Gesture Recognition using Recurrent Neural Networks

Kouichi Murakami and Hitomi Taguchi

Human Interface Laboratory
Fujitsu Laboratories LTD. Kawasaki
1015, Kamikodanaka, Nakahara-ku,
Kawasaki, 211, Japan
E-mail: unicon@flab.fujitsu.co.jp

ABSTRACT

A gesture recognition method for Japanese sign language is presented. We have developed a posture recognition system using neural networks which could recognize a finger alphabet of 42 symbols. We then developed a gesture recognition system where each gesture specifies a word. Gesture recognition is more difficult than posture recognition because it has to handle dynamic processes. To deal with dynamic processes we use a recurrent neural network. Here, we describe a gesture recognition method which can recognize continuous gesture. We then discuss the results of our research.

KEYWORDS: Artificial reality, gesture recognition, sign language, and neural networks.

1. INTRODUCTION

Research on artificial reality is being made to develop new media which can provide people with the sensation of doing something they are not actually doing [1]. One of the key technologies to provide reality is natural interaction between human and objects or living-things in the virtual world constructed in a computer. Our objective to study gesture recognition is to establish natural interaction in the virtual world. There are several types of human gestures: human movement in a virtual world, interaction with virtual objects such as grasping, communication with virtual living-things such as indicating *come here*, and expressing human feelings. We first applied the gesture recognition to Japanese sign language, because gestures used are well defined.

In general, sign language includes finger alphabets represented by different postures formed with hands and words represented by postures and movements of hands. The finger alphabet is represented by posture that can be processed statically. We developed a posture recognition system. Finger alphabet recognition was performed using a neural network. The results showed a recognition rate of 98% for registered people.

Although finger alphabet recognition can be used to indicate human intention to computers, it is not sufficiently natural. Thus, the gesture which represents words must be recognized. We have developed a method to recognize sign language word recognition and gesture recognition. Gesture recognition is more difficult than posture recognition, because it has to consider movement as well as posture.

We first mention the finger alphabet recognition system we had developed. Next we point out problems in sign language word recognition, and describe the gesture recognition system. Finally, we discuss the results of our experiments.

2. FINGER ALPHABET RECOGNITION

A system for recognition Japanese finger alphabets (42 characters) in the sign language was constructed [2]. The computational model called parallel distributed processing (PDP), using data structures called neural networks, was used for this system because of its learning capabilities. It has been applied to models of pattern recognition. These models assume that information processing takes place through the interactions of a large number of simple processing elements called units, each sending excitatory and inhibitory signals to other units.

2.1. Principle of neural networks

Before discussing the method, we will introduce the PDP model needed for subsequent discussion. See [3]

for details.

A neural network consists of a large number of processing units with multiple inputs and one output. As indicated in Figure 1, the n-input signals are represented by an n-dimensional vector $\mathbf{X} = (x_1, x_2, \ldots, x_n)$. The output from unit j is given by:

$$X_j = \Phi(\sum_i W_{ji} X_i + \theta_j) \quad (1)$$

where X_i is the output of unit i, W_{ji} is the weight of the link from unit i to unit j, and θ_j is the bias of unit j, which is considered below as one of the weights to be adjusted. In general, the conversion function Φ is a sigmoidal nonlinear function with the characteristics indicated in the figure. The neural network consists of multiple processing units linked to each other with weights. Figure 2 shows a three-layer network that is used here. These layers are called the input, hidden, and output layers. The units on a layer are linked to the units on the adjacent layer on a one-to-one basis.

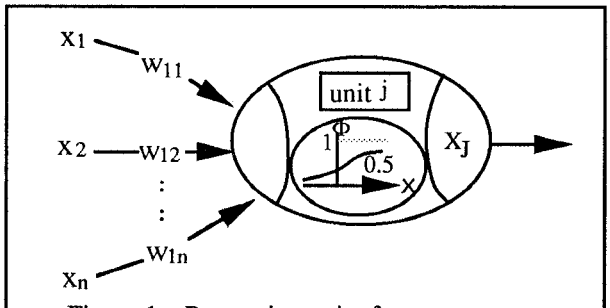

Figure 1. Processing unit of neurocomputer.

Processing in this neural network is as follows: Given an input vector to the input layer, an output vector from the output layer is obtained according to equation(1). This is called forward calculation. This processing can be consider as performing a mapping function, $\mathbf{V} = T(\mathbf{U})$ where $\mathbf{U}$ is the input vector and $\mathbf{V}$ is the output vector. For this calculation, the weights of the network must be given. Let $\mathbf{W}$ be the set of weights W_{ji} and biases θ_j for all units at all layers of the network. These weights can be determined automatically from the input and the desired output vector. The back propagation method is widely used as a learning algorithm to determine the weights of networks. With this method, a training set $\mathbf{P}$ of several patterns consisting of the input vector $\mathbf{U}$ and the desired output vector $\mathbf{V}^{desired}$, is taught to the system. Once learning is completed, the network weight set $\mathbf{W}$ is determined and the mapping function T from input $\mathbf{U}$ to output $\mathbf{V}$ is also determined. The square error between the resulting output vector $\mathbf{V}$ and the desired output vector $\mathbf{V}^{desired}$ is defined as the energy of the network E as follows:

$$E = \sum_p \sum_i (V_{i,p} - V^{desired}_{i,p})^2 \quad (2)$$

Here, index p indicates an input-output pattern in the training set $\mathbf{P}$, and i indexes the components of the output vector $\mathbf{V}$. The back propagation method adjusts the weights to minimize this energy E.

The basic idea of the learning algorithm is as follows. It is assumed that the trainer provides the desired values of the output units. An error signal is generated at each output unit by comparing the desired output with the actual output. The weight on the links coming into the output units are then changed by an amount proportional to the error signal. Error signals are then propagated back down these links to the hidden units, and error signals for the hidden units are computed by adding the propagated signals. Essentially, the network assigns blame to units that lead to a large error in the output vector. Units with a large amount of blame change their weights more to compensate for the error. The algorithm is thus a gradient search in a weight space for a set of weights that implements the desired function.

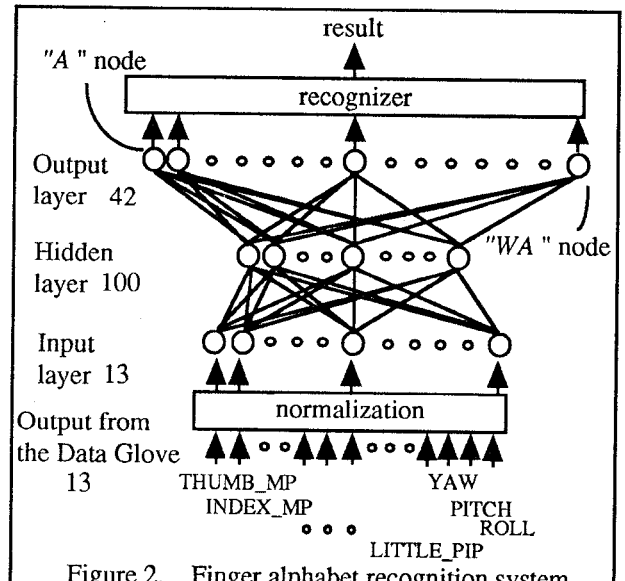

Figure 2. Finger alphabet recognition system.

2.2. Method

The finger alphabet was recognized using a neural network that employed a three-layered back propagation algorithm. Figure 2 shows the system configuration. The 13 data items (10 for bending, 3 for angles in the coordinate) from the Data Glove are normalized between -1.0 to 1.0 according to the dynamic range of each data. For example, 0 ° corresponds to -1.0 and 90 ° corresponds to 1.0 in the case of bending angle. Thirteen nodes in the input layer encode each posture and 42 nodes in the output layer correspond to each

character.

This scheme consists of the learning mode and the recognition mode. In the learning mode, posture and character pairs are given to the system. Values representing the posture are given to the input nodes. At the same time, one of the output nodes corresponding to the presented character is set to 1. In this way, we taught 42 characters showing posture to the system. As will be described later, several instances of posture for the same character are taught to increase the recognition rate. It takes several hours to learn a set of Japanese finger alphabets using a SUN/4 workstation. In the recognition mode, the most activated output node is supposed to represent the character for the given posture. The recognition process is done within several microseconds using a SUN/4 workstation.

2.3. Recognition rate

We evaluated the recognition rate for a specific person using the set of posture data. Table 1 lists the recognition rate for the patterns taught.

Table 1. Recognition rates for registered people

Number of learning patterns	Recognition rate for registered people (%)	Recognition rate for unregistered people (%)
42	71.4	47.8
126	84.8	60.9
206	98.0	77.0

Recognition rate for unregistered people:Mean value for five people.

After 42 patterns corresponding to the set of 42 characters were taught to the network, the recognition rate for the trainers was 71.4%. When three patterns for each character were taught, the recognition rate improved. When additional patterns for poorly discriminated characters were taught for a total 206, the recognition rate rose to 98.0%. This demonstrates the power of this neural network method.

Although the recognition rate for the trainers (registered people) was 98.0%, it was only 77.0% for people not part of the training process (unregistered people). To improve the general recognition rate, we made a dictionary in which data for six randomly selected people were used during the learning mode. We assume that features dependent on each person are generalized by mixing the data. By using a total of 252 learning patterns, six for each character, the general recognition rate increased to 92.9%. The recognition rate of the trainer was obviously higher and was 94.3%.

2.4. Automatic Recognition

There was a problem in the above recognition system. Since the system does not know when sampling starts and ends, the operator has to manually indicate the timing to the machine. A more natural system is required which automatically detects delimiters of the input data.

In the network for finger alphabet recognition, each node of the output layer corresponds to one character. When the value of a node is the greatest in the output layer, the character corresponding to the node is selected to be recognized. To solve the delimiter problem, a new method has been developed. The method is to recognize and output the character corresponding to a node only when the value of the node is the greatest in the output layer and exceeds a predetermined threshold value. As a result, the system does not respond to vague hand postures that have not been learned. This enables continuous and natural finger alphabet recognition.

3. SIGN LANGUAGE WORD RECOGNITION

Ten sign language words shown in Figure 3 were chosen as the words to be recognized in the experiments [4]. These words are vague to some extent, and provide a good sample.

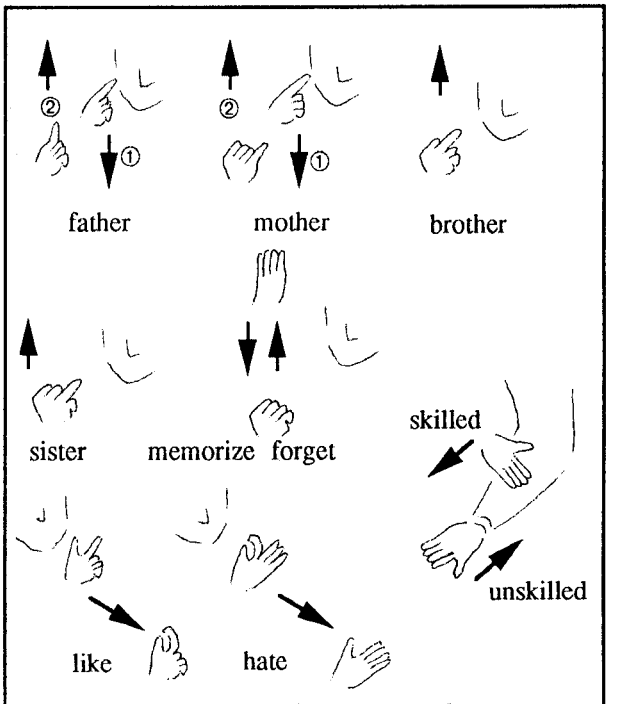

Figure 3. Sign language words.

Three problems must be solved to achieve sign language word recognition.

(1) How to process time-series data

(2) How to encode input data to improve the recognition rate

(3) How to detect delimiters between sign language words

3.1 Processing of time-series data

We use the recurrent neural network proposed by Elman [5] shown in Figure 4 to process time-series data. The architecture of this network is explained below: The hidden layer is allowed to feed back on itself. The value of the hidden units at time t are copied onto the context units. The value of the context units and that of the input units are input to the hidden units at time $t+1$. At time $t=0$ the value of the context units are set to 0. Learning follows the back propagation algorithm.

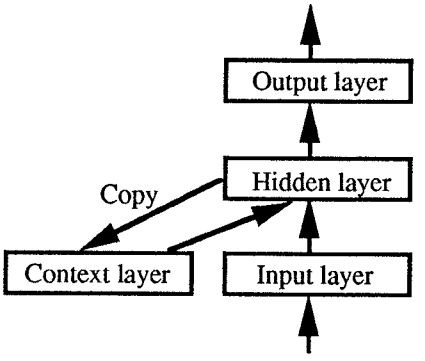

Figure 4. Recurrent neural network.
Each box represents a pool of units. Each forward arrow represents a set of trainable connections from each sending unit to each receiving unit. The backward arrow represents a copy operation from the hidden layer to the context layer.

In this recurrent network, the set of context units provides the system with memory in the form of a trace of processing at the previous time slice. As Rumelhart pointed out, the pattern of activation on hidden units corresponds to an *encoding* or *internal representation* of the input pattern. By the nature of back-propagation, such representations correspond to the input pattern partially processed into features relevant to the task. In this way, the context layer holds the history of the input data. The recurrent network uses the history to enable recognition of the time-series data. Our basic network scheme is shown below:

- Input layer: 16 nodes. The 16 data items (10 for bending, 3 for angles in the coordinate, and 3 for positional data) from the Data Glove are normalized and input.
- Hidden layer: 150 nodes.
- Output layer: 10 nodes. Each node of this layer corresponds to one word, and the word corresponding to the node with the greatest value in the output layer is recognized.

3.2. Encoding of input data

We improved the encoding scheme of input data to raise the gesture recognition rate.

3.2.1. Augmented positional data

Although positional data is not important for posture recognition, positional data is required for gesture recognition because the trajectory of the hand must be tracked. We augmented the positional data using pre-wiring network and two kinds of positional data (the relative position from the starting point and the absolute position from the source) as shown in Figure 5.

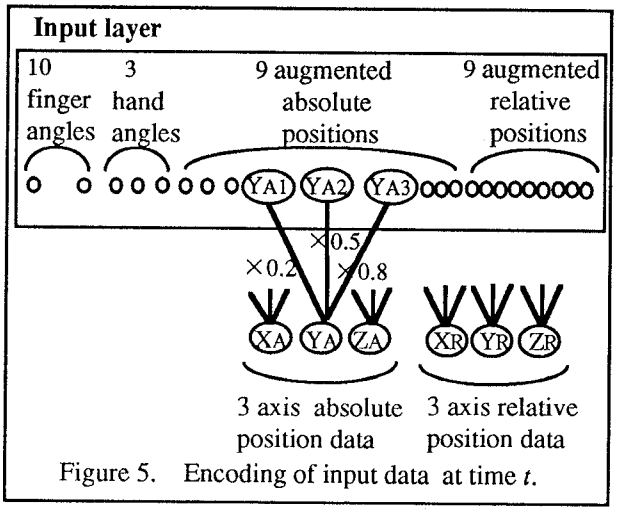

Figure 5. Encoding of input data at time t.

The position was encoded through three pre-wires having different weights (0.2, 0.5, and 0.8) and was given to three input nodes. Since the positional information from the Data Glove is the absolute position from the source, the relative position is calculated and given to the network. The amount of positional data was thus increased to reflect on the trajectory.

3.2.2. Filtering in time-space

The mean error for the Data Glove is about 5 mm and is subject to fluctuation. We tried to stabilize collection of raw data from the Data Glove. Data of three different time points are given to the input layer, which works as a filter or a window as shown in Figure 6. These data points are shifted for the next sampling period. To input the augmented and filtered data, 93 input nodes are used instead of 16 nodes.

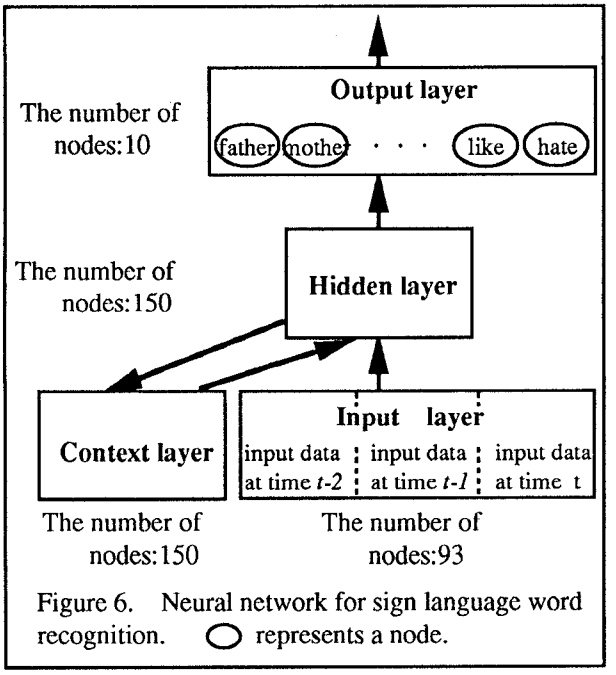

Figure 6. Neural network for sign language word recognition. ○ represents a node.

3.3. Detection of the delimiter between words

Segmentation of a sign language word is useful in gesture recognition because there may be many meaningless movements between individual words. Methods for checking the starting and ending points of a gesture are implemented in the processing. This enables sign language words to be recognized continuously and automatically without specifying delimiters.

To check the starting point of the sampling data, we assume that each gesture always starts from a specific posture. This assumption allows the system to detect the starting of the data sampling. As described in Section 2.4, the posture recognition system can automatically recognize the starting and ending of the data sampling, so we can use our posture recognition system as a starting point detector. We integrated these two neural networks as shown in Figure 7. The first network always receives the raw data from the Data Glove and determines the starting time of sampling. If the raw data is considered to represent a gesture, it is sent to the next network for gesture recognition. The beginning posture of each sign language word is taught to this neural network. That is, when the starting point of a sign language word is detected, the sign language recognition process starts.

To check the ending point of the sampling data, we hold the history of the result at each sampling point as shown in Figure 8. The end of a sign language word is judged from this history. When valid results are presented continuously for a time, the system will assume that the gesture has been recognized, and will output the final result. When invalid results appear continuously for a time, the system outputs a message indicating that the gesture cannot be recognized.

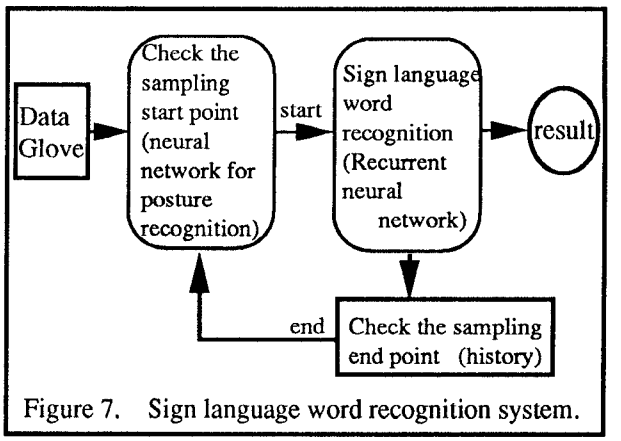

Figure 7. Sign language word recognition system.

4. EXPERIMENTS

Experiments on the recognition of ten sign language words were made using this recognition system. The learning time for ten words took 4 days with a SUN/4 workstation. The learning rate was 0.01 and the momentum was 0.05. The sampling rate of the Data glove is 30Hz. The ten sign language words used for testing are as follows:

(1) The pair of *skilled* - *unskilled*: These words use the same hand posture, but they differ in the direction of movement.

(2) The pair of *father* - *mother*: Each word is represented by two movements. They are the same for the first movement, but differ in the second.

(3) The pair of *memorize* - *forget*: These words use the same movement but in opposite directions.

(4) The pair of *brother* - *sister* and *like* - *hate*: These words use the same movement and the similar hand posture.

Figure 8 shows how the system recognizes the sign language word *mother* in (2). Note that no data is output when $t = 0$ and when $t = 1$, since the data at three sampling points of time are input to the sign language word recognition network. At first, there is little difference between the output value of the *father* node and *mother* node. This indicates that the system confuses *father* with *mother* in the beginning. As the mother-specific movement is recognized, the output value of the *mother* node increases quickly and the word *mother* is recognized.

	The character corresponding to the output node										The character corresponding to the node with greatest value
	father	**mother**	**brother**	**sister**	**memorize**	**forget**	**skilled**	**unskilled**	**like**	**hate**	
t = 0											
t = 1											
t = 2	0.29	0.27	0.00	0.00	0.00	0.04	0.04	0.00	0.23	0.02	**father**
t = 3	0.10	0.72	0.00	0.00	0.00	0.05	0.00	0.00	0.02	0.01	**mother**
t = 4	0.04	0.92	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	**mother**
t = 5	0.03	0.95	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	**mother**
t = 6	0.02	0.96	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	**mother**
t = 7	0.02	0.96	0.00	0.00	0.00	0.01	0.00	0.00	0.01	0.00	**mother**
t = 8	0.01	0.97	0.00	0.00	0.00	0.01	0.00	0.00	0.01	0.00	**mother**
t = 9	0.01	0.97	0.00	0.00	0.00	0.01	0.00	0.00	0.01	0.01	**mother**
t = 10	0.01	0.97	0.00	0.00	0.00	0.01	0.00	0.00	0.01	0.01	**mother**
t = 11	0.01	0.97	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	**mother**
t = 12	0.01	0.97	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**mother**
t = 13	0.00	0.97	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**mother**
											result = mother

Figure 8. Recognizing the sign language word for **mother**.

We evaluated how recognition rate has been improved with the method described in Section 3.2. Experiments were made to evaluate private dictionaries for specific people. Each recognition rate is the mean value of the five measurements obtained under the same conditions as the learning mode. The recognition rate with those encoding methods was 96%, whereas it was only 80% without being augmented and filtered. The recognition rate has been improved by using the new encoding method.

Certain sign language word pairs such as *skilled-unskilled, father-mother,* and *memorize-forget,* are easy to be confused with each other. The method using recurrent neural network in our system can hold sufficient past history to successfully recognize these word pairs.

5. CONCLUSION

A neural network has been successfully applied to gesture recognition. We use a recurrent neural network for the gesture recognition system. The proposed automatic sampling method is very useful for a sign language words recognition system. Several improvements such as augmented and filtered data work effectively. We are conducting further to develop a method for handling a large vocabulary.

ACKNOWLEDGMENTS

We would like to thank Dr. Morita and Prof. Oomura for their advice and suggestions, and other members of the artificial reality group for their support and help.

REFERENCES

[1] F.P.Brooks,Jr, "Grasping Reality Through Illusion - Interactive Graphics Serving Science," *SIGCHI Proceeding*, May 1988, pp.1-11

[2] H.Ushiyama, K.Hirota, and K.Murakami, "Hand Gesture Interpretation using Neural Networks," *40th Information Processing Conference Proceedings* ,Vol.4, 1990, pp.152 (In Japanese)

[3] D.E.Rumelhart and J.L.McClelland, *Parallel Distributed Processing,* MIT Press,1986

[4] K.Maruyama, *Sign language hand book* ,Dynamic Sellers Publishing Co ,1981 (In Japanese)

[5] J.L.Elman, "Finding structure in time," *CRL Technical Report 8801* ,1988