www.icgst.com

Hand Gesture Recognition Using Fuzzy Neural Network

Nguyen Dang Binh, Toshiaki Ejima
Intelligence Media Laboratory, Kyushu Institute of Technology
680-4, Kawazu, Iizuka, Fukuoka 820, JAPAN
[ndbinh, toshi]@mickey.ai.kyutech.ac.jp

Abstract

In this paper, we introduce a new approach to gesture recognition based on incorporating the idea of fuzzy ARTMAP [1] in the feature recognition neural network and Hussain and Kabuka [3]. It has already shown that neural networks are relatively effective for partially corrupted images. However, a distinct subnet is created for every training pattern. Therefore, a big network is obtained when the number training patterns is large. Furthermore, recognition rate can be hurt due to the failure of combining features from similar training patterns. We would like to improve this technique by considering input images as fuzzy resources for hand gesture recognition. Training patterns are allowed to be merged, based on the measure of similarity among features, resulting in a subnet being shared by similar patterns and network size is reduced and recognition rate is increased. The high gesture recognition rates and quick network retraining times found in the present study suggest that a neural network approach to gesture recognition be further evaluated.

Keywords: Gesture recognition, Fuzzy ARTMAP, matching degree, recognition-by-parts.

1. Introduction

Human-Computer Interaction (HCI) is getting increasingly important as computer's influence on our lives is becoming more and more significant. For HCI, machine-recognition of hand gestures brings vision of more accessible computer systems as an alternative controller, especially when conventional controls will be possible using keyboard and mouse alone. Among a variety of gestures, hand gesture is the most expressive and the most frequently used one. The objective of this effort was to explore the utility of a feature recognition neural network-based approach to the recognition of hand gestures.

We propose a new approach to hand gestures recognition based on feature recognition neural network in which develop a neural network architecture, which incorporates the idea of fuzzy ARTMAP [1] in feature recognition neural network [3]. Firstly, gray level for each pixel in a monochrome image will be used to determine membership function value. Hence the input layer of neural network contains fuzzy entries corresponding to each pixel. Secondly, we divide an initial physic image G to regions for obtaining a logical image representation P. It is very important that the size of logical images is invariant for various physic images. Further more, the transformation process from G to P is based on defuzzification. It is clear that neural network technique is more effective for logical images than for physical images. Thirdly, an initial physical image is also divided into regions. For each region some characteristic features will be extracted using neural network. Combining the features of all regions of image forms results. Training patterns are allowed to be merged based on the measure of similarity among features. A subnet is allowed to be shared by similar patterns. A minimal number of subnets is learned automatically to meet accuracy criteria. Therefore, the network size can be reduced and training patterns better. We implemented already these techniques for hand gestures recognition. Experimental results have shows the efficiency of proposed techniques.

The organization of the rest of the paper is as follows. In Section 2 describes acquisitions and preprocessing, problem statement. In section 3, presents the architecture of our network. We discuss how a trained network performs and hand gesture recognition on input patterns in section 4. The next section presents results of experimentation with our system. We summarize the contribution of this work and identify areas for further work in the conclusion section

2. Acquisitions and Preprocessing, Problem Statement

In our work, gestures are required using a desktop camera observing a stationary background, under room lighting. Skin pixels are detected as those colors are inside an axis-aligned box in HSV color space. This is reliable for most pixels in a typical hand image, but results in lost pixels at the edges of the hand.

For the remainder of the paper, images will be represented as 2D vectors. All images will be considered to be the binarized, canonicalized versions as in the

figure 1a. We wish to recognize gestures based on a previous acquired training set. Let us assume that we have K gestures, and for the k^{th} gesture we have obtained M_k training examples; N is the number of image pixels 32x32 here which are numbered 1,2…1024 from left to right and top to bottom. Any 4 x 4 pixels is called a subpattern and the subpatterns are numbered 1, 2… 64 (Fig.1. (b)).

The training examples are denoted x_{mk} for m=1…N and k=1…K. A classifier, which explains the training data perfectly, will present in Section 4.2. The algorithm performs the following steps in succession to recognize any pattern:

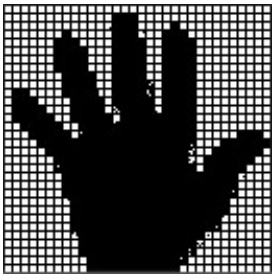
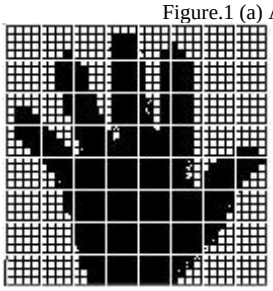


Figure.1 (a) An example pattern



1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

Figure.1 (b) Its 64 nominal subpatterns

1) Learns all the training patterns and remembers their subsections;

2) Divides the new input pattern into 64 segments as described above;

3) Compares each of the segments individually against the corresponding segments of the trained patterns to compute any match;

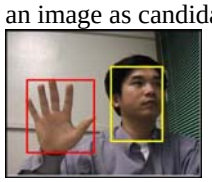
4) Finds the total number of segments that matched for any pattern and finds the closest overall match.

We trust in ingenuity and a representative training set to find classifiers for which good performance on test examples. The task of this paper is to define an accurate classifier, which may be computed rapidly enough to allow real time operation.

3. Hand Tracking

The general engineering of the system means that preprocessing is reliable on every frame. Lighting is controlled with just enough care to ensure that most skin pixels are detected using sample image processing. This work, we develop a real time hand tracking method based on the Kaman filter and hand blobs Analysis, which is robust and reliable on hand tracking in stationary background and then the hand region extraction fast and accurately. We need to consider the trade-off between the computation complexity and robustness. The segmented image may or may not include the lower arm. To avoid this, the user is wearing long-sleeves shirt. This is necessary because it is not used a wrist-cropping procedure. We must first extract hand region in each input image frame in real time.

Extracting hand region**.** Extracting hand based on skin color tracking [10], we drew on ideas from robust statistics and probability distributions with Kalman filter in order to find a fast, simple algorithm for basic tracking. Our system identifies image regions corresponding to human skin by binarizing the input image with a proper threshold value. We then remove small regions from the binarized image by applying a morphological operator and select the regions to obtain an image as candidate of hand.



(a)The origin frame

(b) The binary image

(c)Hand extracted

Figure 2: Hand tracking

4. Development of Feature Recognition Algorithm

4.1 Network architecture

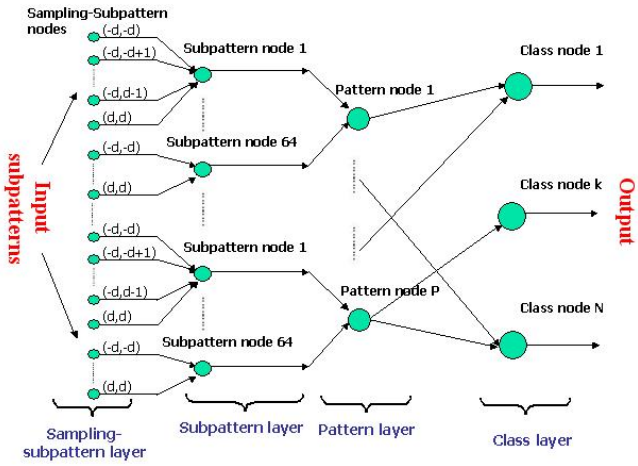


Figure 3: The architecture of GRFNN

A gesture recognition fuzzy neural network (GRFNN) consists of four layers, excluding the input layer, as show in Fig.3. Subpatterns of an input pattern are presented to the sampling-subpattern layer. Sampling-subpattern nodes take care of the deformation, i.e., shift, noise, or size, of the input pattern. Each subpattern node summarizes the measure of similarity measure between the input pattern and the stored pattern. Subpattern node is responsible for the match between an input nominal subpattern and the stored pattern. However, for tolerating possible deformation of the input pattern, we have to consider the neighboring subpatterns of an input subpattern. Suppose we allow a deformation of up to ± d (d is a positive integer) pixels in either X or Y directions. We have to consider all the neighboring subpatterns within the distance d in order to detect a possible deformation. Each neighboring subpattern is taken care of in a sampling-subpattern node. Therefore, a subpattern node may receive the output of up to $(2d+1)^2$ sampling-pattern nodes. Each subpattern node stores. A sampling-subpattern node weight W, which is shared by all its sampling-subpattern nodes. A sampling-subpattern node computer based on the input pattern and its node weight, a value and outputs the value to the associated subpattern

node. The value computed by a sampling-subpattern node measure the similarity between an input subpattern with distance (dx, dy), -d≤ dx ≤d, -d≤dy≤d from the underlying input nominal subpattern and the node weigh stored in the sub pattern node. A sub pattern node investigates the output values of all its sampling-subpattern nodes and takes. The maximum of them as its output to the third layer (pattern layer).

Similarly, a pattern node in the pattern and the pattern layer reflects the similarity measure between the input pattern and the stored pattern. A pattern node is connected to one category node in the category layer, indicating the class of the input pattern. The third layer contains pattern nodes. A pattern is linked by 64 sub pattern nodes, with link weight W associated with each link. Also, a vigilance parameterρ, 0≤ρ≤1, is associated with each pattern node. Each pattern node receives value from all its sub pattern and computers a number from these values (the neurons in 3 layers computer based on inputs and weight. It have a values, this values will be fired one of neurons in 4 layers). The numbers from all pattern nodes are involved in triggering one of the class nodes, indicating that the input pattern has been appropriately classified (classification). For convenience, we use the following notation in referring to nodes: let N_i be a pattern node. Then $N_{i,j}$ denote the j[th] subpattern node of N_i. $N_{i,j,(d_x,d_y)}$ denotes the sampling-subpattern node of $N_{i,j}$ that takes care of the input subpattern, which is (d_x, d_y). Away from the subpattern. As usual, X is in the horizontal direction and Y is in the vertical direction. A positive (negative) d_x denotes a right (left) shift and a positive (negative) d_y denotes a down (up) shift of the subpattern. We may call $N_{i,j,(d_x,d_y)}$ as the $(d_x, d_y)^{th}$ sampling-subpattern node of $N_{i,j}$. We define a subnet of a pattern node N_k to be the sub-network consisting of the pattern node N_k, its sub pattern nodes and sampling-subpattern nodes, together with all the associated links.

4.2 Network creation and training gesture

Suppose we are given a set of training patterns. Each pattern is represented by a row matrix G of 1024 pixels, and each sub pattern by a row matrix I_j of 16 pixels, $1 \le j \le 64$, namely,

$$P_j = \lfloor P_{j1}, P_{j2}, \ldots, P_{j64} \rfloor \text{ and } G = [P_1, P_2, \ldots, P_k]$$

Where $P_{j,k}$ is the normalized gray level of the corresponding pixel, i.e, $P_{j,k} \in \{-1,1\}$ 1≤k≤64, 1≤j≤64. With 1 representing black and –1 representing white. For convenience, we represent the input to a sampling-subpattern node $N_{i,j,(d_x,d_y)}$ by $P_j(d_x,d_y)$, $P_j(0,0)$ may be abbreviated as P_j .. Since we are doing supervised classification, it is assumed that the class of each input training pattern is known. As mentioned earlier, each sub pattern node stores a node weight shared by all its sampling-subpattern nodes. For a subpattern node $N_{i,j}$, its node weight $W_{i,j}$ is defined to be

$$W_{i,j} = \lfloor W_{i,j,1}, W_{i,j,2}, \ldots, W_{i,j,64} \rfloor \quad (1)$$

Where $W_{i,j,k} \in Z$, $1 \le k \le 64$, is an integer. Support an input training pattern G with class C is presented to the network. Each sampling-subpattern node $N_{i,j,(d_x,d_y)}$ computers its output $Out_{i,j,(d_x,d_y)}$ by

$$Out_{i,j,(d_x,d_y)} = \frac{W_{i,j} * P_j^T(d_x, d_y)}{\alpha + |W_{i,j}|} \quad (2)$$

Where $|W_{i,j}| = \sum_{k=1}^{64} |W_{i,j,k}|$, the superscript T stands for matrix transposition and α≥0. Since each element of $P_j^T(s_x, s_y) \in \{-1,1\}$ is either 1 or –1, the following relationship holds: $-|W_{i,j}| \le W_{i,j} * P_j^T(d_x, d_y) \le |W_{i,j}|$

Therefore, we have $-1 \le Out_{i,j,(d_x,d_y)} \le 1$. Apparently, $Out_{i,j,(d_x,d_y)}$ measure the similarity between $P_j(d_x,d_y)$ and the node weight $W_{i,j}$ stored in $N_{i,j}$. The more $P_j(d_x,d_y)$ is similar to the stored weight $W_{i,j}$, the closer $Out_{i,j,(d_x,d_y)}$ is to 1. On the contrary, the more $P_j(d_x,d_y)$ is deferent from $W_{i,j}$, the closer $Out_{i,j,(d_x,d_y)}$ is to –1. All the outputs of $N_{i,j,(d_x,d_y)}$ *(sampling-subpattern nodes)* are sent to respective $N_{i,j}$ *(subpattern nodes)*. Each sub pattern node $N_{i,j}$ takes the maximum value of $Out_{i,j,(d_x,d_y)}$ *(all its input)*, by the way:

$$Out_{i,j} = \max(Out_{i,j,(-d,-d)}, \ldots, Out_{i,j,(0,0)}, \ldots, Out_{i,j,(d,d)}) \quad (3)$$

and sends this value to its pattern node N_i the way $O_{i,j}$ is computed reflects the sprit of recognition by parts. Also, this account for the tolerability of GRFNN of deformation, noise, and shift in position. Obviously, -1 ≤ $Out_{i,j} \le 1$ for every possible i and j. Let the Priority index Pr_i for the pattern node N_i be defined by

$$Pr_i = \sum_{j=1}^{64} (3 \times Out_{i,j} - 2)^{1/3} \quad (4)$$

With j ranging over the 64 subpatterns nodes of N_i. Using priority indexes makes the training procedure more efficient. The priority indexes of all pattern nodes stored in decreasing order and placed in the priority list. Suppose the largest priority index in the priority list is Pr_k. Let the pattern node corresponding to P_k be N_k, the class for N_k be C_k and N_k's vigilance be ρ_k.

Compute the following matching degree M_k for N_k

$$M_k = \frac{\sum_{j=1}^{64} (\omega_{k,j} \wedge Out_{k,j} + 1)}{\sum_{j=1}^{64} (\omega_{k,j} + 1)} \quad (5)$$

Where $\omega_{k,j}$ is link weight between N_k and $N_{k,j}$. The operator Λ is defined as in [9] $\omega_{k,j} \wedge Out_{k,j} = \min(\omega_{k,j}, Out_{k,j})$. Since $(\omega_{k,j} \wedge Out_{k,j}) \le \omega_{k,j}$; $\omega_{k,j} \ge -1$ (to be clear later), we have $0 \le M_k \le 1$.

Then we have the following cases:

1) If $M_k \geq \rho_k$ and $C=C_k$ then we modify the pattern stored in the subnet of N_k by changing the associated node weights and link weights as follows:

$$\begin{aligned} W_{k,j} &\leftarrow W_{k,j} + P_j(d_x, d_y) \\ \omega_{k,j} &\leftarrow \omega_{k,j} \wedge Out_{k,j} \end{aligned}, \quad 1 \leq j \leq 64 \qquad (6)$$

Where $P_j(d_{jx}, d_{jy})$ is the input to $N_{k,j,(d_{jx},d_{jy})}$ whose output value to $N_{k,j}$ is the largest among the sampling-subpattern nodes of $N_{k,j}$. Then we are done with the input training pattern A. Equation above intends to increase the outputs value of $N_{k,j,(d_{jx},d_{jy})}$ more than the output values of the other sampling-subpattern nodes of $N_{k,j}$ when an identical input pattern is presented to the network next time.

If the first case has never occurred, then it means that the training pattern should not be combined in any existing pattern subnet. In this case, we create a new pattern subnet for storing this training pattern. Let N_n be the pattern node of this new subnet. The node weight $W_{n,j}$ of the jth subpattern node of N_n is initialized by

$$W_{n,j} \leftarrow I_j, \ 1 \leq j \leq 64 \qquad (7)$$

and the jth link weight, $W_{n,j}$ of N_n is initialized to 1, namely,

$$\omega_{n,j} \leftarrow 1, \ 1 \leq j \leq 64 \qquad (8)$$

and the vigilance ρ_n associated with N_n is set to an initial value which depends on how much degree of fuzziness is allowed for N_n to include subnet the other inputs. If the network already contains a class node for C, then we connect N_n to this class node; otherwise we create a class node for C and connect N_n to it.

2) If $M_k \geq \rho_k$, $C \neq C_k$, $M_k < 1$, then we increase ρ_k as follows:

$$\rho_k \leftarrow M_k + \beta, \qquad (9)$$

Where β is a very small positive real number. With this increase in ρ_k, the next time when an identical input pattern is presented to the network, M_k would be no longer greater than or equal to ρ_k.

3) If $M_k \geq \rho_k$, $C \neq C_k$, $M_k = 1$, then the modification becomes

$$\begin{aligned} \rho_k &\leftarrow 1 \\ \omega_{k,j} &\leftarrow O_{k,j} + \beta, \quad 1 \leq j \leq 64, \end{aligned} \qquad (10)$$

Where β is a very small positive real number. In this case, M_k would be slightly less than ρ_k the next time when an identical input pattern is presented to the network, since the numerator of (4) takes the smaller of $\omega_{k,j}$ and $O_{k,j}$.

4) If M_k is smaller than the vigilance ρ_k of N_k, then we do not modify anything in the subnet of N_k.

If any of the last three cases occurs, we select the nest highest priority index in the priority list and continue the above process iteratively until either the first case occurs or every member of the priority list is considered. Notice that priority indexes help training in the following way. For a training pattern G with class C if no class node of C exists in the network, then we do not need to go through the above procedure at all. We are creating a new subnet by applying (7) and (8). Next time when an identical pattern is presented, it is sure that this subnet will be activated since it will be the first element in the priority list. This will not cause any problem for the recognition phase, since priority index are applied in the same way as described in the next section.

Two or more pattern nodes may connect to an identical class node indicating that the patterns stored in these subnets are in the same class. This case occurs if training patterns of a class are clustered in groups. However, the patterns in one cluster are not similar enough, measured by matching degrees, to the patterns in another cluster. As a result, each cluster results in a different subnet.

The above procedure is iterated with the training pattern set until the networks is stable, i.e., none of the vigilance in the next work changes.

It usually requires thousands of iteration cycles for stability of the network. The value of the parameters and learning constants during the training process play an essential role in the convergence and the results of the network. In the Section 4.4 we will propose a fuzzy control scheme to change these parameters and learning constants during training.

4.3 The effects of the learning parameters to the network performance

The network are determined by choice parameter $\alpha \geq 0$; a learning rate parameter $\beta \in [0,1]$; and a vigilance parameter $\rho \in [0,1]$. The values of vigilance parameters are adjusted in the training phase of the network, as will be described above. Similar to fuzzy ARTMAP, vigilance parameters ρ control the accuracy of classifying input training patterns. In the (2), the choice parameter α is chosen close to 0, then the first category (3) will always be the category whose weigh vector $W_{i,j}$ is the largest coded subset of the input vector P_j. As show below, the choice function $Out_{i,j}$ in (3) can be interpreted as a fuzzy membership of the input P_j. The limit $\alpha \to 0$ is called the conservative limit because small values of α tend to minimize recording during learning. In our research we choice $\alpha=0$. The choice function $Out_{i,j}$ in (3) primarily reflects the degree to which the weight vector $W_{i,j}$ is a fuzzy subset of the input vector P_j. Resonance occurs if match function (5) the chosen category meets the vigilance criterion $M_k \geq \rho_k$. Mismatch reset occurs if $M_k < \rho_k$ then a new index $Out_{i,j}$ is the chosen, by (3). The search process continues until the chosen $Out_{i,j}$ satisfies $M_k \geq \rho_k$. We consider the general equation used to correct weight in the network layer:

$$\omega_{k,j}^{(new)} = \eta(O_{k,j} \wedge \omega_{k,j}^{(old)}) + (1-\eta)\,\omega_{k,j}^{(old)} \qquad (11)$$

Fast learning corresponds to setting $\eta=1$. The learning law used in the EACH system of Salzberg [6][7] is equivalent to (6) in the fast learning limit. GRFNN is affected by the value of the shift parameter, **d**, for GRFNN was set to 2. A large **d** may, on one hand, hurt recognition rate due to different patterns being group into the same class, but may, on the other hand, do good for recognition rate due to high resistibility to shifts in the position. The size and performance of GRFNN is also affected by the value of the initial vigilance. The initial vigilance used in GRFNN was set to 0.95. A large value

of vigilance allows a small number of patterns to be combined in a subnet, resulting in a large network size and a low tolerance for variation. On the contrary, a small value of vigilance encourages many patterns to be combined in a subnet, resulting in a small network size and high resistibility to noise.

4.4 Fuzzy control for learning parameter

We propose fuzzy control concepts and techniques to determine adaptively the learning parameters so that the performance of the neural network will be improved. A general architecture for such system is shown in figure 4. Although as follows fuzzy ARTMAP, learning always converges because all adaptive weights are monotonically non-increasing. With additional processing, this useful stability property could lead to the unattractive property of category proliferation as too many adaptive weights converge to zero.

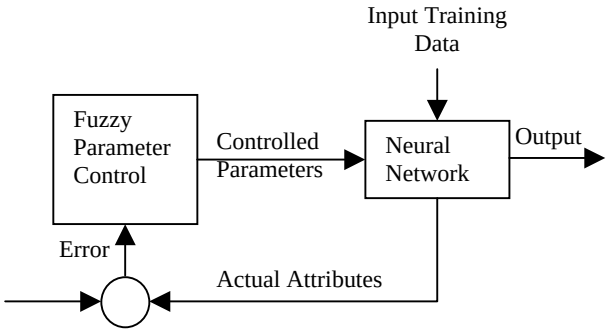


Figure 4: System using a fuzzy controller for the adaptation of neural learning parameter

The fuzzy controller is used to adapt the values of learning parameters according to certain heuristics. There is usually an attribute of the neural network that we wish to control. In our case they are a learning rate parameter β and a vigilance parameter ρ. Several fuzzy IF- THEN rules in the fuzzy controller use in our algorithm.

5. Gesture Recognition

After the training phase, the network is ready for recognizing unknown pattern. Suppose G is a normalized input pattern presented to the trained network.

Firstly, the priority indexes are stored, as before, in decreasing order in the priority list. Suppose the largest priority index in the priority in list is Pr_k. Let the pattern node corresponding to Pr_k be N_k, the class for N_k be C_k with vigilance ρ_k.

Secondly, If M_k is greater than or equal to ρ_k, then the input pattern is classified to C_k and we are done. If M_k is less than ρ_k, then we are selecting the next highest priority index in the priority list and proceed the above iteratively.

Finally, if network cannot find a pattern node with the matching degree being greater than or equal to its vigilance, then we classify the input pattern to the class represented by the class node connected by the pattern node with the highest priority index.

6. Experimental Results

We evaluate the performance of our recognition by testing its ability to classify gestures of both training and testing data. The effect of the learning parameters to the network performance used to describe the system on its performance is studied. In addition, we discuss some problems in the performance of some gestures due to the similarities between them. We also compare with other approaches. After network implementation into gesture recognition, we estimate the result of network through patterns. In this section we present the results of recognition of 36 gestures including the America sign language (ASL) letterspelling alphabet and digits (Fig. 5).

6.1 Results of GRFNN based hand gesture recognition

Data set: The data set used for training and testing the recognition system consist of binary images for all the thirty six gestures shown in Fig. 5. Samples for each gestures ware taken from database in [8]. For a single gesture, we have collected 200 isolated samples from 36 gestures (in total, 7200 isolated samples for all gestures in the system) and have partitioned them into training and test sets, as shown in Table 1. We are tested with many categories and styles performance of data patterns, for example, noise gestures, deformation hand gesture and rotate θ angle….

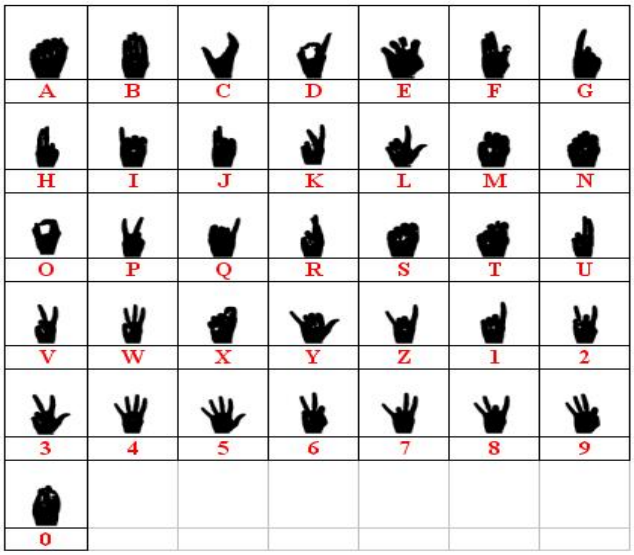


Figure 5: The ASL gestures set to be recognized

The recognition ratio is the ratio of correctly recognized gestures over the number of the input gestures:

Recognition rate: We evaluate the performance of our system based on its ability to correctly recognized gestures to their corresponding input gestures. The metric that we use to accomplish this job is called the recognition rate. The recognition rate is defined as the ratio of the number of correctly recognized gestures to the total number of input gestures sample.

$$\text{Recognition rate} = \frac{\text{No. of correctly recognized gestures}}{\text{Total number of input gestures}} \times 100\%$$

Experiments show that the system can work correctly with sufficient training data, to extend the training database is only a compromise to reduce the lack of training data. Most of the misclassified samples correspond to the gestures that are similar to each other. As an example, Fig.5 shows the gestures “M” and “N”. Because these gestures are similar, their corresponding features are also similar. Therefore, it is probable for a sample of the gesture “M” to be classified as “N” or vice

versa. The same true for the gestures “S”, “T” and “X”, and gestures “A” and “M”. Due to size normalization make distortions to the original data, and deteriorate the performance of the system. These distortions are due to either the estimation error or the binarization of the normalize. In try to enhance the performance of the problematic gestures, it was noticed that some of these gestures perform better using relatively higher number of rules.

6.2 Comparison with other approaches

Not many vision-based approaches have been reported for hand gesture recognition. GRFNN represent robust features since they tend to be sensitive to image noise as well as image rotation or shifts. The complexity of the operations is reduced, due to the patterns have allowed to be merged based on the measure of similarity among features. Hidden Markov Models (HMMs) have been recently studied for hand gesture recognition has many successes [5][12]. However, little effort has been devoted to feature development and incorporation of subcharacter HMMs into neural network. Compared to template based methods and GRFNN, our proposed system offers a more flexible framework for gesture recognition, and can be used more efficiently in scale invariant systems. In other worlds, the GRFNN offers promising potential to solve hand gestures recognition.

Gestures	Test Data	Error	Correct	Recognition (%)
“A”	100	0	100	100
“B”	100	4	96	96
“C”	96	1	95	98,96
“D”	100	0	100	100
“E”	100	1	99	99
“F”	100	5	95	95
“G”	100	2	98	98
“H”	100	3	97	97
“I”	101	0	101	100
“K”	92	8	84	91,3
“L”	100	0	100	100
“M”	98	6	92	93,88
“N”	99	12	87	87,88
“O”	101	6	95	94,06
“P”	100	0	100	100
“Q”	100	0	100	100
“R”	96	0	96	100
“S”	100	15	85	85
“T”	99	12	87	87,88
“U”	96	0	96	100
“V”	101	0	101	100
“W”	101	0	101	100
“X”	100	5	95	95
“Y”	101	0	101	100
“Z”	101	0	101	100
“0”	99	0	99	100
“1”	100	0	100	100
“2”	101	0	101	100
“3”	101	0	101	100
“4”	100	0	100	100
“5”	101	0	101	100
“6”	99	0	99	100
“7”	102	0	102	100
“8”	100	0	100	100
“9”	101	0	101	100
Total	**3438**	**80**	**3406**	**92,19**

Table: Test data for gestures recognition

7. Conclusions

We have proposed an improvement to Hussain and Kabuka’s method for hand gesture recognition that is shown to be robust for ASL gestures. The use of vigilance parameters and matching degrees permits the combination of similar training patterns automatically in the same subnet. Consequently, networks obtained by our method can be better than those obtained by Hussain and Kabuka’s method, and it is not required that training patterns be preselected by human experts. Moreover, with the embodiment of fuzzy theories, the recognition rate of our method can be better than that of Hussain and Kabuka’s method. In fact, the complexity of the operations is reduced, due to the patterns have allowed to be merged based on the measure of similarity among features. Consequently, we believe that the proposed model can be successfully used in real-time hand gestures recognition applications. The future work consists of the evaluation of the proposed model on an online database of the hand gesture images.

References

[1] G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynold, and D. B. Rosen. "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," IEEE Trans. Neural Networks, Vol. 3, No 5, pp.698-713, September 1992.

[2] G. A. Carpenter, S. Grossberg, and J.H. Reynolds. "ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network," Neural Networks, vol. 4, pages 565-588, 1991

[3] B. Hussain and M. R. Kabuka, "A novel feature recognition neural network and its application to character recognition," IEEE Transaction on Pattern Analysis and Machine Intelligence. Vol. 16, pages 98-106, January 1994.

[4] O. Al-Jarrah, A.Halawani. Recognition of gesture in Arabic sign language using neuro-fuzzy system," Artificial Intelligence 133, pages 117-138, 2001.

[5] Chan Wa Ng, S. Rangganath. Real-time gesture recogntition system and application. Image and Vision computing 20 pages 993-1007, 2002.

[6] S. L. Salzberg, Learning with nested generalized examples," Ph.D. (Technical Report TR-14-89), Department of Computer Science, Harvard Universtity, Cambridge, MA, 1989.

[7] S. L. Salzberg," A nearest hyperrectangle learning method," Machine Learning, vol. 6,pages 251-276, 1991.

[8] R. Lockton, A. W. Fitzgibbon. Real-time gesture recognition using deterministic boosting, Proceedings of British Machine Vision Conference , 2002.

[9] H. J.Zimmerman. Fuzzy Set Theory and It’s applications (2nd ed. Norwell, MA: Kluwer, 1991).

[10] G.R. Bradski, S. Clara. Computer Vison Face Tracking For Use in a Perceptual User Interface. Intel Technology Journal Q2'98.

[11] S. J. Lee and H. L. Tsai. "Pattern fusion in feature recognition neural networks for handwritten character recognition," IEEE Trans. Systems, Man, and Cybernetics, Part B, Vol. 28, No 4, pages 612-617, August 1998.

[12] Starner, T. and Pentland. 1995. Real-Time American Sign Language Recognition from Video Using Hidden Markov Models, TR-375, MIT Media Lab.