Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

A Variable Node-to-Node-Link Neural Network and Its Application to Hand-Written Recognition

S.H. Ling
Centre of Multimedia Signal Processing,
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

F.H.F. Leung
Centre of Multimedia Signal Processing,
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

H.K. Lam
Division of Engineering,
King's College London,
Strand, London,
United Kingdom

Abstract - This paper presents a variable node-to-node-link neural network (VN^{2}NN) trained by real-coded genetic algorithm (RCGA). The VN^{2}NN exhibits a node-to-node relationship in the hidden layer, and the network parameters are variable. These characteristics make the network adapt to the changes of the input environment, enable it to tackle different input sets distributed in a large domain. Each input data set is effectively handled by a corresponding set of network parameters. The set of parameters are governed by the other nodes. Taking the advantage of these features, the proposed network ensures better learning and generalization abilities. Application of the proposed network to hand-written graffiti recognition will be presented so as to illustrate the improvement.

I. Introduction

Neural network can approximate any smooth and continuous nonlinear functions in a compact domain to an arbitrary accuracy [2]. Three-layer feed-forward neural networks have been employed in a wide range of applications such as system modeling and control, forecasting [3]-[4], recognition [5]-[6], etc. Owing to its specific structure, a neural network can realize a learning process [2]. Learning of the network usually consists of two steps: designing the network structure and defining the learning process. The structure of a neural network affects the non-linearity of its input-output relationship. The learning algorithm governs the rules to optimize the connection weights. A typical network structure has a fixed set of connection weights after the learning process. However, a fixed set of connection weights may not be suitable to learn the information behind the data that are distributed in a vast domain separately.

For neural networks, the learning process aids to find a set of optimal network parameters. Traditionally, two major classes of learning rules, namely the error correction and gradient rules, were proposed. The error correction rules [2], such as the α-LMS algorithm, perception learning rules and May's rule, adjust the network parameters to correct the network output errors corresponding to the present input patterns. Some of the error correction rules are only applicable to linear separable problems. The gradient rules [7], such as the MRI, MRII, MRIII rules and backpropagation techniques, adjust the network parameters based on the gradient information to reduce the mean square error over all input patterns. One major weakness of the gradient methods is that the derivative information of the error function is needed such that it has to be continuous and differentiable. Also, the learning process is easily trapped in a local optimum, especially when the problem is multimodal and the learning rules are network structure dependent.

Global search algorithms such as genetic algorithm [8]-[9] were proposed. Unlike the gradient rules, these search algorithms are less likely to be trapped in local optima and do not require a differentiable or even continuous error function. Thus, they are more suitable for searching in a large, complex, non-differentiable and multimodal domain [8].

In this paper, modifications are made to the neural networks such that the parameters of the activation functions in the hidden layer are changed according to the network inputs. To achieve this, node-to-node links are introduced in the hidden layer. The structure of the VN^{2}NN is shown in Fig. 1. In each hidden node, the input from the lower neighbour control the bias term of the activation function and the input from the upper neighbour influence the sharpness of the activation function. Conceptually, the introduction of the node-to-node links increases the degree of freedom of the network model. Also, the structure of the node-to-node link is determined by the training algorithm. The resulting neural network is found to have a better learning and generalization abilities. The enhancements are due to the fact that the parameters in the activation functions of the hidden nodes are allowed to change in order to cope with the changes of the network inputs in different operating sub-domains. As a result, the VN^{2}NN seems to have a dedicated neural network to handle the inputs of different operating sub-domains. This characteristic is good for solving problems with input data set distributed in a large spatial domain.

This paper is organized as follows. The VN^{2}NN will be presented in section II. In section III, the training of the parameters of the VN^{2}NN using RCGA [1] will be presented. The application to hand-written recognition will be discussed in section IV. A conclusion will be drawn in section V.

0-7803-9490-9/06/$20.00/©2006 IEEE

II. VARIABLE NODE-TO-NODE-LINK NEURAL NETWORK

A variable neural network with node-to-node relations in the hidden layer is shown in Fig. 1. An inter-node link with weight $\tilde{m}_i$ is connected from the $(i+d_m)$-th node to the *i*-th node. Similarly, an inter-node link with weight $\tilde{r}_i$ is connected from the *i*-th node to the $(i-d_r)$-th node, i = 1, 2, …, n_h.. d_m and d_r are the node-to-node distances, i.e. if d_m = 2, the link with weight $\tilde{m}_3$ will be connected from node 3 to node 1. Similarly, if d_r = 3, the link with weight $\tilde{r}_6$ will be connected from node 3 to node 6. As a result, the total number of node-to-node links is $2\times n_h$, where n_h is the total number of hidden nodes. The node-to-node relationship can enhance the degree of freedom of the neural network model if it is made adapting to the changes of the inputs. Consequently, the learning and the generalization abilities of the VN^{2}NN can be high.

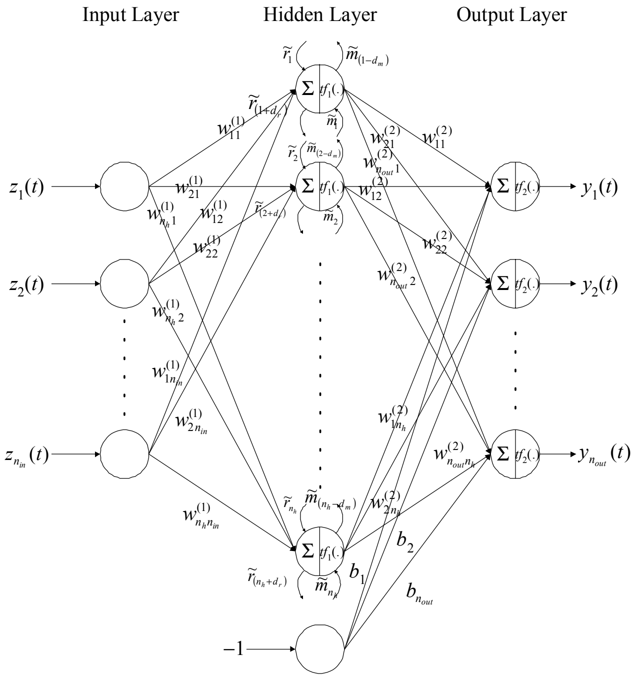


Fig. 1. Structure of the variable node-to-node-link neural network.

Conceptually, the proposed neural network can be regarded as consisting of two units, namely the rule-base (RB) and the data-processing (DP) neural networks as shown in Fig. 2. The RB neural network (with the node-to-node links) stores some rules governing how the DP neural network handles the input data. By using the VN^{2}NN, some problems that a traditional neural network with a limited number of parameters cannot provide a good performance may be solved. Fig. 3 shows an example to demonstrate the inadequacy of a traditional neural network to these problems. In this figure, S1 and S2 are two sets of data in a spatial domain. To solve a mapping problem, the weights of a traditional neural network can be trained to minimize the error between the network outputs and the desired values. However, the two data sets are separated too far apart for a single neural network to model. Then, the neural network may only model the data set S (average of S1 and S2). This problem might be alleviated if the neural network employs a larger number of parameters. To improve the learning and generalization abilities of the neural network, the structure shown in Fig. 2 is proposed. Referring to Fig. 3, when the input data belongs to S1, the RB neural network will provide the rule (network parameters corresponding to S1) for the DP neural network to handle the S1 data. Similarly, when the input data belongs to S2, the rules corresponding to S2 will be used by the DP neural network to handle the input data. In other words, it operates like two individual neural networks handling their corresponding input data. Consequently, the proposed neural network is suitable to handle a large number of data. In this paper, the proposed neural network will be applied to the problem of handwritten recognition, which involves lots of data.

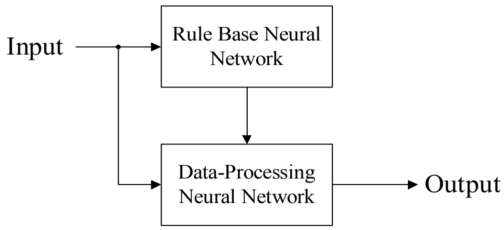


Fig. 2. Proposed architecture of the neural network.

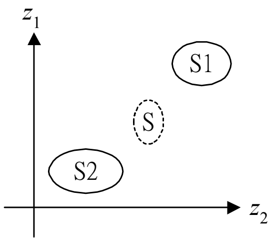


Fig. 3. Diagram showing two sets of data in a spatial domain.

Referring to Fig. 1, $z(t)=\left[z_1(t) \quad z_2(t) \quad \cdots \quad z_{n_{in}}(t)\right]$ denotes the input vector; n_{in} denotes the number of input nodes; *t* denotes the current input number which is a non-zero integer; $w_{ij}^{(1)}$, i = 1, 2, …, n_h, j = 1, 2, …, n_{in}, denotes the connection weights between the *j*-th node of the input layer and the *i*-th node of the hidden layer; n_h denotes the number of hidden nodes; $w_{ki}^{(2)}$, k = 1, 2, …, n_{out}, i = 1, 2, …, n_h, denotes the connection weights between the *i*-th node of the hidden layer and the *k*-th node of the output layer; n_{out} denotes the number of output nodes. $\tilde{m}_i$ and $\tilde{r}_i$ are the connection weights of the links between hidden nodes (there are $2n_h$ node-to-node links); d_m is the node-to-node distance between the i-th node and the ($i+d_m$)-th node, d_r is the node-to-node distance between the i-th node and the ($i-d_r$)-th node; b_k denotes the bias to the output nodes; $tf_1(\cdot)$ and $tf_2(\cdot)$ denote the activation functions of the hidden and output nodes respectively; $\mathbf{y}(t)=\left[y_1(t) \quad y_2(t) \quad \cdots \quad y_{n_{out}}(t)\right]$ denotes the output vector. The input-output relationship of the proposed neural network is governed by the following equation:

$$y_k(t)=tf_2\left(\sum_{i=1}^{n_h} w_{ki}^{(2)} f_{s_i}(\mathbf{z}(t))-b_k\right),\ k=1,\,2,\,\ldots,\,n_{out} \qquad (1)$$

Fig. 4 shows the proposed neuron at node i, the output of the neuron $f_{s_i}(\cdot)$ is defined as:

$$f_{s_i}(\mathbf{z}(t))=tf_1(\chi_i(t),\widetilde{m}_i(t),\widetilde{r}_i(t)),\ i=1,\,2,\,\ldots,\,n_h \qquad (2)$$

$$\chi_i(t)=\sum_{j=1}^{n_{in}} w_{ij}^{(1)} z_j(t) \qquad (3)$$

$$\widetilde{m}_i(t)=m_i\sum_{j=1}^{n_{in}} \widetilde{w}_{(i+d_m)j}^{(1)} z_j(t) \qquad (4)$$

$$\widetilde{r}_i(t)=r_i\sum_{j=1}^{n_{in}} \widetilde{w}_{(i-d_r)j}^{(1)} z_j(t) \qquad (5)$$

where m_i and r_i are parameters to be trained.

$$\widetilde{w}_{(i+d_m)j}^{(1)}=\begin{cases} w_{((i+d_m)-n_h)j}^{(1)} & \text{for } i+d_m>n_h \\ w_{(i+d_m)j}^{(1)} & \text{otherwise} \end{cases} \qquad (6)$$

$$\widetilde{w}_{(i-d_r)j}^{(1)}=\begin{cases} w_{(n_h+(i-d_r))j}^{(1)} & \text{for } i-d_r<1 \\ w_{(i-d_r)j}^{(1)} & \text{otherwise} \end{cases} \qquad (7)$$

$$tf_1(\chi_i(t),\widetilde{m}_i(t),\widetilde{r}_i(t))$$

$$=\frac{2}{1+e^{-\left(\frac{(\chi_i(t)-\widetilde{m}_i(t))}{2(\widetilde{r}_i(t))^2}\right)}}-1$$

$$=\frac{2}{1+e^{-\left(\frac{\left(\sum_{j=1}^{n_{in}} w_{ij}^{(1)} z_j(t)-m_i\sum_{j=1}^{n_{in}} \widetilde{w}_{(i+d_m)j}^{(1)} z_j(t)\right)}{2\left(r_i\sum_{j=1}^{n_{in}} \widetilde{w}_{(i-d_r)j}^{(1)} z_j(t)\right)^2}\right)}}-1\in\left[-1 \quad 1\right] \qquad (8)$$

$tf_2(\cdot)$ can be any commonly used activation functions such as the pure linear, hyperbolic tangent sigmoid, logarithmic sigmoid activation functions [2, 7]. As mentioned early, the node-to-node links enhances the degree of freedom of the modelled function. In each neuron in hidden layer, the input from the lower neighbour ($\widetilde{m}_i(t)$) concerns the bias term while the input from the upper neighbour ($\widetilde{r}_i(t)$) influences the sharpness of the edges of the hyper-planes in the search space. It can be seen from (8) that the proposed activation function $tf_1(\cdot)$ is characterized by the varying mean ($\widetilde{m}_i(t)$) and the varying standard deviation ($\widetilde{r}_i(t)$) respectively. Their values will be changed according to the changes of the inputs of the activation function, i.e. $\chi_i(t)$. The mean is used to control the bias as shown in Fig. 5 and the standard deviation is used to control the sharpness as shown in Fig. 6. Referring to Fig. 3, when the input data belongs to S1, the corresponding $\chi_i(t)$ will drive the other nodes (with $\widetilde{m}_{(i-d_m)}$ and $\widetilde{r}_{(i+d_r)}$) to handle the S1 data. Similarly, when the input data belongs to S2, the corresponding $\chi_i(t)$ will drive the other nodes to handle it accordingly. In other words, it operates like two individual neural networks handling the corresponding input data. Comparing with the conventional feed-forward neural network with fixed parameters after training, the VN^{2}NN can offer a better performance. In the VN^{2}NN, the optimal values of the parameters $w_{ij}^{(1)}$, $w_{ki}^{(2)}$, m_i, r_i, b_k, d_m and d_r are obtained by an improved RCGA [1].

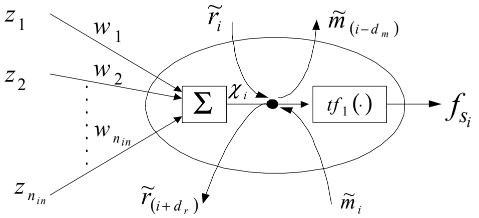


Fig. 4 Proposed neuron at node i.

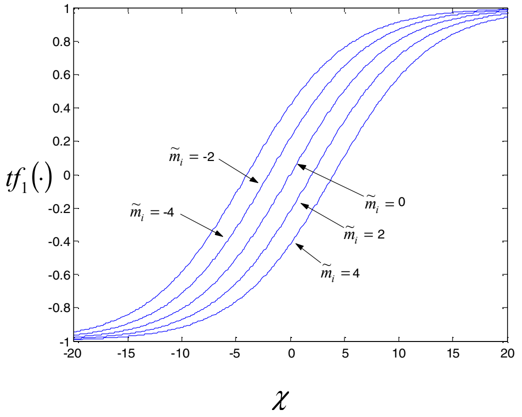


Fig. 5 Samples of the activation function $tf_1(\cdot)$ of the proposed neuron with different $\widetilde{m}_i$, ($\widetilde{r}_i=0$).

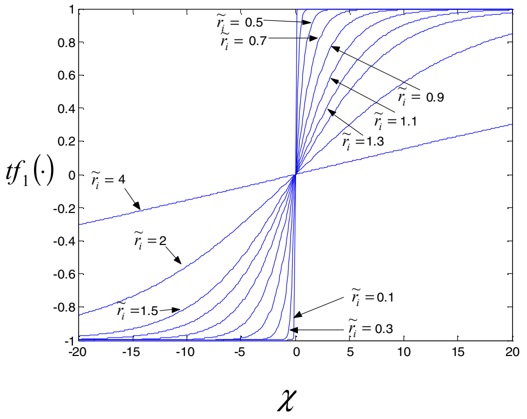


Fig. 6 Samples of the activation function $tf_1(\cdot)$ of the proposed neuron with different $\tilde{r}_i$, ($\tilde{m}_i = 0$).

III. Learning with Real-Coded Genetic Algorithm

RCGA is a powerful global search algorithm that has been widely applied in various optimization problems. One superior characteristic of RCGA is that the detailed information of the nonlinear system is not necessarily known. Hence, RCGA is suitable to handle some complex optimization problems. As the RCGA search process is mainly based on random search techniques, the global optimization ability and convergence may be in doubt within a finite search time. In this paper, RCGA is employed to optimize the fitness function which, is characterized by the network parameters of the VN^{2}NN. The fitness function is a mathematical expression quantitatively measures the performance of the RCGA tuning process. A larger fitness value indicates better tuning performance. By adjusting the values of the network parameters, the fitness function is maximized (cost value is minimized) based on the RCGA. During the tuning process, offspring with better fitness values evolves. The mutation operation will contract gradually with respect to the iteration number. After the tuning process, the obtained network parameter values will be used by the proposed neural network. As the proposed neural network is a feed-forward one, the outputs are bounded if its inputs are bounded which happens for most of the real-life applications. Consequently, no convergence problem is present for the neural network itself.

The input-output relationship of the proposed VN^{2}NN can be described by,

$$\mathbf{y}^d(t) = g(\mathbf{z}^d(t)),\ t = 1, 2, \ldots, n_d . \tag{9}$$

where $\mathbf{z}^d(t) = \begin{bmatrix} z_1^d(t) & z_2^d(t) & \cdots & z_{n_{in}}^d(t) \end{bmatrix}$ and $\mathbf{y}^d(t) = \begin{bmatrix} y_1^d(t) & y_2^d(t) & \cdots & y_{n_{out}}^d(t) \end{bmatrix}$ are the given inputs and the desired outputs of an unknown nonlinear function $g(\cdot)$ respectively; n_d denotes the number of input-output data pairs. The fitness function of the RCGA depends on the application. One common fitness function is defined as,

$$fitness = \frac{1}{1+err} \tag{10}$$

In (10), *err* can be the mean square error (MSE) defined as:

$$err = \frac{\sum_{t=1}^{n_d}\sum_{k=1}^{n_{out}}\left(y_k^d(t) - y_k(t)\right)^2}{n_d n_{out}} \tag{11}$$

The objective is to maximize the fitness value of (10) (minimize *err*) using RCGA by setting the chromosome to be $\begin{bmatrix} w_{ij}^{(1)} & w_{ki}^{(2)} & b_k & m_i & r_i & d_m & d_r \end{bmatrix}$ for all *i*, *j* and *k*. The range of the fitness value of (10) is (0, 1]. After training, the values of these network parameters will be fixed during the operation. The total number of tuned parameters (n_{para}) of the proposed VN^{2}NN is:

n_{para} = number of parameters between input and hidden layers + number of parameters between hidden and output layers + number of parameters for m_i, r_i, d_m, d_r

$$n_{para} = n_{in}n_h + n_{out}(n_h + 1) + (2n_h + 2) \tag{12}$$

$$= (n_{in} + n_{out} + 2)n_h + n_{out} + 2 \tag{13}$$

IV. Application Example and Results

A hand-written graffiti pattern recognition problem is given to illustrate the superior learning and generalization abilities of the proposed network on a classification problem with large number of input sets. In general, the neural/neural-fuzzy network approaches are model-free, and they learn the features of the training patterns in an off-line manner. The features can then be recognized using the trained neural/neural-fuzzy network. In [12], the orthogonality and information measures were employed to evaluate the features of some characters. These two measures will be taken as the inputs of the multi-layer feed-forward neural networks for classification. The self-organizing map approach can also be found in [13], where a self-organizing map model was used to implement a modular classification system. An improved neocognitron approach, which is good in dealing with two-dimensional pattern recognition problems, was proposed in [14] for digit classification. However, the computational demand of the approach is high because of the complex structure of the neocognitron. A combined neural network architecture [5], which consists of two neural networks connected in cascade, can also be found to handle the recognition problems. The first neural network is for feature extraction while the second one acts as a classifier. Different kinds of neural and statistical classifiers were also reported in [15].

A. Neural network based hand-written recognition system

In this example, the digits 0 to 9 and three (control) characters (*backspace*, *carriage return* and *space*) are recognized by the VN^{2}NN. These graffiti are shown in Fig. 7. A point of each graffiti is characterized by a number based on the *x*-*y* coordinates on a writing area. The size of the writing area is $x_{\max}$ by $y_{\max}$. The bottom left corner is set as (0, 0). Ten uniformly sampled points of the graffiti will be taken as the inputs of the recognizer. The points are taken in the following way. First, the input graffiti is divided into 9 uniformly distanced segments characterized by 10 points, including the start and the end points. Each point is labeled as (x_i, y_i), i = 1, 2, …, 10. The first 5 points, (x_i, y_i), i = 1, 3, 5, 7 and 9, taken alternatively are converted to 5 numbers ρ_i respectively by using the formula $\rho_i = x_i x_{\max} + y_i$. The other 5 points, ($x_i$, y_i), i = 2, 4, 6, 8 and 10, are converted to 5 numbers respectively by using the formula $\rho_i = y_i y_{\max} + x_i$. These ten numbers, z_i, i = 1, 2, … ,10, are used as the inputs of the proposed graffiti recognizer. The hand-written graffiti recognizer as shown in Fig. 8 is proposed. Its inputs are defined as follows,

$$\bar{\mathbf{z}}(t) = \frac{\mathbf{z}(t)}{\|\mathbf{z}(t)\|} \tag{14}$$

where $\bar{\mathbf{z}}(t) = [\bar{z}_1(t) \quad \bar{z}_2(t) \quad \cdots \quad \bar{z}_{10}(t)]$ denotes the normalized input vectors of the proposed graffiti recognizer; $\mathbf{z}(t) = [z_1(t) \quad z_2(t) \quad \cdots \quad z_{10}(t)]$ denotes the ten points in the writing area; $\|\cdot\|$ denotes the l_2 vector norm. The 16 outputs, $y_k(t)$, k = 1, 2, …, 16, indicates the similarity between the input pattern and the 16 standard patterns respectively. The input-output relationship of the training patterns is defined such that the output $y_i(t) = 1$ and others are zero when the input vector belongs to pattern i, i = 1, 2, …, 16. For example, the desired outputs of the pattern recognition system are [1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] for the digit "0(a)", [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0] for the digit "0(b)", and [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1] for the character "space". After training, a graffiti determiner is used to determine the output of the graffiti. A larger value of $y_j(t)$ implies that the input pattern matches more closely to the corresponding graffiti pattern. For instance, a large value of $y_0(t)$ implies that the recognized input pattern is "0".

B. Results and analysis

To train the neural network of the hand-written graffiti recognition system, a set of training patterns governing the input-output relationship of the network will be used. 1600 training patterns (100 patterns for each graffiti) are used in this example. The training patterns consist of the input vectors and its corresponding expected output. The fitness function is given by (10), with

$$err = \sum_{k=1}^{16} \frac{\sum_{t=1}^{100} \left(\frac{y_k(t)}{\|\mathbf{y}(t)\|} - \frac{y_k^d(t)}{\|\mathbf{y}^d(t)\|} \right)^2}{16 \times 100} \tag{15}$$

where $\mathbf{y}^d(t) = [y_1^d(t) \quad y_2^d(t) \quad \cdots \quad y_{16}^d(t)]$ denotes the expected output vector and $\mathbf{y}(t) = [y_1(t) \quad y_2(t) \quad \cdots \quad y_{16}(t)]$ is the actual network output defined as,

$$y_k(t) = tf_2\left(\sum_{i=1}^{n_h} w_{ki}^{(2)} f_{s_i}(\mathbf{z}(t)) - b_k \right), k = 1, 2, \ldots, 16. \tag{16}$$

where $f_{s_i}(\mathbf{z}(t)) = tf_1\left(\sum_{j=1}^{10} w_{ij}^{(1)} z_j(t), \tilde{m}_i(t), \tilde{r}_i(t) \right)$, $i = 1, 2, \ldots, n_h$

$$\tag{17}$$

where $tf_2(\cdot)$ is a pure linear transfer function in this application. In order to test the generalization ability of the proposed neural network, a set of testing patterns consisting of 480 input patterns (30 patterns for each graffiti) is used.

For comparison purpose, a conventional 3-layer fully connected feed-forward neural network (FFCNN) [7] and a wavelet neural network (WNN) [11] trained by the RCGA [1] are also used in this example. WNN is a kind of feed-forward neural network in which a multiscale wavelet is used as the transfer function in the hidden layer. It is good at handling recognition problems [16]. For all cases, the initial values of the parameters of the neural network are randomly generated. In this application, the lower and upper bounds of the network parameters of the VN^{2}NN are $[w_{ij}^{(1)} \quad w_{ki}^{(2)} \quad b_k \quad m_i] \in [-4 \quad 4]$, $r_i \in [0.5 \quad 2]$ and $[d_m \quad d_r] \in [1 \quad (n_h - 1)]$. For the WNN and FFCNN, the network parameters are ranged from –4 to 4. The number of iteration to train the neural networks is 15000. For RCGA [1], the probability of crossover (μ_c) and the probability of mutation (μ_m) are 0.8 and 0.05 respectively; the weights of crossover w_a and w_b are set at 0.5 and 1 respectively. The shape parameter of mutation ζ is 2. The population size is 50. All the results are averaged ones out of 20 runs.

The average training, best training, average testing and best testing results in terms of mean square error (MSE), and the recognition accuracy rate of all approaches are summarized in Table I. It can be seen from this table that the recognition system implemented by the VN^{2}NN outperforms that by the WNN and FFCNN. The best results are achieved when the number of hidden node (n_h) is set at 20 for VN^{2}NN and n_h = 24 for WNN and FFCNN. The number of parameter (n_{para}) of VN^{2}NN is 576, which is less than WNN (n_{para} = 648) and FFCNN (n_{para} = 664). Comparing with WNN and FFCNN, the average training and testing errors of VN^{2}NN are 0.0157 and 0.0186 respectively. They imply 96.82% and 76.90% improvement over WNN, and 129.3% and 101.1% improvement over FFCNN, respectively. In terms of the average testing recognition accuracy rate, the VN^{2}NN (96.96%) gives a better result than the WNN (93.92%) and FFCNN (91.25%).

Fig. 9 shows some selected outputs of the VN^{2}NN, WNN and FFCNN for the 480 (30 for each digit/command) testing graffiti. In this figure, the x-axis represents the pattern number for the corresponding digit/command. Pattern numbers 1 to 30 are for the digit '0'(a), numbers 31-60 are for the digit '0'(b), and so on. The y-axis represents the output y_i. As mentioned before, the input-output relationship of the patterns will drive the output $y_i(t) = 1$ and others to zero when the input vector belongs to pattern i, i = 1, 2, …, 16. For instance, the desired outputs **y** of the pattern recognition system are [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0] for digit "0(b). Referring to Fig.9b, we can see that the output y_2 for pattern numbers 31-60 by the VN^{2}NN are most near to 1, and the other are most near to zero. It shows that the accuracy rate achieved by the VN^{2}NN is good.

Table II shows the overall recognition rate (training and testing) for the digit (0-9) and the commands implemented by the VN^{2}NN, WNN, and FFCNN. Referring to this table, it can be seen that the digit and command recognition done by the proposed network is better. Overall, more than 4% improvement in the recognition accuracy rate is recorded. The digits '1', '6', '7', '9' and the command 'space' are done particularly well by the VN^{2}NN.

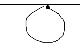
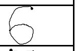
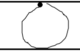
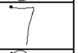
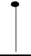
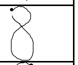
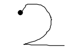
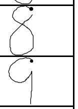
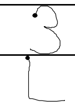
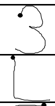
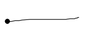
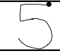
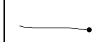
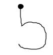
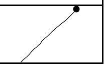
Digits or Characters	Strokes	Digits or Characters	Strokes
0(a)		6	
0(b)		7	
1		8(a)	
2		8(b)	
3		9	
4		Backspace	
5(a)		Carriage Return	
5(b)		Space	

Fig. 7. Graffiti digits and commands (with the dot indicating the starting point of the graffiti).

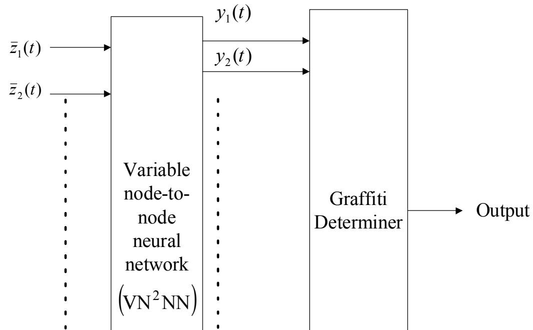


Fig. 8. Architecture of the hand-written graffiti recognizer.

V. Conclusion

A variable node-to-node-link neural network has been proposed in this paper. The parameters of the proposed neural network will be trained by RCGA. Thanks to the variable property and the node-to-node relations in the hidden layer, the learning and generalization abilities of the proposed network can be increased. An application on hand-written graffiti recognition has been given to illustrate that the proposed VN^{2}NN is good to handle a large set of data distributed in a large domain.

Acknowledgment

The work described in this paper was substantially supported by a grant from the Hong Kong Polytechnic University (PhD Student Account Code RG9T).

References

[1] S.H. Ling and F.H.F. Leung, "An improved genetic algorithm with average-bound crossover and wavelet mutation operations," *Soft Computing* (to be published).

[2] F.M. Ham and I. Kostanic, *Principles of Neurocomputing for Science & Engineering*. McGraw Hill, 2001.

[3] S.H. Ling, F.H.F Leung, H.K. Lam, and P.K.S. Tam, "Short-term electric load forecasting based on a neural fuzzy network," *IEEE Trans. Industrial Electronics*, vol. 50, no. 6, pp.1305-1316, Dec. 2003.

[4] F.H.F. Leung, H.K. Lam, S.H. Ling, and P.K.S. Tam, "Tuning of the structure and parameters of neural network using an improved genetic algorithm," *IEEE Trans. Neural Networks*, vol. 14, no. 1, pp. 79-88, Jan. 2003.

[5] C.A. Perez, C.A. Salinas, P.A. Estévez, and P.M. Valenzuela, "Genetic design of biologically inspired receptive fields for neural pattern recognition," *IEEE Trans. Systems, Man, and Cybernetics – Part B: Cybernetics*, vol. 33, no. 2, pp. 258-270, Apr. 2003.

[6] K.F. Leung, F.H.F. Leung, H.K. Lam, and S.H. Ling, "On interpretation of graffiti digits and commands for eBooks: neural-fuzzy network and genetic algorithm Approach," *IEEE Trans. Industrial Electronics,* vol. 51, no. 2, pp.464-471, Apr. 2004.

[7] B. Widrow and M.A. Lehr, "30 years of adaptive neural networks: Perceptron, madaline, and backpropagation," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1415-1442, Sept. 1990.

[8] Z. Michalewicz, *Genetic Algorithm + Data Structures = Evolution Programs*, 2nd extended ed. Springer-Verlag, 1994.

[9] Lawrence Davis, *Handbook of genetic algorithms.* NY: Van Nostrand Reinhold, 1991.

[10] R. Buse, Z.Q. Liu, and J. Bezdek, "Word recognition using fuzzy logic," *IEEE Trans. Fuzzy Systems*, vol. 10, no. 1, Feb. 2002.

[11] S. Yao, C.J. Wei, and Z.Y. He, "Evolving wavelet neural networks for function approximation," *Electron. Lett.*, vol.32, no. 4, pp. 360-361, Feb. 1996.

[12] P.D. Gader and M.A. Khabou, "Automatic feature generation for handwritten digit recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, no. 12, pp. 1256-1261, Dec. 1996.

[13] B. Zhang, M. Fu, H. Yan, and M.A. Jabri, "Handwritten digit recognition by adaptive-subspace

self-organizing map (ASSOM)," *IEEE Trans. Neural Networks*, vol. 10, no. 4, pp. 939-945, Jul. 1997.

[14] D.R. Lovell, T. Downs, and A.C. Tsoi, "An evaluation of the neocognitron," *IEEE Trans. Neural Networks*, vol. 8, no. 5, pp. 1090-1105, Sep. 1997.

[15] L. Holmström, P. Koistinen, J. Laaksonen, and E. Oja, "Neural and statistical classifiers – Taxonomy and two case studies," *IEEE Trans. Neural Networks*, vol. 8, no. 1, pp. 5-17, Jan. 1997.

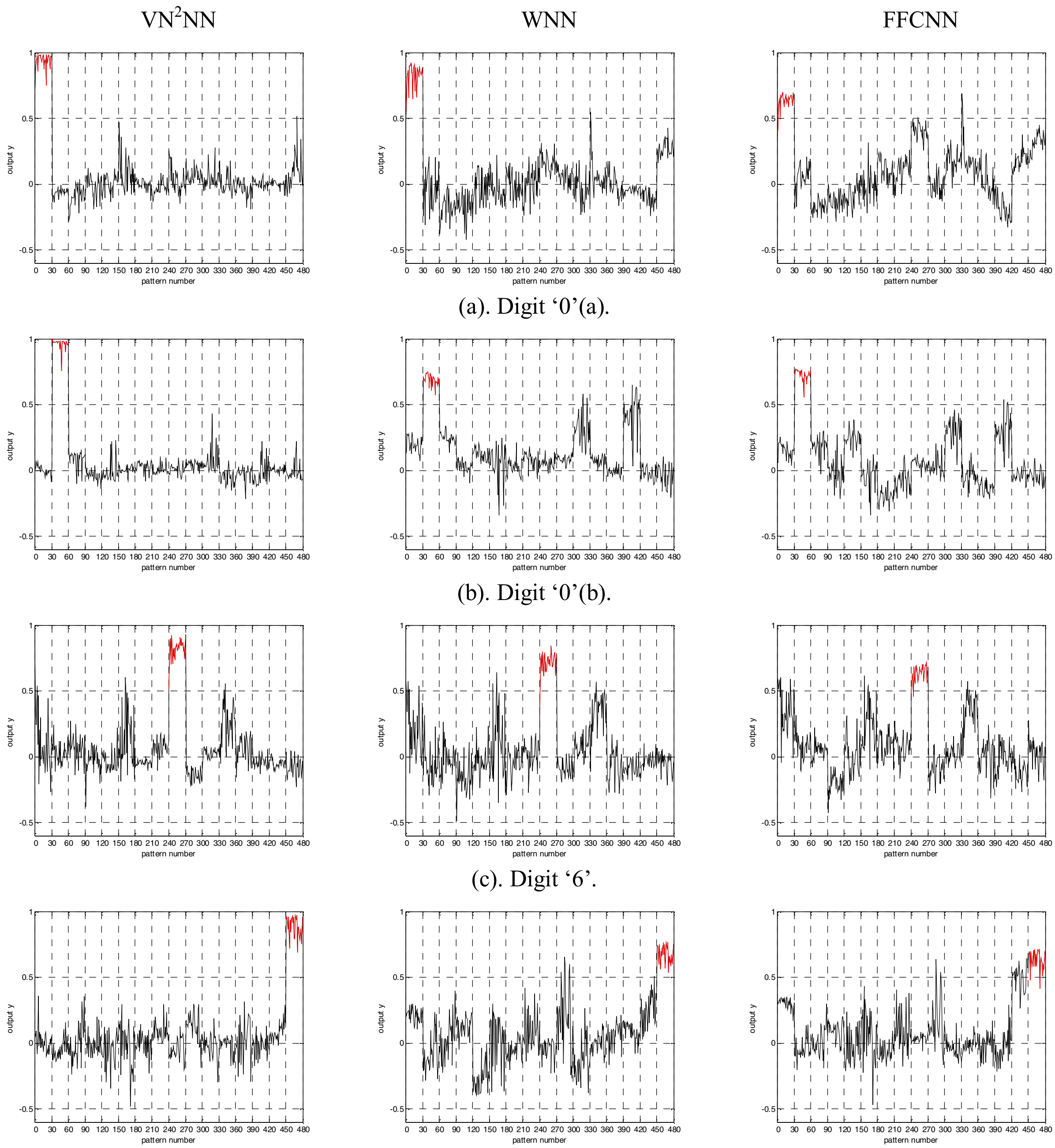
[16] C.K. Ho and M. Sasaki, "Brain-wave bio potentials based mobile robot control: wavelet-neural network pattern recognition approach," in *Proc. IEEE Int. Conf. System, Man, and Cybernetics*, vol. 1, Oct 2001, pp. 322-328.

TABLE I. AVERAGE TRAINING AND TESTING ERROR FOR VN^{2}NN, WNN AND FFCNN.

	VN^{2}NN		**WNN**		**FFCNN**	
	MSE	Recognition accuracy rate	MSE	Recognition accuracy rate	MSE	Recognition accuracy rate
Average training	0.0157	97.38%	0.0309	93.23%	0.0360	91.50%
Best training	0.0139	98.06%	0.0278	94.13%	0.0326	93.06%
Average testing	0.0186	96.96%	0.0329	93.92%	0.0374	91.25%
Best testing	0.0171	97.29%	0.0308	94.38%	0.0361	92.92%

TABLE II. AVERAGE RECOGNITION ACCURACY RATE OF THE VN^{2}NN, WNN AND FFCNN. (THE UNDERLINED RESULTS HAVE MORE THAN 4% IMPROVEMENT.)

	VN^{2}NN		**WNN**		**FFCNN**	
	Recognition accuracy rate (%) of the average network for the training patterns	Recognition accuracy rate (%) of the average network for the testing patterns	Recognition accuracy rate (%) of the average network for the training patterns	Recognition accuracy rate (%) of the average network for the testing patterns	Recognition accuracy rate (%) of the average network for the training patterns	Recognition accuracy rate (%) of the average network for the testing patterns
0(a)	97.35	99.33	97.00	98.67	96.05	94.00
0(b)	97.80	98.67	97.80	98.67	99.40	99.33
1	95.20	98.67	73.60	77.33	61.70	57.33
2	97.60	99.33	98.15	77.33	97.65	100.0
3	96.40	96.67	93.20	94.67	87.60	97.33
4	99.00	94.00	98.25	98.00	98.50	96.66
5(a)	98.40	94.67	95.60	95.33	90.90	90.00
5(b)	97.20	100.0	95.50	100.0	97.40	100.0
6	94.10	95.33	85.60	85.33	85.20	84.00
7	99.05	98.67	94.90	93.33	95.40	93.33
8(a)	99.85	95.33	98.40	92.67	97.45	94.67
8(b)	99.35	90.67	99.70	91.33	100.0	95.33
9	90.00	95.33	76.10	86.00	78.40	83.33
Back Space	98.20	96.67	97.60	100.0	98.05	98.67
Return	100.0	98.67	99.45	98.67	98.20	94.67
Space	98.60	99.33	91.15	95.33	82.15	80.67
Overall	**97.38**	**96.96**	**93.23**	**93.92**	**91.50**	**91.25**



(a). Digit ‘0’(a).

(b). Digit ‘0’(b).

(c). Digit ‘6’.

(d). Digit ‘space’.

Fig. 9. Output values of the VN^{2}NN, WNN, and FFCNN for the 480 (30 for each type) testing graffiti.