local interconnection structure neural network. In: 18th International Conference on Pattern Recognition; 2006. pp. 145-148.
- [14] Wu JD, Liu CT. Finger-vein pattern identification using principal component analysis and the neural network technique. *Expert Syst Appl* 2011; 38: 5423-5427.
- [15] Simard PY, Steinkraus D, Platt JC. Best practices for convolutional neural networks applied to visual document analysis. In: Seventh International Conference on Document Analysis and Recognition; 2003. pp. 958-963.
- [16] Lee Y, Khalil-Hani M, Bakhteri R. FPGA-based finger vein biometric system with adaptive illumination for better image acquisition. In: IEEE Symposium on Computer Applications and Industrial Electronics; 2012. pp. 107-112.
- [17] Zharov VP, Ferguson S, Eidt JF, Howard PC, Fink LM, Waner M. Infrared imaging of subcutaneous veins. *Laser Surg Med* 2004; 34: 2004.
- [18] Niblack W. *An Introduction to Digital Image Processing*. Birkerød, Denmark: Strandberg Publishing Company, 1985.
- [19] Khalil-Hani M, Eng P. Personal verification using finger vein biometrics in FPGA-based system-on-chip. In: 7th International Conference on Electrical and Electronics Engineering; 2011. pp. II-171-II-176.
- [20] Refaeilzadeh P, Tang L, Liu H. Cross-validation. In: Liu L, Özsü M, editors. *Tamer Encyclopedia of Database Systems*. Amsterdam, the Netherlands: Springer, 2009. pp. 532-538.
- [21] Haykin S. *Neural Networks: A Comprehensive Foundation*. 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall, 1999.
- [22] LeCun Y, Bottou L, Orr B, Muller K. Efficient BackProp. In: Haykin SS. *Neural Networks: Tricks of the Trade*. Amsterdam, the Netherlands: Springer, 1998.
- [23] Kohonen T. *The self-organizing map*. *Proc IEEE* 1990; 78: 1464-1480.