A Space-Time Delay Neural Network for Motion Recognition and Its Application to Lipreading in Bimodal Speech Recognition

Student: Wen-Chieh Lin Advisor: Dr. Chin-Teng Lin

Institute of Control Engineering

National Chiao Tung University

ABSTRACT

The researches of motion recognition have received more and more attention in recent years due to the increasing need for computer vision in many domains, such as the surveillance system, multimodal human computer interface, and traffic control system. Most of the existing approaches separate the motion recognition task into the spatial feature extraction and time-domain recognition subtasks. However, since the information of motion resides in the space-time domain instead of the time domain or space domain independently, it seems more reasonable to fuse the feature extraction and classification in the space and time domains into a single framework. Based on this concept, we propose a Space-Time Delay Neural Network (STDNN) that can deal with the space-time dynamic information for motion recognition. The STDNN is an unified structure, in which the low-level spatiotemporal feature extraction and high-level space-time-domain recognition are fused. It possesses the spatiotemporal shift-invariant recognition ability that is inherited from the time delay neural network (TDNN) and space displacement neural network (SDNN), where the TDNN and SDNN are good at the temporal and spatial shift-invariant recognition, respectively. Unlike the multilayer perceptron (MLP), TDNN, and SDNN, the STDNN is constructed by the vector-type nodes and matrix-type links such that the spatiotemporal information can be gracefully represented in a neural network. Two experiments are done to evaluate the performance of the proposed STDNN. In the moving Arabic numerals (MAN) experiment, which simulates the object's free moving in the space-time domain on image sequences, the STDNN shows its good generalization ability on the spatiotemporal shift-invariant recognition. In the lipreading experiment, the STDNN recognizes the lip motions based on the inputs of real image sequences. It shows that the STDNN has better performance than the existing TDNN-based system, especially on the generalization ability. In addition to the lipreading application, the STDNN can be applied to other problems since no domain-dependent knowledge is used in the experiment.

Acknowledgements

I would like to thank my advisor, Dr. Chin-Teng Lin for his hearty guidance and great inspiration. I also wish to thank all members in the Information Media Laboratory in NCTU for their suggestion and encouragement. In particular, I would express my thanks to Hsi-Wen Nien and Hsin-Hui Liang who assist me in the implementation of bimodal speech recognition system. It a delightful experience to work with them.

Finally, I would thank to my parents for their endless love and support in my life.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Space and time coordinate the physical world that we are living. Everything we encounter exists at some space-time point. The things may be still or active, and the forms or behaviors of them may be different each time they present. In spite of these distortions, people have inherently marvel abilities to recognize them as well. To build an ideal recognizer capable of dealing with natural patterns in our daily life, such as speech, image, or motion, the recognizer should be insensitive to patterns' distortions in the time or space domain, or both of which.

Conventionally, there are some criterions for judging the recognizer's tolerance for distortions of input patterns. For example, the translation invariant property of a recognizer means that the recognizer can correctly recognize an object no matter where it is. We summarize these criterions for time-domain and space-domain distortions in Fig. 1.1. Some physical analogies between time-domain and space-domain criterions can be observed from the viewpoint of expansion of dimensions. For example, the shift-invariant criterion (in the time domain) corresponds to the translation-invariant criterion (in the space domain); the warping-invariant criterion (in the time domain) corresponds to the scaling- and deformation-invariant criterions (in the space domain).

There have been intensive researches on tackling these distortion problems either

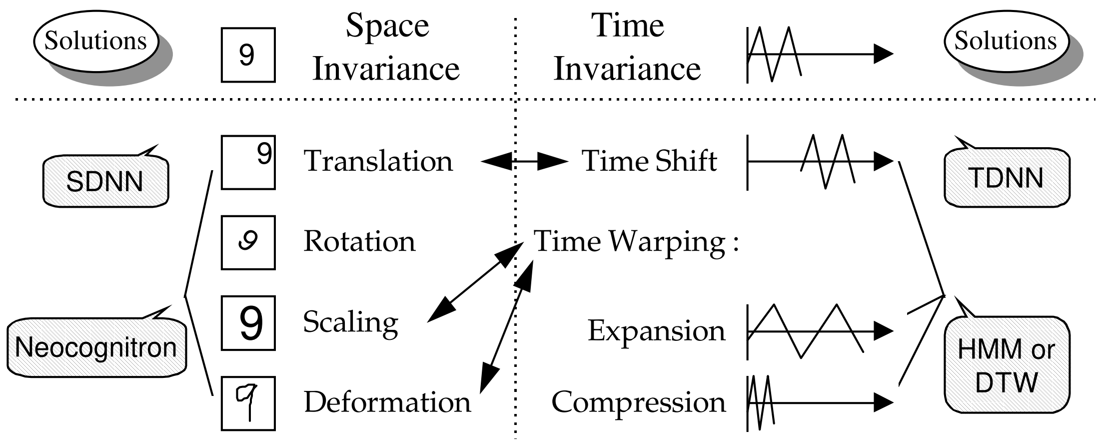


Figure 1.1: Invariant criterions for ideal recognizer.

on the time domain or space domain[1]. Typical examples are speech recognition and image recognition, which are on the time domain and space domain, respectively. The researchers solved the problems in two ways: one is to select invariant-features for these distortions, such as the FFT and moment features; the other is to endow the classifier with the invariant ability to these distortions. Some typical classifiers of them are shown in Fig. 1.1. As shown in the figure, in the space domain, there are Neocognitron [2] and Space Displacement Neural Network (SDNN), which are used for optical character recognition. The neocognitron overcomes the patterns' deformation and translation problem, while the SDNN solves the patterns' translation problem. The SDNN referred here indeed represents a class of neural networks [3][4][5]. In these neural networks, each node has a local receptive field which is a small rectangular window consisting of part of nodes in the previous layer, and weight sharing is applied to all nodes' receptive fields that are in the same layer. In the time domain, the typical classifiers capable of tackling the distortion problems are Time Delay Neural Network (TDNN) [1], Recurrent Neural Network (RNN), Hidden Markov Model (HMM), and Dynamic Time Warping (DTW) which

[1]The space domain referred here is indeed the 2-D image space domain.

are used for speech recognition. The TDNN overcomes the patterns' time-shift distortion, while the other three overcome the patterns' time-warping distortion.

However, there are only a few researches on the topic of integrated space-time-domain recognition, such as motion recognition. The dash line between the time domain and space domain in Fig. 1.1 is just like a watershed that separates the research fields of space-domain and time-domain pattern recognition in the past. Owing to the experience in developing a bimodal speech recognition system that incorporates the image recognition techniques into the conventional speech recognition system, we have the opportunity to acquire the rich sources from these two areas, and to bring the space-domain recognition and time-domain recognition into cooperation.

We would clarify the problems to be solved more clearly since the motion recognition is a diverse topic. First, we proceed the motion recognition on monocular images that are also mostly adopted in related researches. There are two reasons for using monocular images; human can recognize a motion only depending on an image sequence, so we believe that machine vision can also do the same thing; besides, in the monocular images, we need not take care of the problem of recovering 3-D geometry. Second, the input data are real image sequences. Some earlier researchers attach markers on the actor so that the analysis can be done more easily, but it is impractical in real situations. Third, we concentrate on the main part of recognition and the local tracking of the actor[2]. The global tracking is assumed to be supported by other systems. The motion types may be categorized to the non-stationary motion and stationary motion. An object's motion usually accompanies with a translation in position. Sometimes, the translation is much larger than our eyesight; for example, the motion of a flying bird includes a motion of wings and a

[2]The actor referred here means the area of interest in a motion. It may be a human, an animal, an object, or any others that perform the motion.

translation of body movement in the 3-D space. This is the case of nonstationary motion. On some other occasions, the translation is within our eyesight, such as facial expressions, hand gestures, or lipreading. This is the case of stationary motion. For nonstationary motion, global tracking to keep the actor within eyesight and local tracking to extract the actor from its background are needed. For stationary motion, only local tracking is needed.

Until recent years, the related topics about motion in computer vision focused predominately on the issues of geometry, either the three-dimensional geometry of a scene or the geometric motion of a moving camera [6]. One noted example, that is more related to the motion recognition, is the modeling of human motion, in which many researchers are working on recovering the three-dimensional geometry of moving people [7][8]. Besides, a specific area that has received some attention is the interpretation of moving light displays (MLDs) [8]. In the experiments of MLDs, some bright spots are attached to an actor dressed in black, and the actor is moving in front of a dark background. The results of MLDs are highly dependent on the ability to solve the correspondence problem and accurate tracking of joint and limb positions.

Nowadays, motion recognition has received more attentions. Yamato et al. [9] use HMM to recognize different tennis strokes. The image sequence is transformed into feature (mesh feature) vector sequence and then is vector quantized to a symbol sequence. The symbol sequences are used as input sequences for HMM in training and testing. The temporal invariance is accomplished by HMM, but spatial invariance is not fulfilled since the mesh feature is sensitive to the position-displacement as described in their experiments. Chiou and Hwang [10] also adopt HMM as the classifier in the lipreading task, but the features fed into HMM are extracted by a neural-network-based active contour model. Waibel et al. [11] develop a bimodal speech recognition system, in which a TDNN is used to perform lipreading, and downsampled images or 2D-FFT transformed images are used as features. In the last two cases of lipreading [10][12], the image sequences are color images and the color information is used in locating the mouth region. All of the systems treat the space-time-domain recognition problem as the time-domain recognition problem by transforming image sequences into feature vector sequences.

Polana and Nelson [13] put emphasis on the recognition of periodic motions. They separate the recognition task into two stages: detecting and recognizing. In the detecting stage, the motion of interest in an image sequence is tracked and extracted on the basis of the periodic nature of its signatures. The period of the object's motion is measured using a Fourier transform [14][15]. With the assumptions that the object producing periodic motion is moving along a locally linear trajectory and the height of the object does not change over time, it achieves translation and scale invariance in the space domain. In the recognizing stage, a template matching against spatiotemporal template of motion features is used. Temporal scale invariance is achieved by motion features, and shift invariance is achieved by template matching at all possible shifts. Generally speaking, Polana and Nelson achieve most of the invariant criterions in the time and space domains for periodic motions. However, the assumption that the entire image sequence must consist of at least four cycles of the periodic motion may be unrealistic in some situations. In some human motions, such as running and walking which are examples used in their experiments, this assumption is suitable. In other cases, such as open-the-door action, sit-down action, facial expressions, hand gestures, or lipreading, this assumption is unrealistic.

Most of the existing approaches separate the motion recognition task into spatial feature extraction[3] and time-domain recognition; i.e., they treat an image sequence as a feature vector sequence by feature extraction. By spatial feature extraction, the information in each image can be highly condensed into a feature vector, and the time-domain recognition can be done afterward. However, the spatial feature extraction is usually domain-dependent, and some delicate preprocessing tricks may be needed in particular. The developed recognition system is consequently restricted to one application since it is time-consuming to redesign the feature extraction unit for different applications. On the other hand, for motion recognition, the information dose not only exist in the space domain or time domain separately, but in the integrated space-time domain. It seems unreasonable to distinguish the feature extraction and recognition in the space domain and time domain separately. Therefore, we fuse the feature extraction and classification together in a single framework for the space-time-domain recognition in this paper.

We adopt the neural networks to develop our system since its powerful learning ability and flexibility have been demonstrated in many applications. To achieve our goal, we need a model that can treat the space-time 3-D dynamic information; i.e., the model should have the ability to learn the 3-D dynamic mapping that maps the input pattern varying in the space-time 3-D domain to a desired class. However, we find that the conventional neural network structure cannot treat this kind of problems well. The earlier MLP can learn the nonlinear static mapping between the input set and the output set. The inventions of the TDNN and RNN bring the neural network's applications into the spatiotemporal domain, in which the 1-D dynamic mapping between the input set and the output set can be learned. The SDNN, which is evolved from the TDNN, further makes the neural networks have the ability to learn the 2-D dynamic mapping. The related development, however, is

[3] The feature extraction we referred here includes all processes needed to transform an image to a feature vector.

stopped after the SDNN. Though one reason is that the need of such models is not apparent in the past, the main reason is that the ordinary constituents, which are used in the MLP, TDNN, and SDNN, are difficult to construct the complex network that can represent and learn the 3-D, or higher dimensional dynamic information gracefully.

In this paper, we propose a Space-Time Delay Neural Network (STDNN) that embodies the space-time-domain recognition and the spatiotemporal feature extraction in a single neural network. The STDNN is a multilayer feedforward neural network constructed by vector-type nodes (cell) and matrix-type links (synapse). The synapses between layers are locally connected, and the cells in the same layer use the same set of synapses (weight sharing). The STDNN's input is the real image sequence and its output is the recognition result. By building every layer into a spatiotemporal cuboid, the STDNN preserves the inherent spatiotemporal relation in motion. The size of the cuboids are shrunk more and more from the input layer to the final output so that the information is condensed from the real image sequence to the recognition result. For the training, due to the novel structure of the STDNN, two new supervised learning algorithms are derived based on the backpropagation rule.

The STDNN possesses the shift-invariant property in the space-time domain because it inherits the TDNN's shift-invariant property and the SDNN's translation-invariant property. The STDNN's spatiotemporal shift-invariance ability means that the accurate tracking in the space-time domain is not needed. Once the whole motion occurs in the image sequence, the STDNN can handle it. The actor need neither locate in the centroid nor start his/her motion in the beginning of the image sequence.

This paper is organized as follows. Chapter 2 and Chapter 3 describe the

STDNN's structure and learning algorithms, respectively. Two recognition experiments, moving Arabic numerals (MAN) and lipreading, are illustrated in Chapter 4. The former exhibits the STDNN's spatiotemporal shift-invariant property, while the later shows the practical application of the STDNN. Finally, we give the conclusion in Chapter 5.

Chapter 2

Space-Time Delay Neural Network (STDNN)

The motion recognition problem involves the information processing of time, space, and features. This high-dimensional information, however, is not easy to be represented explicitly by the ordinary neural networks. Because the basic units and interconnections used in these neural networks are often treated as scalar individuals. To visualize the high-dimensional information concretely, the basic units and interconnections should not be scalar-type nodes and links.

The proposed STDNN employs a vector-type node and matrix-type link as the basic unit and interconnection, respectively. The vector-type node, denoted as cell, embodies several hidden nodes which presents the activation of different features in a specific spatiotemporal region. The matrix-type link, denoted as synapse, fully connects the hidden nodes in any two cells it connects. In this manner, the feature information is concealed in cells, so the time and space relation can be represented as a 3-D box — the way we use to visualize the time and space information. Figure 2.1 shows the cell and synapse, and the hidden nodes and interconnections inside them.

A three-layer STDNN is depicted in Fig. 2.2. In the input layer, an image sequence is lined up along the t-axis, and each pixel in an image is viewed as a cell in the input layer. The cell in the input layer generally contains one node in this

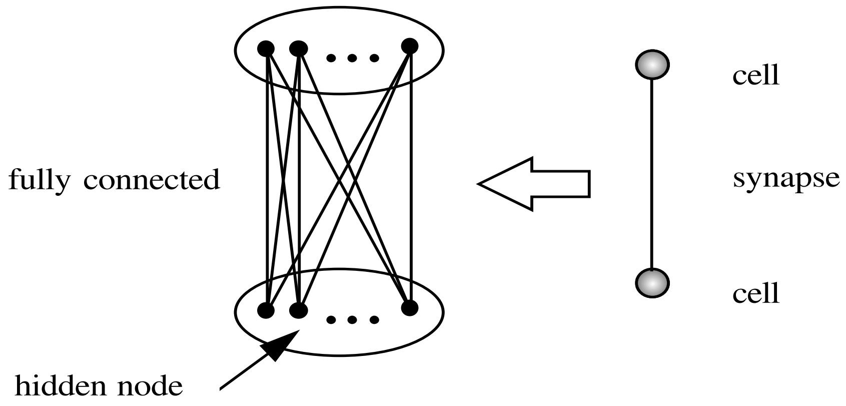


Figure 2.1: The cells and synapse in the STDNN.

case, but in other cases, where the elements of input data may be a vector, it could contain as many nodes as the length of the vector. Based on the contextual relation of time and space, the cells arranged along t-axis, y-axis, and x-axis forms a 3-D cuboid in each layer. The cells in the hidden layer generally contain multiple hidden nodes so that there is enough dimension space to reserve the feature information. In the output layer, the number of hidden nodes in the cell equals to the number of classes to be recognized. For example, if there are four classes of inputs to be classified, the number of hidden nodes should be four. As shown in Fig. 2.2, the information is concentrated layer by layer, and the final output is obtained by summing up the outputs of cells in the output layer and taking average. The last stage of the averaging is designed to acquire the shift-invariant ability because every cell in the output layer plays an equal part in the final decision.

To achieve the shifting invariant ability in both the space and time domains, each cell has a locally-linked receptive field that is a smaller spatiotemporal box consisting of cells in the previous layer. Moreover, all cells in the same layer use the same set of synapses to calculate the weighted sum of the net inputs from the activation values of cells covered in the receptive box. The net input of the cell Z

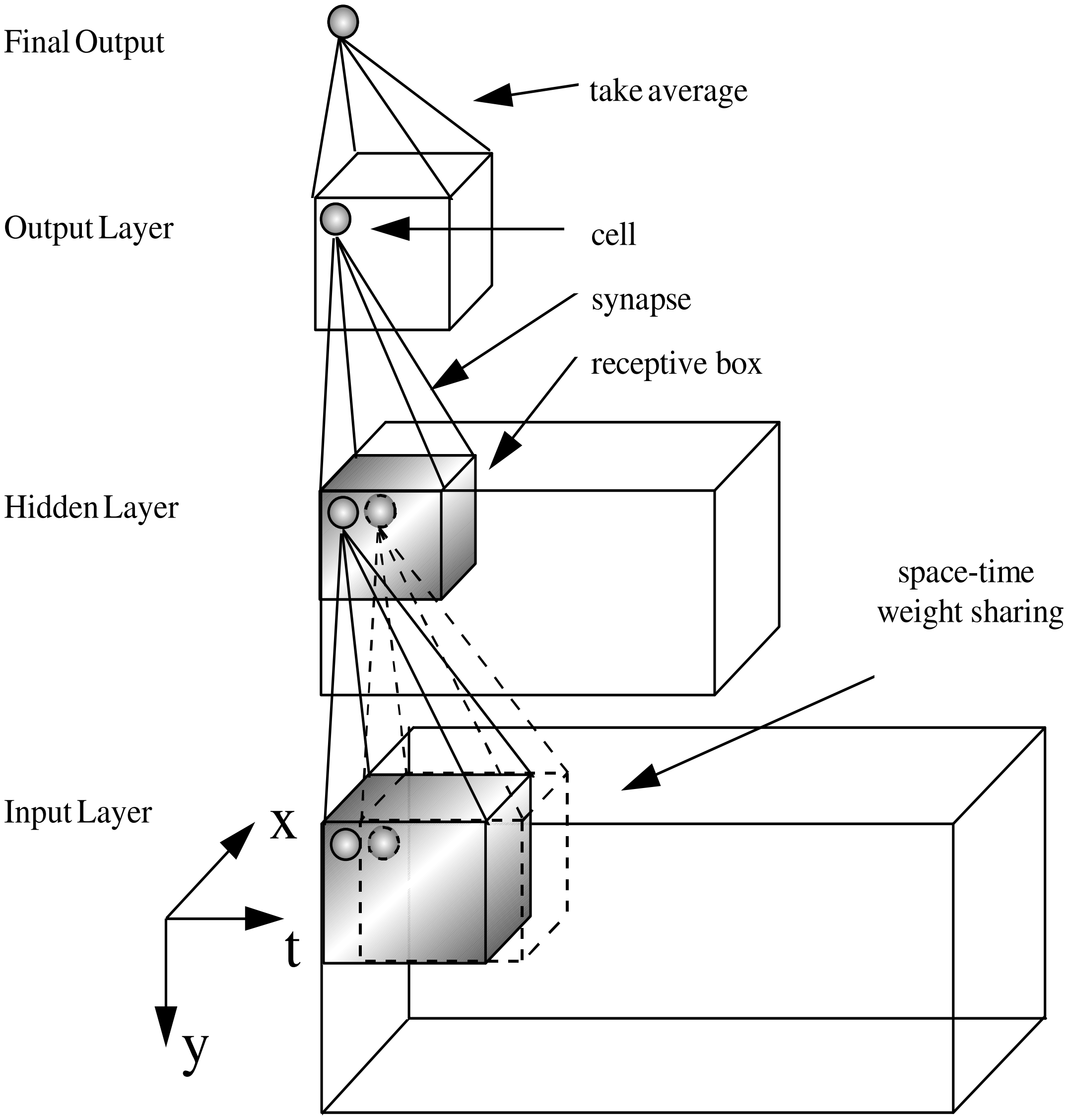


Figure 2.2: A three-layer STDNN.

at layer l with location (q_t, q_y, q_x) is expressed by

$$\boldsymbol{NetZ}(q_t, q_y, q_x) = \sum_{i_t=0}^{I_t-1} \sum_{i_y=0}^{I_y-1} \sum_{i_x=0}^{I_x-1} \boldsymbol{W}(i_t, i_y, i_x)\boldsymbol{X}(q_t\alpha_t + i_t, q_y\alpha_y + i_y, q_x\alpha_x + i_x) + \boldsymbol{B}(q_t, q_y, q_x), \tag{2.1}$$

where $\boldsymbol{NetZ}(\cdot) \in \Re^{H^l \times 1}$ is the net input of cell Z, $\boldsymbol{X}(\cdot) \in \Re^{H^{l-1} \times 1}$ are the cell output values in the $(l-1)$th layer, $\boldsymbol{B}(\cdot) \in \Re^{H^l \times 1}$ is the bias cell, and $\boldsymbol{W}(\cdot) \in \Re^{H^l \times H^{l-1}}$ are the synapses in the lth layer. The output of cell Z is

$$\boldsymbol{Z}(q_t, q_y, q_x) = a\left(\boldsymbol{NetZ}(q_t, q_y, q_x)\right), \tag{2.2}$$

where $\boldsymbol{Z}(\cdot) \in \Re^{H^l \times 1}$, and $a : \Re^{H^l} \to \Re^{H^l}$ is the activation function. The indexes used in Eq. (2.1) and Eq. (2.2) are defined as follows.

$(q_t, q_y, q_x) \equiv$ the location of cell Z in layer l.

$\alpha_t, \alpha_y, \alpha_x \equiv$ the step size for the receptive box moving along t-axis, y-axis, and x-axis, respectively, at each time step.

$(q_t\alpha_t, q_y\alpha_y, q_x\alpha_x) \equiv$ the origin of the receptive box.

$(q_t\alpha_t + i_t, q_y\alpha_y + i_y, q_x\alpha_x + i_x) \equiv$ the location of cell X in layer $l-1$.

$(i_t, i_y, i_x) \equiv$ the space-time delay index of the synapse W.

$I_t, I_y, I_x \equiv$ the size of receptive box along t-axis, y-axis, and x-axis, respectively.

The practical view of Eqs. (2.1) and (2.2) is drawn in Fig. 2.3. When the cell Z is locating at (q_t, q_y, q_x), the origin of the receptive box is set at $(q_t\alpha_t, q_y\alpha_y, q_x\alpha_x)$. Relative to this origin, the cell X with local coordinate (i_t, i_y, i_x) is fed to the cell Z, where (i_t, i_y, i_x) ranges from $(0, 0, 0)$ to $(I_t - 1, I_y - 1, I_x - 1)$. Since the set of synapses is identical in the same layer, the index of synapses are local coordinates

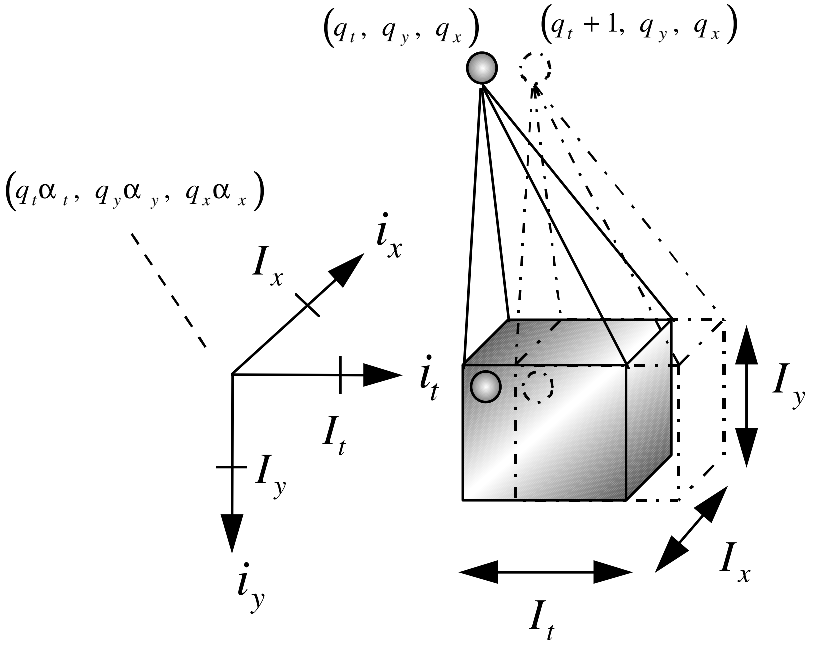


Figure 2.3: Practical view of Eqs. (2.1) and (2.2).

that are only relative to the origin of the receptive box. The same index is used for the synapses which have the same relative positions in different receptive boxes.

To make mathematic symbols clear and easy to understand, the notations of 3-D indexes are redefined as follows

$$\begin{aligned}
\boldsymbol{q} &\equiv (q_t, q_y, q_x), \\
\boldsymbol{i} &\equiv (i_t, i_y, i_x), \\
\boldsymbol{I} &\equiv (I_t, I_y, I_x), \\
\boldsymbol{\alpha} &\equiv (\alpha_t, \alpha_y, \alpha_x), \qquad (2.3)
\end{aligned}$$

and Eq. (2.1) and Eq. (2.2) can be rewritten as

$$\begin{aligned}
\boldsymbol{NetZ(q)} &= \sum_{\boldsymbol{i}=\boldsymbol{0}}^{\boldsymbol{I}-\boldsymbol{1}} \boldsymbol{W(i)X(q \circ \alpha + i) + B(q)}, \qquad (2.4) \\
\boldsymbol{Z(q)} &= \boldsymbol{a(NetZ(q))}, \qquad (2.5)
\end{aligned}$$

where $\circ$ is defined as the one-by-one array multiplication, $\boldsymbol{q} \circ \boldsymbol{\alpha} = (q_t\alpha_t, q_y\alpha_y, q_x\alpha_x)$. This operator will be used frequently in this paper.

The operation of the STDNN can be viewed in another way. The receptive box travels in a spatiotemporal hyperspace and reports its findings to the correspondent cell in the next layer whenever it goes to a new place. When the searching in the present layer is complete, the contents of all cells in the next layer are filled, and the searching in the next layer starts. In this way, the locally spatiotemporal features in every receptive box are extracted and fused layer by layer till the final output is generated. In other words, the STDNN gathers the locally spatiotemporal features that appear at any regions in the hyperspace to make the final decision. Hence, the STDNN possesses the shift-invariant ability in both the time and space domains.

Chapter 3

Learning Algorithms for the STDNN

The training of the STDNN is based on supervised learning, and the gradient descent method is used to derive the weight updating rules. Since the STDNN is evolved from the TDNN, the derivation of the learning algorithms of STDNN can acquire much inspiration from that of TDNN. The TDNN topology, however, is in fact embodied in a broader class of neural networks in which each synapse is represented by a finite-duration impulse response (FIR) filter. The latter neural network is referred to as the FIR multilayer perceptron (MLP). There are many discussions about the training of the FIR MLP. Due to the difference in the way the gradients are computed and the error function is defined, many different forms of training algorithms for the FIR MLPs have been derived [16][17][18][19].

Unfortunately, these training algorithms for the FIR MLP cannot be applied directly to the training of the STDNN because of the different constituents of the STDNN and FIR MLP. In the FIR MLP, the synapse is treated as an FIR filter in which every coefficient represents a weight value on a specific delay link. Unlike the STDNN, the input node of each FIR filter is a scalar node; i.e., the feature information is not submerged into a cell. The derivation of the training algorithms for the FIR MLP thus focuses on the adaptation of the FIR synapse rather than on that of the scalar synapse in ordinary MLPs.

We derive the training algorithms of the STDNN from a different viewpoint. Unlike the FIR MLP that hides the time-delay information into a FIR filter, the STDNN hides the feature information into a cell such that time-space information can be easily visualized. Consequently, the training algorithms for the STDNN are derived from the standpoint of the vector-type cell and the matrix-type synapse.

In the following sections, two learning algorithms are derived. The first one is derived from the intuitive way that is used firstly in the training of the TDNN [1]. In this method, a static equivalent network is constructed by unfolding the STDNN in time and space, and then the standard backpropagation algorithm is used for training. The second one adopts an instantaneous error function and accumulated gradients computation, which is somewhat like one of the algorithms proposed for the FIR MLP training [19]. These two learning algorithms will be referred to *static method*, and *instantaneous error method*, respectively.

3.1 Static Method for Training the STDNN

For clear explanation, a learning algorithm for the 1-D degenerative case of the STDNN will be discussed first. This degenerative network will be referred to the 1-D STDNN. As shown in Fig. 3.1, the 1-D STDNN is actually a TDNN but in which the hidden nodes originally lined up in each vertical axis are grouped in a cell.

A three-layer 1-D STDNN unfolded in time is depicted in Fig. 3.2. In this figure, the cell in the input layer is denoted as $\boldsymbol{X}(m) \in \Re^{H^{L-2}\times 1}$, where m is the timing index, and H^{L-2} is the number of the hidden nodes embodied in each cell. With the same symbology, the cell in the hidden layer and output layer can be represented by $\boldsymbol{Z}(q) \in \Re^{H^{L-1}\times 1}$ and $\boldsymbol{Y}(n) \in \Re^{H^{L}\times 1}$, respectively. The synapse $\boldsymbol{V}(n;j) \in \Re^{H^{L}\times H^{L-1}}$

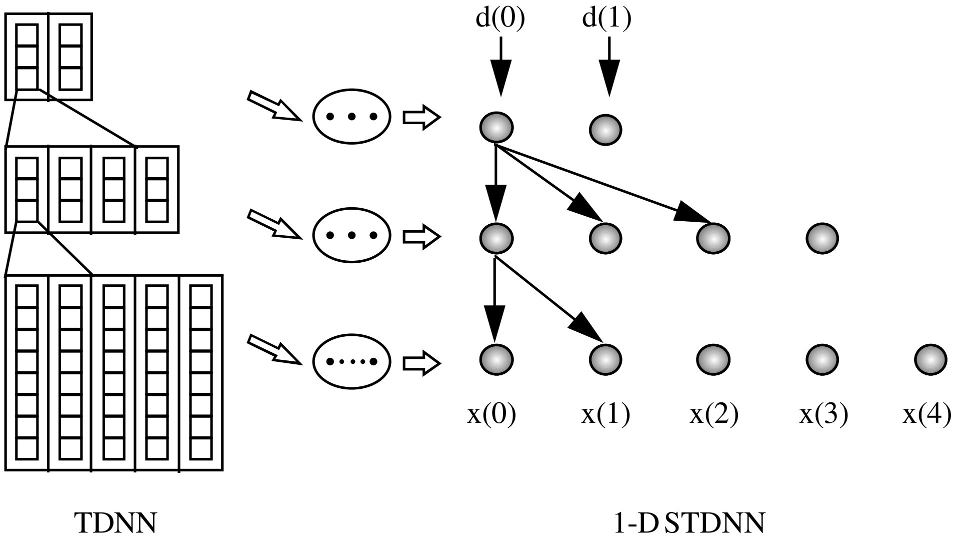


Figure 3.1: An 1-D STDNN and TDNN.

connects the cell $\boldsymbol{Y}(n)$ and $\boldsymbol{Z}(n\beta + j)$, where β is the offset each time the receptive field moves, and j is the number of unit-delays, ranging from 0 to $J-1$. The parameter J that is the total number of unit-delays can also be practically viewed as the length of the receptive field. As depicted in Fig. 3.2, when the cell Y moves from n to $n+1$, the receptive field jumps β cells, and the subsequent J cells make up the receptive field of $\boldsymbol{Y}(n+1)$.

In the unfolded 1-D STDNN, many synapses are replicated. To keep track of which static synapses are actually the same in the original (folded) network, the augmentive parameter n is introduced. The parameter n used in the unfolded network distinguishes the synapses which are the same ones in the 1-D STDNN. For example, the synapses $\boldsymbol{V}(n; j)$ and $\boldsymbol{V}(n+1; j)$ represent two different ones in the unfolded network, but they are identical in the 1-D STDNN. Similarly, the synapses between the hidden layer and the input layer are denoted as $\boldsymbol{W}(q; i) \in \Re^{H^{L-1} \times H^{L-2}}$ that connects the cell $\boldsymbol{Z}(q)$ and $\boldsymbol{X}(q\alpha + i)$, where α is the offset each time the receptive field moves, and i is the number of unit-delays.

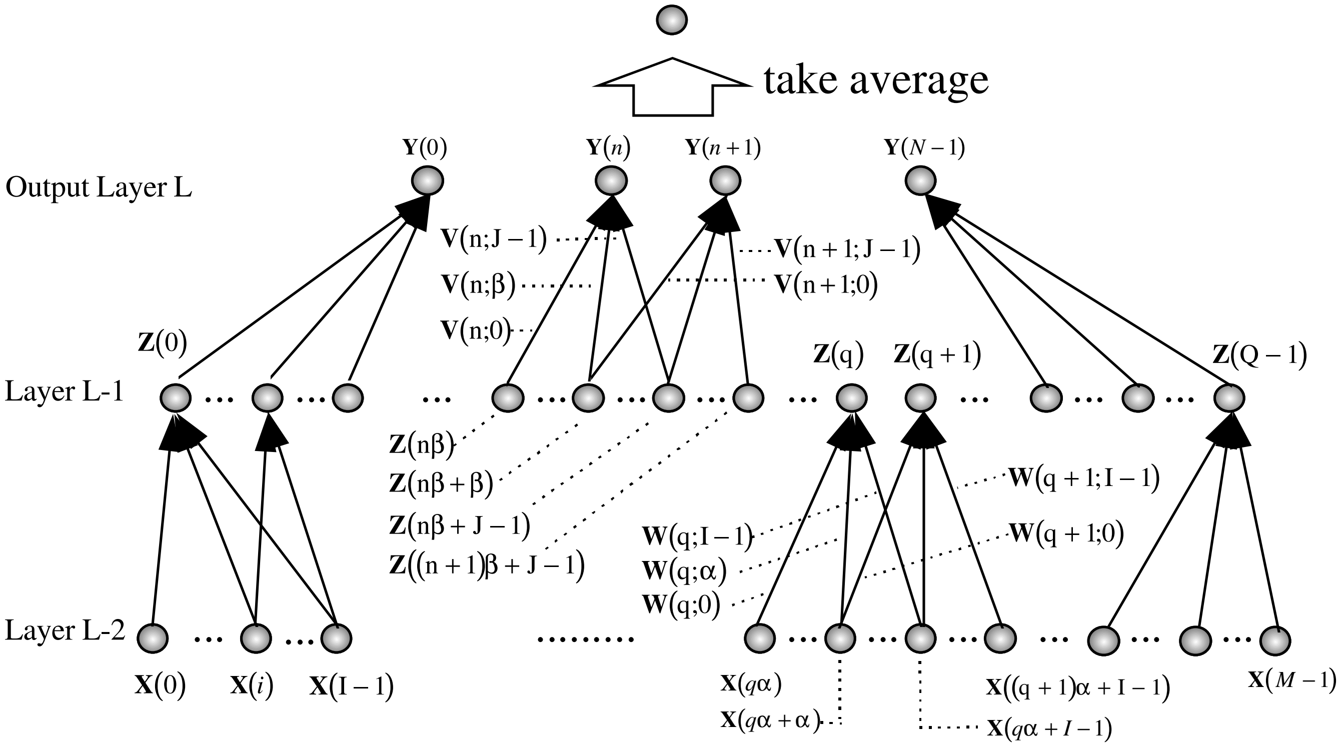


Figure 3.2: An 1-D STDNN unfolded in time.

According to Eq. (2.5), the output of these cells can be expressed by

$$\boldsymbol{Y}(n) = a\left(\sum_{j=0}^{J-1} \boldsymbol{V}(n;j)\boldsymbol{Z}(n\beta + j) + \boldsymbol{C}(n)\right), \qquad n = 0, \cdots, N-1, \tag{3.1}$$

$$\boldsymbol{Z}(q) = a\left(\sum_{i=0}^{I-1} \boldsymbol{W}(q;i)\boldsymbol{X}(q\alpha + i) + \boldsymbol{B}(q)\right), \qquad q = 0, \cdots, Q-1, \tag{3.2}$$

where $\boldsymbol{C}(n)$ and $\boldsymbol{B}(q)$ are the bias cells. The final output $\boldsymbol{O}$ is the average of $\boldsymbol{Y}(n)$ summing over n

$$\boldsymbol{O} = \frac{1}{N}\sum_{n=0}^{N-1} \boldsymbol{Y}(n). \tag{3.3}$$

Let $\boldsymbol{d}$ denote the desired output of the STDNN, then the square error function is defined by

$$E = \frac{1}{2}(\boldsymbol{d} - \boldsymbol{O})^{\boldsymbol{T}}(\boldsymbol{d} - \boldsymbol{O}). \tag{3.4}$$

Applying the gradient descent method to minimize the above error function, we have

$$\Delta\boldsymbol{V}(n;j) = -\eta\frac{\partial E}{\partial\boldsymbol{V}(n;j)}, \tag{3.5}$$

$$\Delta\boldsymbol{W}(q;i) = -\eta\frac{\partial E}{\partial\boldsymbol{W}(q;i)}. \tag{3.6}$$

Thus, the synapse updating rule of the output layer and hidden layer can be obtained by differentiating the error function of Eq. (3.4) with respect to the matrix $\boldsymbol{V}(n;j)$, and $\boldsymbol{W}(q;i)$. Only the resulting equations are listed here. The detailed derivation can be found in Appendix A.

Weight updating rule for the output layer

The weights in the output layer of the STDNN are updated by

$$\Delta\boldsymbol{V}(n;j) = \eta\Delta_{\boldsymbol{Y}(n)} \cdot \boldsymbol{Z}(n\beta+j)^T, \tag{3.7}$$

where η is the learning constant, and the error signal for cell $\boldsymbol{Y}(n)$ is defined as

$$\Delta_{\boldsymbol{Y}(n)} = \frac{1}{N}(\boldsymbol{d}-\boldsymbol{O}) \circ a'\left(\boldsymbol{netY}(n)\right), \tag{3.8}$$

where $\boldsymbol{netY}(n)$ is the net input of cell $\boldsymbol{Y}(n)$

$$\boldsymbol{netY}(n) = \sum_{j=0}^{J-1}\boldsymbol{V}(n;j)\boldsymbol{Z}(n\beta+j). \tag{3.9}$$

Weight updating rule for the hidden layer

The weights in the hidden layer of the STDNN are updated by

$$\Delta\boldsymbol{W}(q;i) = \eta\Delta_{\boldsymbol{Z}(q)} \cdot \boldsymbol{X}(q\alpha+i)^T, \tag{3.10}$$

where the error signal for cell $\boldsymbol{Z}(q)$ is defined as

$$\Delta_{\boldsymbol{Z}(q)} = \sum_{(n,j)\in\varphi}\boldsymbol{V}(n;j)^T \cdot \Delta_{\boldsymbol{Y}(n)} \circ a'\left(\boldsymbol{netZ}(q)\right), \tag{3.11}$$

where $\varphi = \{(n;j) \mid n\beta + j = q\}$ is the set of cells consisting of all fan-outs of $\boldsymbol{Z}(q)$, and $\boldsymbol{netZ}(q)$ is the net input of cell $\boldsymbol{Z}(q)$,

$$\boldsymbol{netZ}(q) = \sum_{i=0}^{I-1}\boldsymbol{W}(q;i)\boldsymbol{X}(q\alpha+i). \tag{3.12}$$

Finally, the weight changes are summed up and taken average to achieve weight sharing; i.e., the weight updating is performed until all error signals are backpropagated and all replicated weight changes are accumulated,

$$\Delta V(j) = \frac{1}{N} \sum_{n=0}^{N-1} \Delta V(n;j), \tag{3.13}$$

$$\Delta W(i) = \frac{1}{Q} \sum_{q=0}^{Q-1} \Delta W(q;i), \tag{3.14}$$

where Q and N are the numbers of replicated sets of synapses in the output layer and hidden layer, respectively.

For the tuning of the bias cells $\boldsymbol{C}(n)$ and $\boldsymbol{B}(q)$, the weight updating rules listed above can still be applied by setting the cell values as -1, and using a bias synapse connecting it to the output cells $\boldsymbol{Y}(n)$ and $\boldsymbol{Z}(q)$, respectively. In this way, the adjusting of a cell's bias values is accomplished by updating its bias synapse using the above weight updating rules.

The physical meaning of the above equations can be perceived by comparing them with those used in the standard backpropagation algorithm. These equations are merely in the same form as those used in the backpropagation algorithm, if we temporarily ignore the fact that each node here is a vector and the weight link is a matrix, and drop the transpose operators that maintains the legal multiplication of matrices and vectors.

The generalization of the 1-D STDNN to the 3-D case is easily accomplished by replacing the 1-D indexes with 3-D indexes. That is, we only need to change the timing indexes, the number of unit-delays, and offsets by Eq. (2.3).

3.2 Instantaneous Error Method for Training the STDNN

The instantaneous error method is derived by unfolding the STDNN in another way, which is originally used for the on-line adaptation of the FIR MLP. From the experience of the derivation of the first learning algorithm, we shall still start the derivation of the second one from the 1-D STDNN because of its clarity and easiness for generalization. Figure 3.3 illustrates the difference between these two unfolding methods. In Fig. 3.3(a), a three-layer 1-D STDNN is shown, in which the moving offset of the receptive field in each layer is one cell for each time. Its static equivalent networks unfolded by the first and second methods are shown in Fig. 3.3(b) and Fig 3.3(c), respectively. As shown in Fig. 3.3(c), many smaller subnetworks are replicated. The number of subnetworks is equal to the number of cells in the output layer of the 1-D STDNN. The instantaneous outputs are subsequently generated by these subnetworks, whenever there are enough input data coming. For example, as shown in Fig. 3.3(c), $\boldsymbol{Y}(0)$ is generated by the first subnetwork when the input sequence from $\boldsymbol{X}(0)$ to $\boldsymbol{X}(3)$ presents. As $\boldsymbol{X}(4)$ arrives at the next time step, the output $\boldsymbol{Y}(1)$ is generated by the second subnetwork according to the input sequence from $\boldsymbol{X}(1)$ to $\boldsymbol{X}(4)$.

In the on-line adaptation of the FIR MLP, the synapses of the subnetwork are adjusted at each time step; that is, every time when the output $\boldsymbol{Y}(n)$ produces, the synapses are immediately adjusted according to the desired output $\boldsymbol{d}(n)$. In the 1-D STDNN, however, the synapses are not adjusted immediately at each time step. On the contrary, the changes of synapses are accumulated until the outputs of all subnetworks are generated. Then the synapses are adjusted according to the average of the accumulated changes.

The difference between the structures of the STDNN and FIR MLP leads us to

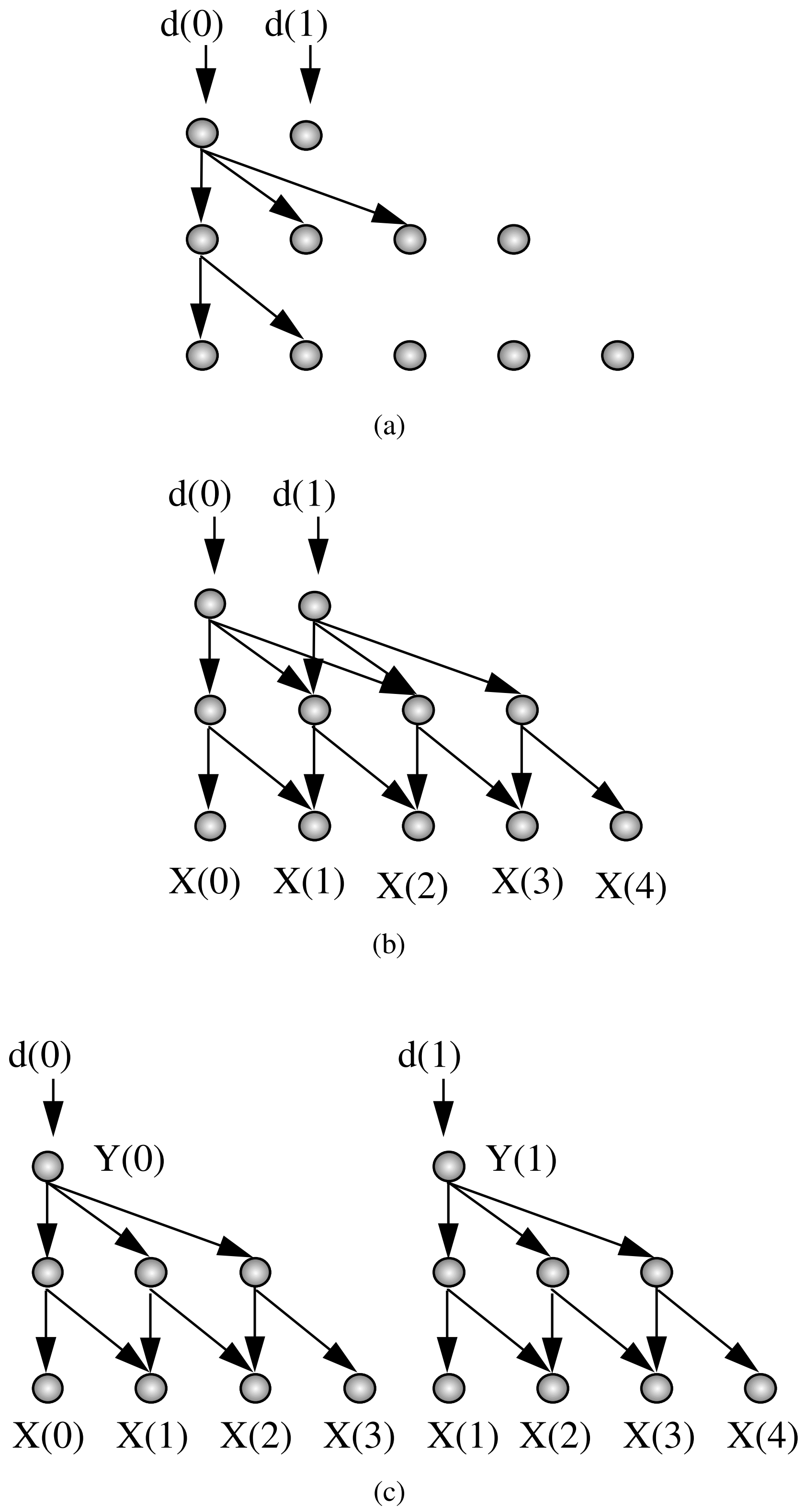


Figure 3.3: Illustration of different unfolding methods of 1-D STDNN.

adopt the accumulated-change manner to update synapses. Since the number of unfolded subnetworks of the STDNN is greater than that of the FIR MLP generally, the on-line adaptation on every subnetwork would take much longer time in the STDNN. Therefore, for the consideration of training speed, the changes of the synapses are accumulated, and the synapses are updated at once when the output at the last time step is generated.

The synapse updating rules can thus be described by the following equations. Although the listed equations are in the 1-D case, they can be generalized to the 3-D case by using Eq. (2.3). Given the desired output $\boldsymbol{d}(n)$, the instantaneous error function at time instant n is defined by

$$E(n) = (\boldsymbol{d}(n) - \boldsymbol{Y}(n))^T(\boldsymbol{d}(n) - \boldsymbol{Y}(n)). \tag{3.15}$$

Since each subnetwork remains a static network, the same derivation used in the static method can be applied again. The resulting synapse updating rules are obtained as follows.

Weight updating rule for the output layer of the nth subnetwork

The weights in the output layer of the nth subnetwork of the STDNN are updated by

$$\Delta\boldsymbol{V}(n;j) = \eta\Delta_{\boldsymbol{Y}(n)} \cdot \boldsymbol{Z}(n\beta + j)^T, \tag{3.16}$$

where the error signal for cell $\boldsymbol{Y}(n)$ is defined as

$$\Delta_{\boldsymbol{Y}(n)} = (\boldsymbol{d}(n) - \boldsymbol{Y}(n)) \circ a'\left(\boldsymbol{netY}(n)\right). \tag{3.17}$$

Weight updating rule for the hidden layer of the nth subnetwork

The weights in the hidden layer (e.g., the $(L-1)$th layer) of the nth subnetwork of the STDNN are updated by

$$\Delta\boldsymbol{W}(q;i) = \eta\Delta_{\boldsymbol{Z}(q)} \cdot \boldsymbol{X}(q\alpha + i)^T, \qquad q = 0 \cdots R^{L-1} - 1 \text{ and } i = 0 \cdots I^{L-1} - 1, \tag{3.18}$$

where the range of timing index q bounded by R^{L-1} is the number of replicated sets of synapses in each subnetwork, and the range of synapse index i bounded by I^{L-1} is the size of the receptive field at layer $L-1$. With the redefined range of index q, the error signal for cell $\boldsymbol{Z}(q)$ is the same as Eq. (3.11).

The number of replicated sets of synapses in each layer can be computed as follows. Let R^l denote the number of replicated sets of synapses in layer l, and I^{l-1} denote the length of the receptive field in layer $l-1$. Then we have

$$R^{l-1} = R^l \times I^{l-1}, \qquad R^L = 1 \text{ and } l = L, L-1, \cdots, 0. \tag{3.19}$$

For example, we have $N = 2$, $L = 2$, $R^2 = 1$, $I^1 = 3$, and $R^1 = 1 \cdot 3 = 3$ in Fig. 3.3(c).

Because the number of replicated sets of synapses is changed, the weight average equations, Eq. (3.13) and Eq. (3.14), are modified to be

$$\Delta V(j) = \frac{1}{N \cdot R^L} \sum_{n=0}^{N-1} \Delta V(n; j), \tag{3.20}$$

$$\Delta W(i) = \frac{1}{N \cdot R^{L-1}} \sum_{n=0}^{N-1} \sum_{q=0}^{R^{L-1}} \Delta W(q; i), \tag{3.21}$$

where N is the number of total subnetworks.

Chapter 4

Recognition Experiments

To evaluate the performance of the STDNN, two groups of experiments are set up. The first one is the Moving Arabic Numerals (MAN), by which the STDNN's spatiotemporal shift-invariant ability is tested. The other one is the lipreading of Chinese isolated words, in which the practical application of the STDNN is illustrated.

4.1 Moving Arabic Numerals

A good neural network should have enough generalization ability to deal with the conditions which have never learned before. In this experiment, we show the STDNN's generalization ability about input shifting in space and time using the minimum training data. Although the experimental patterns may be rather simpler than those in the real situation, the designed patterns provide a good criterion for testing the shift-invariant capability.

An object's action appearing in different spatiotemporal region is simulated by a man-made image sequence in this experiment. It occurs when the tracking system cannot acquire the object accurately such that the object is not located in the image centroid nor synchronized with the beginning of the image sequence. There are two reasons for using man-made image sequences. First, many motions can be produced in a short time, and thus the experiments can be repeated easily. Second, some types of motions are difficult or even impossible for a natural object to perform, but they can be easily performed by simulation.

As shown in Fig. 4.1, an image sequence consists of sixteen frames, eight of which contain Arabic numerals. The eight numerals represent the consecutive transition of a motion, and the motion may start at any frame under the constraint that the whole motion is complete in an image sequence. By changing the order of the numerals appearing in the sequence, three classes of motion are made. The three classes are all constituted by eight basic actions (numerals); the first one and the second one act in the reverse order, while the last one acts as the mixture of the other two. We can view these motions as watching a speaker pacing on the platform. In that situation, we easily concentrate our attention to his/her facial expression or hand gesture, but ignore the background he/she is situated in contrast.

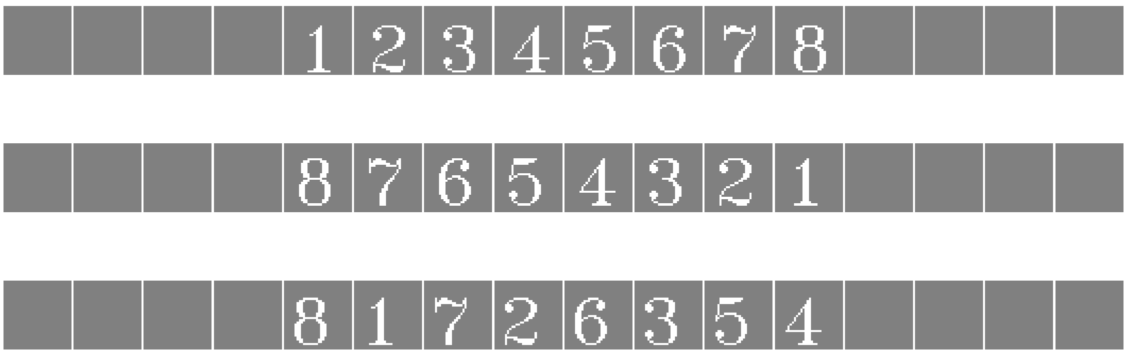


Figure 4.1: All training patterns used in the MAN experiment.

The experiment is proceeded as follows. At the training stage, one pattern for each class is chosen as the training pattern, in which the starting (x, y, t) coordinate is fixed at some point. All of the training patterns are shown in Fig. 4.1. After training, the patterns with numeral's (x, y, t) coordinates randomly moved are used to test the performance of the STDNN.

According to the degree of freedom that numerals are allowed to move in a frame, three sets of motions are tested. The numerals in the first set are moved in a block way; i.e., eight numerals are tied together, and the position of each of them is changed in the same manner. In the second set of motion, the numerals are moved in a slowvary way, in which only the 1st, 3rd, 5th, and 7th numerals are randomly moved, and the positions of the rest are bounded to the interpolation points of the randomly moved ones (e.g., the 2nd numeral is located at the middle point of the new positions of the 1st and 3rd numerals). In the last one, each numeral is freely moved in the frame independently. Basically, the first test set simulates the stationary motion, the second one simulates the nonstationary motion, and the last one simulates the highly nonstationary motion. Some patterns of these test sets are shown in Fig. 4.2, Fig. 4.3, and Fig. 4.4, respectively.

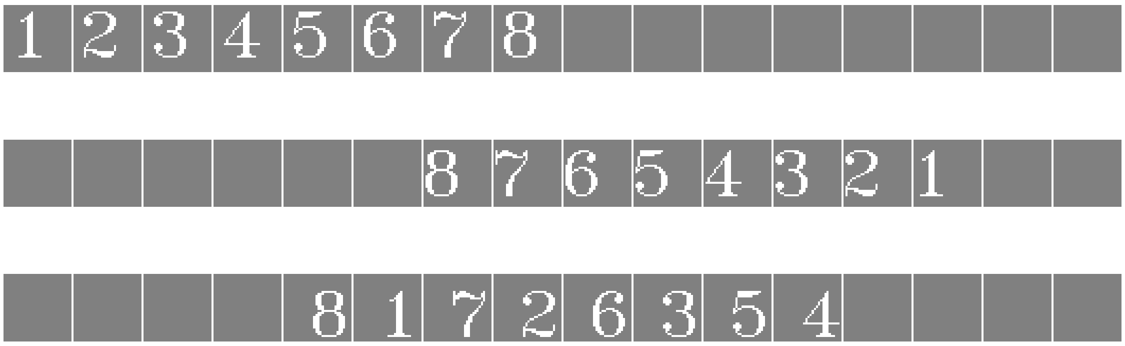


Figure 4.2: Some stationary motion patterns in the first test set.

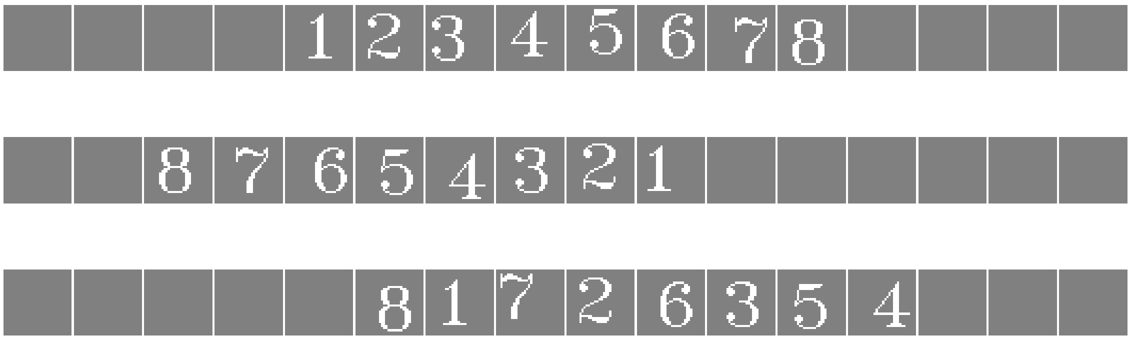


Figure 4.3: Some nonstationary motion patterns in the second test set.

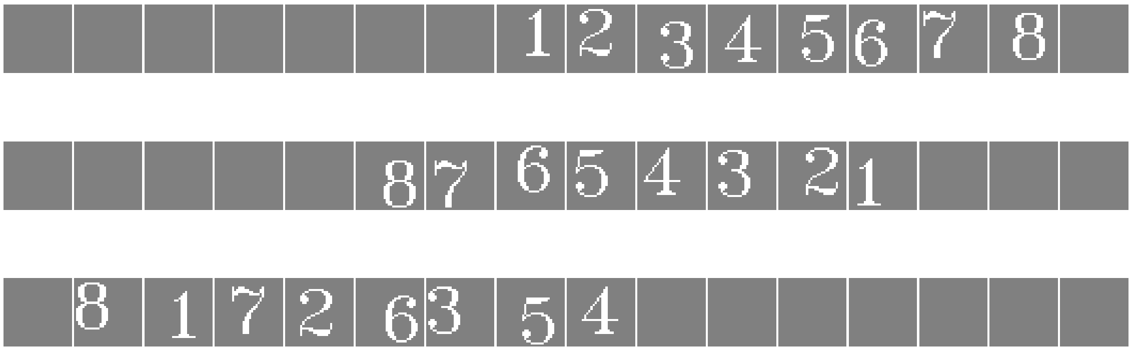


Figure 4.4: Some highly nonstationary motion patterns in the third test set.

The recognition rates of the STDNN on the three test sets, in which each class contains 30 patterns, are 85%, 80%, and 70%, respectively[1]. Several correctly-recognized patterns are shown in Figs. 4.2-4.4. The STDNN's configuration used in this experiment is listed in Table 4.1, in which H denotes the number of hidden nodes of the cell in each layer, (I_t, I_y, I_x) denotes the size of the receptive box, $(\alpha_t, \alpha_y, \alpha_x)$ denotes the moving offset of the receptive box, and (Q_t, Q_y, Q_x) denotes the numbers of the cells along t-axis, y-axis, and x-axis.

	Network Parameters			
	H	(I_t, I_y, I_x)	$(\alpha_t, \alpha_y, \alpha_x)$	(Q_t, Q_y, Q_x)
layer 0	1	(4,7,6)	(2,2,2)	(16,32,32)
layer 1	6	(3,5,4)	(1,2,2)	(7,13,14)
layer 2	6	(2,3,3)	(1,1,1)	(5,5,6)
layer 3	3			(4,3,4)

Table 4.1: The STDNN's network configuration in the MAN experiment.

Although the recognition rate in the MAN experiment is not very good, it is reasonable if we realize the disproportion between the hyperspace spanned by only three training patterns and that spanned by 90 test patterns. The variations in the test image sequences include translations in space and time, and changes of numerals'

[1] Although the image sequences shown are bi-level images, they are normalized to [-1, 1] before enter the STDNN. That is, they are treated as real numbers in the STDNN.

order, while in the training patterns, we only have a serial of numerals fixed at a specific time and space. The recognition rate can be improved by increasing the number of training patterns to cover the input space, but this is not the case we desire. Considering the real situation such as the example of the pacing speaker mentioned previously, it is impossible to gather so many training patterns covering the whole hyperspace the speaker may pass through at all time. Therefore, all we could do is to utilize the limited training data to earn the maximum recognition rate; i.e., enlarge the generalization ability of the neural network as much as possible.

On the other hand, the variations of the test patterns can be narrowed a little in some realistic conditions. Every numeral in the last test set is freely moved, but in some situations the object might not move so fast. Especially, the human motion in the consecutive image sequence is smoothly moving as usual. Therefore, the second set of test patterns, in which the moving of numerals is slowvary, is closest to the real case.

4.2 Lipreading

The STDNN's application in lipreading is illustrated in this experiment. In particular, we shall compare the performance of the STDNN with that of the lipreading recognition system proposed by Waibel et al. [11][12]. The Waibel's system is actually a bimodal speech recognition system in which two TDNNs are used to perform speech recognition and lipreading. In their lipreading system, two types of features, down-sampled images and 2D-FFT transformed images, are used. There are two reasons for us to compare the STDNN with the Waibel's system. First, both of the STDNN and TDNN are neural-network-based classifiers. Second, the TDNN has the time-shift invariant ability and the 2D-FFT features are spatial-shift invariant, both of which are the shift-invariant criterions of the STDNN.

The STDNN can be viewed as a classifier or as a whole recognition system (i.e., a classifier along with a feature extraction subsystem) depending on the type of input data. In the following experiments, we shall compare the performance of the STDNN with that of the Waibel's lipreading system based on the standpoints of the classifier or the whole recognition system. Besides, the comparison of two learning algorithms for the STDNN is also discussed here. Most of the researchers on the FIR MLP learning algorithms compare the performance of their algorithms using simple prediction problems. We try to do such comparisons on the real lipreading application, which is a more complex case.

4.2.1 Experiment Conditions

The motion of lip movements is recorded by a bimodal recognition system we have developed[2]. In recording, the speaker pronounces an isolated-word in front of the CCD camera that grabs the image sequence at the sampling rate of 66 ms per frame. The recording environment is under a well-lighting condition and the speaker is sitted before the camera at a proper distance. There are 16 frames in an image sequence, and the size of each frame is 256×256. The image sequence is then automatically preprocessed by an area of interest (AOI) extraction unit, in which the lip region is extracted from the speaker's face. For the input data in the down-sampled image type, the extracted images are rescaled to the size of 32×32. On the other hand, for the input data in the 2D-FFT type, the extracted images are rescaled to the size of 64×64, and 13×13 FFT coefficients in low-frequency area are selected from the magnitude spectrum of the 64×64 2D-FFT transformed image.

One question may be raised is whether the spatial-shift invariant ability is needed if an AOI extraction unit is adopted. The answer is positive. Because the extracted

[2]For the details, see Appendix B

lip region is always not accurate, the recognition with spatial-shift invariant ability is still necessary. Moreover, the inherent nature of speech makes the lipreading suffer much more from the shift or scale distortions in the time domain than other kind of motions. Therefore, not only the spatial-shift invariant ability but also the time-shift invariant ability is needed in the lipreading task.

In this experiment, two Chinese vocabularies are used for lipreading. One contains Chinese digits from 1 to 6, and the other contains acoustically confusable words consisting of 19 Chinese words[3]. In the following, these two vocabularies are referred to *Chinese digits* and *confusable words*, respectively. Figure 4.5 shows the six image sequences of Chinese digits.

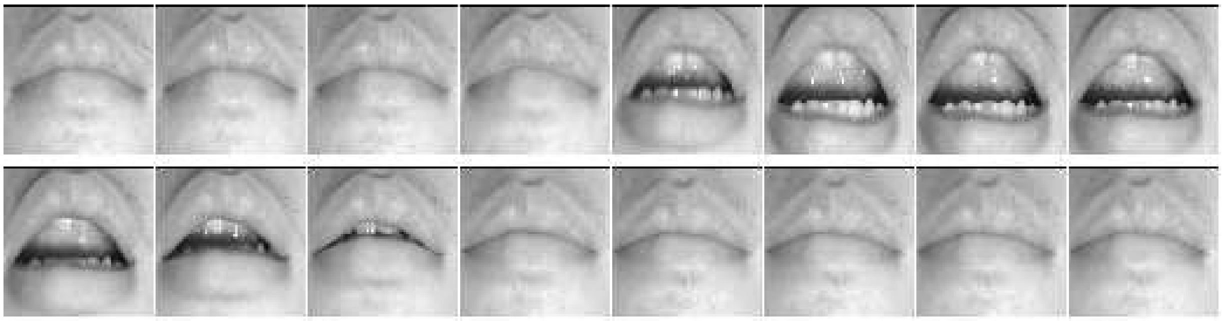
Following the same belief mentioned in the MAN experiment, we still try to use as few training patterns as possible to test the generalization ability of the TDNN and STDNN. In our experiment, we use 5 training patterns for each word. In the testing, two sets of patterns are used. One is recorded at the same time as in recording the training patterns, *but not used for training.* The other is recorded in another day. These two sets are referred to *test set*, and *new test set*, respectively. In the test set, there are 5 patterns for each word in Chinese digits and confusable words. In the new test set, there are 15 patterns for each word of the Chinese digits, and 10 patterns for each word of the confusable words. Figure 4.6 shows the example patterns of Chinese digits recorded at different time.

The purpose of using two test sets is to compare the performance of the systems under varying recording conditions. Acquired from the experience in developing an on-line bimodal speech recognition system[4], the training data recorded at the same time are often very unified such that the condition in the training set does not always

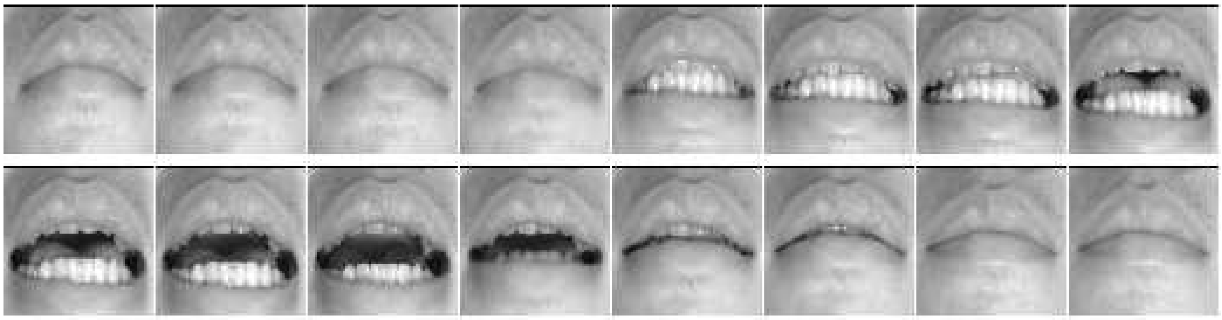
[3]Some words may be easily confusable acoustically, but they are distinct in lip movements. With the auxiliary of lipreading, the bimodal speech recognition system can improve the recognition rate of these words.

[4]For the details, see Appendix B

(a) Chinese digit: 1.

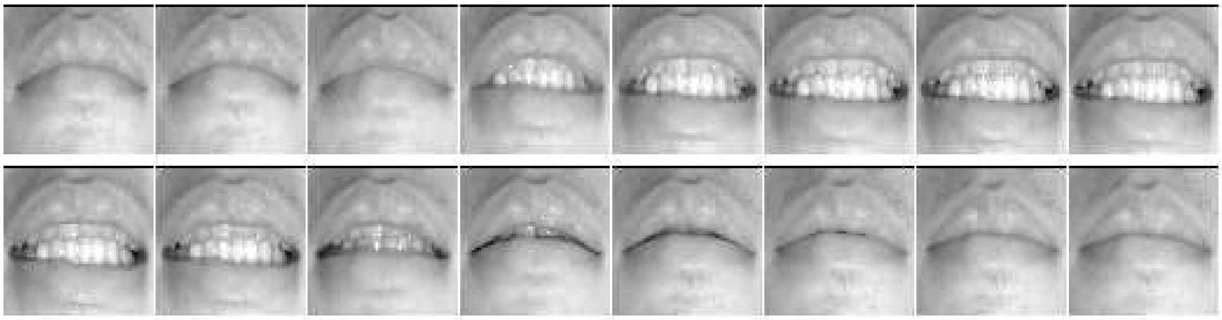


(b) Chinese digit: 2.

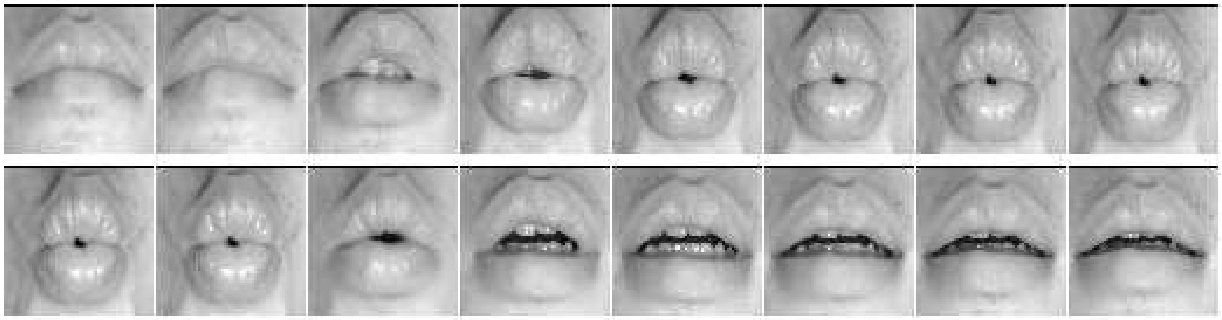


(c) Chinese digit: 3.

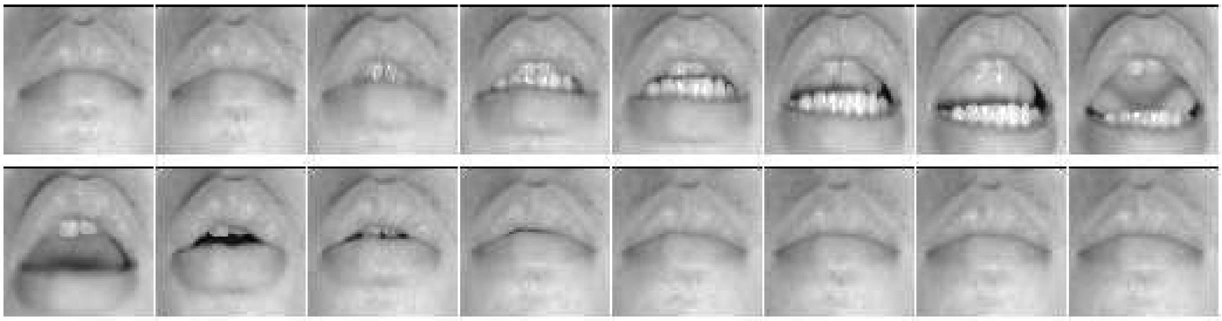
Figure 4.5: Chinese digits: 1 to 6.



(d) Chinese digit: 4.

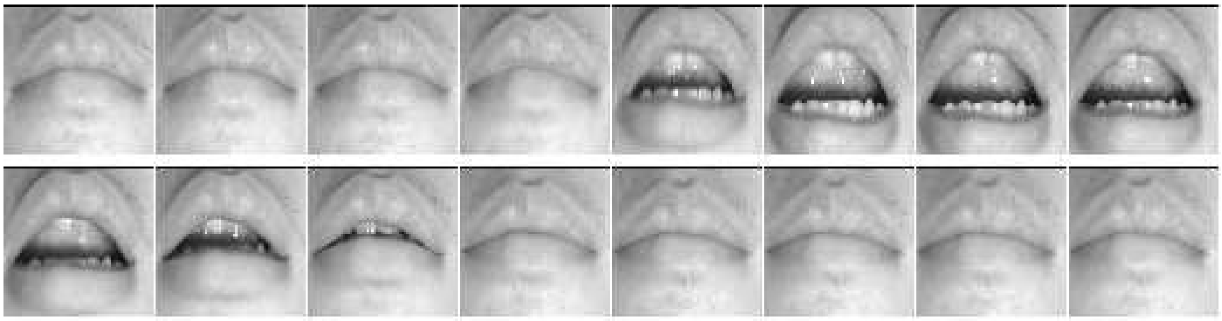


(e) Chinese digit: 5.

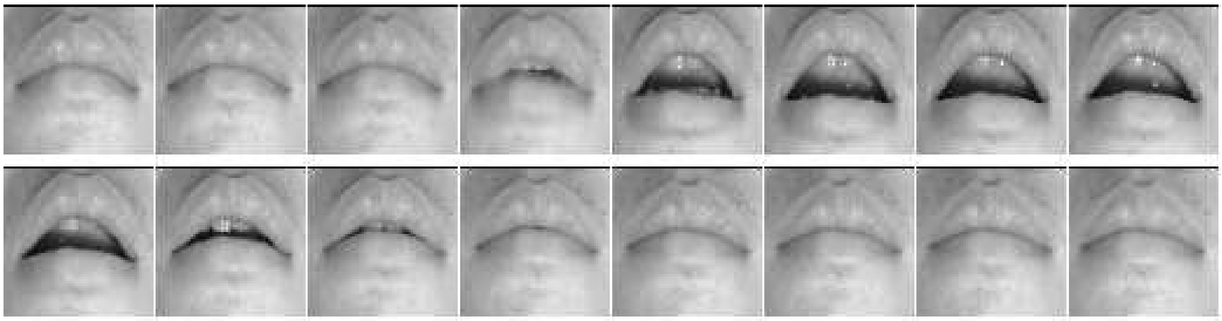


(f) Chinese digit: 6.

Figure 4.5: Chinese digits: 1 to 6 (continued).



(a) Pattern in the training set.



(b) Pattern in the test set.

(c) Pattern in the new test set.

Figure 4.6: Patterns of Chinese digits in the training set, test set, and new test set.

match the condition in on-line environment. Though the recording condition can be kept as unified as possible, it just cannot be the same at all time. One way to solve this problem is to enlarge the training set. However, the building of a large training set is often time-consuming. Another way to solve this problem, that is more practical and efficient, is to improve the learning ability of the classifier or select more robust features, or both.

4.2.2 Experiment Results

The network configurations used for Chinese digits and confusable words recognition are listed in Table 4.2 and Table 4.3, respectively. Based on the combination of different types of input data and neural networks, four networks are constructed for each vocabulary. For all experiments, the learning constant is 0.05, and the moment term is 0.5. The training process is stopped once the mean square error is less than 0.5. From Table 4.2 and Table 4.3, we observe that the TDNN is in fact a special

	Down-sampled							
	TDNN				STDNN			
	H	(I_t, I_y, I_x)	$(\alpha_t, \alpha_y, \alpha_x)$	(Q_t, Q_y, Q_x)	H	(I_t, I_y, I_x)	$(\alpha_t, \alpha_y, \alpha_x)$	(Q_t, Q_y, Q_x)
layer 0	1	(5,32,32)	(1,0,0)	(16,32,32)	1	(4,20,20)	(1,2,2)	(16,32,32)
layer 1	3	(5,1,1)	(1,0,0)	(12,1,1)	5	(7,4,4)	(1,1,1)	(13,7,7)
layer 2	6			(8,1,1)	6			(7,4,4)
	2D-FFT							
	TDNN				STDNN			
	H	(I_t, I_y, I_x)	$(\alpha_t, \alpha_y, \alpha_x)$	(Q_t, Q_y, Q_x)	H	(I_t, I_y, I_x)	$(\alpha_t, \alpha_y, \alpha_x)$	(Q_t, Q_y, Q_x)
layer 0	1	(7,13,13)	(1,0,0)	(16,13,13)	1	(4,9,9)	(1,1,1)	(16,13,13)
layer 1	7	(4,1,1)	(1,0,0)	(10,1,1)	5	(7,5,5)	(1,1,1)	(13,5,5)
layer 2	6			(7,1,1)	6			(7,1,1)

Table 4.2: Network configurations used for Chinese digit recognition.

case of the STDNN. The input layer of the TDNN is treated as 16 separate image frames, and the contextual relation in space is lost after the input data passing to the hidden layer.

For Chinese digits, the results of experiments for the networks trained by the static method and the instantaneous error method are listed in Tabel 4.4 and Ta-

	Down-sampled							
	TDNN				STDNN			
	H	(I_t, I_y, I_x)	$(\alpha_t, \alpha_y, \alpha_x)$	(Q_t, Q_y, Q_x)	H	(I_t, I_y, I_x)	$(\alpha_t, \alpha_y, \alpha_x)$	(Q_t, Q_y, Q_x)
layer 0	1	(6,32,32)	(1,0,0)	(16,32,32)	1	(5,10,15)	(1,2,2)	(16,32,32)
layer 1	5	(5,1,1)	(1,0,0)	(11,1,1)	2	(7,7,7)	(1,1,1)	(12,12,9)
layer 2	19			(7,1,1)	3	(6,6,3)	(1,1,1)	(6,6,3)
layer 3					19			(1,1,1)
	2D-FFT							
	TDNN				STDNN			
	H	(I_t, I_y, I_x)	$(\alpha_t, \alpha_y, \alpha_x)$	(Q_t, Q_y, Q_x)	H	(I_t, I_y, I_x)	$(\alpha_t, \alpha_y, \alpha_x)$	(Q_t, Q_y, Q_x)
layer 0	1	(7,13,13)	(1,0,0)	(16,13,13)	1	(4,9,9)	(1,1,1)	(16,13,13)
layer 1	10	(4,1,1)	(1,0,0)	(10,1,1)	3	(7,5,5)	(1,1,1)	(13,5,5)
layer 2	19			(7,1,1)	19			(7,1,1)

Table 4.3: Network configurations used for confusable-word recognition.

ble 4.5, respectively. For confusable words, the results are listed in Tabel 4.6 and Table 4.7.

According to the concept of pattern recognition, the performance of a recognition system is determined by the feature selection and classifier design. To isolate the effects caused by these two factors from each other, the experiment results are compared from two aspects: the classifier and the whole recognition system.

feature	Down-sampled		2D-FFT	
classifier	TDNN	STDNN	TDNN	STDNN
test set (30 patterns)	83.3%	100%	93.3%	93.3%
new test set (90 patterns)	62.2%	77.8%	61.1%	70%
Training epoch	245	2680	75	115

Table 4.4: Recognition rate of Chinese digits, trained by the static method.

First, from the viewpoint of classifiers, the STDNN is better than the TDNN according to the following observations. In this case, the STDNN and TDNN are compared based on the same input features. For the down-sampled image sequences, the STDNN's recognition rate is 15%∼30% higher than the TDNN's in Chinese digits, and 0%∼30% in confusable words. This result is not surprising because the TDNN only has the time-shift invariant recognition ability. It cannot overcome

feature	Down-sampled		2D-FFT	
classifier	TDNN	STDNN	TDNN	STDNN
test set (30 patterns)	83.3%	100%	93.3%	93.3%
new test set (90 patterns)	60%	90%	64.4%	75.6%
Training epoch	75	110	25	20

Table 4.5: Recognition rate of Chinese digits, trained by the instantaneous error method.

feature	Down-sampled		2D-FFT	
classifier	TDNN	STDNN	TDNN	STDNN
test set (95 patterns)	64.2%	72.6%	85.3%	87.4%
new test set (190 patterns)	14.7%	44.7%	28.9%	33.7%
Training epoch	570	50	200	225

Table 4.6: Recognition rate of confusable words, trained by the static method.

the possible spatial shift existed in the down-sampled images. On the contrary, if the input data are the 2D-FFT transformed images, the STDNN's improvement in recognition rate is not so much. Because of the space-shift invariant property of 2D-FFT, the TDNN would not greatly suffer the space-shift problem. Nevertheless, the recognition rate of the STDNN is still 5%~12% higher than that of the TDNN in all new test sets. This is due to the STDNN's high generalization ability that will be discussed later.

feature	Down-sampled		2D-FFT	
classifier	TDNN	STDNN	TDNN	STDNN
test set (95 patterns)	74.7%	74.7%	86.3%	88.4%
new test set (190 patterns)	14.2%	48.9%	27.4%	39.5%
Training epoch	215	20	55	55

Table 4.7: Recognition rate of confusable words, trained by the instantaneous error method.

Second, from the viewpoint of recognition systems, the STDNN performs better than the Waibel's system in most of experiments except the two experiments on the test sets of confusable words. In this case, the STDNN uses the down-sampled images as features, while the TDNN uses the 2D-FFT transformed images as features. For Chinese digits, the STDNN has higher recognition rate than the Waibel's system by 6.7% in the test set, and 16%~26% in the new test set. For confusable words, the STDNN has lower recognition rate than the Waibel's system by 13% in the test set. However, the recognition rate of the STDNN is 16%~21% higher than that of the Waibel's system in the new test set.

The performance comparison of the whole recognition system is what we concern much, since an unified structure of the motion recognition system, that embodies feature extraction unit and a classifier, is one of the STDNN's goals. The purpose of the performance comparison from the classifier viewpoint is in fact to isolate the effect of different features used in these systems. The experiments shows that the STDNN possesses better ability to extract more robust features, and to learn more information from the training data.

For other lipreading researches, the recognition rates vary a lot due to different specifications and experiment conditions of their systems. Besides, most of the lipreading systems are auxiliary to a speech recognition system, and thus the recognition rate of the whole system is the main concern. Petajan's speech recognition system [20] is the first documented system that contains lipreading subsystem. The experiment result shows that the bimodal speech recognition system achieves 78% recognition rate while the speech recognizer alone has only 65% recognition rate for a 100 isolated words, speaker-dependent system. In our available reports on the recognition rates of the lipreading systems, Pentland [21] achieves 70% recognition rate for English digits in a speaker-independent system; Goldschen [22] achieves

25% recognition rate for 150 sentences in a speaker-dependent system. Waibel et al. [12] achieve 30%~53% recognition rate for 62 phonemes or 42 visemes[5]. Generally speaking, the lipreading system alone doesn't work very well, but it doesn't matter if the goal of lipreading is auxiliary to speech recognition for achieving more robust recognition rate.

On the other hand, the comparisons of two different learning algorithms show that the instantaneous error method converges faster than the static method by threefold to fivefold. In particular, the training epochs are reduced from 2680 to 110 in the experiment of Chinese digits with down-sampled images[6]. Moreover, the network trained by the instantaneous error method has higher recognition rates than that trained by the static method in most of our experiments.

---

[5] One distinct characteristic of lipreading is that the number of distinguishable classes is not directly proportion to the number of words in speech. For example, /b/ and /p/ are distinguishable in speech, but they are difficult to be distinguished in lipreading. Therefore, some researchers define the visemes as the distinguishable classes by lipreading. The relation between viseme set and phoneme set is often an 1-to-n mapping.

[6] In fact, the motivation for deriving the instantaneous error method is partly due to the long training time taken in the static method.

Chapter 5

Conclusion

We have demonstrated the STDNN's space-time-domain recognition ability in this paper. In the MAN experiment, the STDNN shows the spatiotemporal shift-invariant ability learned from the minimum training data, while in the lipreading experiment, it shows higher recognition rate and better generalization ability than the TDNN-based system [12]. The STDNN has several advantages. First, it is a general approach to motion recognition since no a priori knowledge on the application domain is required. Second, because the feature extraction capacity is embedded, it needs very little preprocessing. The input data are almost the original image sequences except the downsampling preprocessing. Third, it possesses shift invariant ability both in the space and time domains such that the recourse for the tracking preprocess can be eliminated. Nevertheless, the STDNN still has some drawbacks. It takes much time to process data, especially in training. In training the STDNN on the downsampled image sequence, which is the worst case, it approximately takes one hour to run 6$\sim$8 epochs on Sparc-20[1]. The STDNN's processing time is dependent on its configurations. If the size and moving offset of the receptive box is small, it needs more time to search through the spatiotemporal cuboids. However, the cost in processing time gains the higher generalization ability. Therefore, there is a

[1]We have never preceded our experiments on the computer with vector processor, which is suitable for implementing the STDNN.

trade-off between the recognition rate and recognition speed. Another shortcoming is that the other invariant criterions in space and time such as rotation and scaling invariance have not been treated in this research yet.

To increase the processing speed, the STDNN's scale needs to be reduced, especially in the input layer. Because human beings do not see a moving pixel but a moving region, some unsupervised learning mechanism can be used to replace the input layer so that the redundancy can be reduced precedently. On the other hand, to treat other invariant criterions, the shape of the receptive box can be adapted in the learning stage so that the deformation in the space-time domain can be overcome.

The novel constituents of the STDNN in fact provide us an imagination about its potential applications. The STDNN suggests a neural network model capable of dealing with the 3-D dynamic information. It can classify the patterns described by the multidimensional information which are varying dynamically in a 3-D space. Besides, for the multidimensional signal processing, it can be treated as a nonlinear 3-D FIR filter. On the other hand, the cells and synapses provide us a tool to construct a more complex neural network that can treat higher-dimensional dynamic information.

Appendix A

Derivation Details of Learning Algorithms

We proceed the derivation by expanding Eqs. (3.1)-(3.4) to the scalar individuals, and differentiate these scalar equations by the chain rule. The results of these scalar individuals are then collected into vectors or matrices to form the equations given in Section 4. All of the denotations used here follow those used in Section 4 except that the numbers of hidden nodes in X, Y, and Z are redenoted as H^0, H^1, and H^2, which replace H^{L-2}, H^{L-1}, and H^L, respectively. The expansions of the cells referred in Eqs. (3.1)-(3.4) are

$$\boldsymbol{Y}(n) = \begin{bmatrix} Y_1(n) \\ \vdots \\ Y_{h^2}(n) \\ \vdots \\ Y_{H^2}(n) \end{bmatrix} \quad \boldsymbol{Z}(q) = \begin{bmatrix} Z_1(q) \\ \vdots \\ Z_{h^1}(q) \\ \vdots \\ Z_{H^1}(q) \end{bmatrix} \quad \boldsymbol{X}(m) = \begin{bmatrix} X_1(m) \\ \vdots \\ X_{h^0}(m) \\ \vdots \\ X_{H^0}(m) \end{bmatrix} \tag{A.1}$$

and the expansions of the synapses are

$$\boldsymbol{V}(n;j) = \begin{bmatrix} V_{11}(n;j) & \cdots & V_{1h^1}(n;j) & \cdots & V_{1H^1}(n;j) \\ \vdots & & \vdots & & \vdots \\ V_{h^21}(n;j) & \cdots & V_{h^2h^1}(n;j) & \cdots & V_{h^2H^1}(n;j) \\ \vdots & & \vdots & & \vdots \\ V_{H^21}(n;j) & \cdots & V_{H^2h^1}(n;j) & \cdots & V_{H^2H^1}(n;j) \end{bmatrix}, \tag{A.2}$$

$$
\boldsymbol{W}(q;i) = \begin{bmatrix} W_{11}(q;i) & \cdots & W_{1h^0}(q;i) & \cdots & W_{1H^0}(q;i) \\ \vdots & & \vdots & & \vdots \\ W_{h^1 1}(q;i) & \cdots & W_{h^1 h^0}(q;i) & \cdots & W_{h^1 H^0}(q;i) \\ \vdots & & \vdots & & \vdots \\ W_{H^1 1}(q;i) & \cdots & W_{H^1 h^0}(q;i) & \cdots & W_{H^1 H^0}(q;i) \end{bmatrix}. \tag{A.3}
$$

With these expansions, the scalar form of Eqs. (3.1)-(3.4) are written as follows

$$
\begin{aligned}
netY_{h^2}(n) &= \sum_j \sum_{h^1} V_{h^2 h^1}(n;j) Z_{h^1}(n\beta + j), && \text{(A.4)} \\
Y_{h^2}(n) &= a(netY_{h^2}(n)), && \text{(A.5)} \\
netZ_{h^1}(n) &= \sum_i \sum_{h^0} W_{h^1 h^0}(q;i) X_{h^0}(q\alpha + i), && \text{(A.6)} \\
Z_{h^1}(q) &= a(netZ_{h^1}(q)), && \text{(A.7)} \\
O_{h^2} &= \frac{1}{N} \sum_n Y_{h^2}(n), && \text{(A.8)} \\
E &= \frac{1}{2} \sum_{h^2} (d_{h^2} - O_{h^2})^2. && \text{(A.9)}
\end{aligned}
$$

From now on, we can apply the same procedures used in the derivation of the standard backpropagation algorithm to derive the updating rules for the STDNN.

For the output layer, we have

$$
\frac{\partial E}{\partial V_{h^2 h^1}(n;j)} = \frac{\partial E}{\partial O_{h^2}} \frac{\partial O_{h^2}}{\partial Y_{h^2}(n)} \frac{\partial Y_{h^2}(n)}{\partial netY_{h^2}(n)} \frac{\partial netY_{h^2}(n)}{\partial V_{h^2 h^1}(n)}. \tag{A.10}
$$

Let

$$
\begin{aligned}
\delta_{Y_{h^2}}(n) &\equiv -\frac{\partial E}{\partial O_{h^2}} \frac{\partial O_{h^2}}{\partial Y_{h^2}(n)} \frac{\partial Y_{h^2}(n)}{\partial netY_{h^2}(n)} \\
&= (d_{h^2} - O_{h^2}) \frac{1}{N} a'(netY_{h^2}(n)), && \text{(A.11)}
\end{aligned}
$$

and

$$
\frac{\partial netY_{h^2}(n)}{\partial V_{h^2 h^1}(n)} = Z_{h^1}(n\beta + j). \tag{A.12}
$$

The scalar weight adjustment in the output layer can be written as

$$
\begin{aligned}
\Delta V_{h^2 h^1}(n;j) &= -\eta \frac{\partial E}{\partial V_{h^2 h^1}(n;j)} \\
&= \eta \delta_{Y_{h^2}}(n) Z_{h^1}(n\beta + j). && \text{(A.13)}
\end{aligned}
$$

By assembling $\delta_{Y_{h^2}}(n)$ in Eq. (A.11) for all h^2, we get

$$\begin{aligned}
\Delta_{\boldsymbol{Y}(n)} &= \begin{bmatrix} \delta_{Y_1}(n) \\ \vdots \\ \delta_{Y_{h^2}}(n) \\ \vdots \\ \delta_{Y_{H^2}}(n) \end{bmatrix} = \frac{1}{N} \begin{bmatrix} d_1 - O_1 \\ \vdots \\ d_{h^2} - O_{h^2} \\ \vdots \\ d_{H^2} - O_{H^2} \end{bmatrix} \circ \begin{bmatrix} a'(netY_1(n)) \\ \vdots \\ a'(netY_{h^2}(n)) \\ \vdots \\ a'(netY_{H^2}(n)) \end{bmatrix} \\
&= \frac{1}{N}(d - O) \circ a'(netY(n)), \qquad \text{(A.14)}
\end{aligned}$$

where $\circ$ is the operator of one-by-one array multiplication defined in Eq. (2.5). Thus, the synapse adjustment in the output layer can be obtained as

$$\Delta V(n;j) = \eta \Delta_{\boldsymbol{Y}(n)} \cdot \boldsymbol{Z}(n\beta + j)^T. \qquad \text{(A.15)}$$

For the hidden layer, we have

$$\frac{\partial E}{\partial W_{h^1 h^0}(q;i)} = \sum_{h^2} \frac{\partial E}{\partial O_{h^2}} \sum_{(n,j)\in\varphi} \frac{\partial O_{h^2}}{\partial Y_{h^2}(n)} \frac{\partial Y_{h^2}(n)}{\partial netY_{h^2}(n)} \frac{\partial netY_{h^2}(n)}{\partial Z_{h^1}(n\beta + j)} \frac{\partial Z_{h^1}(n\beta + j)}{\partial W_{h^1 h^0}(q;i)}. \qquad \text{(A.16)}$$

Using the definition of $\delta_{Y_{h^2}}(n)$ in Eq. (A.11) and changing the order of summation, Eq. (A.16) is reduced to

$$\frac{\partial E}{\partial W_{h^1 h^0}(q;i)} = -\sum_{(n,j)\in\varphi} \sum_{h^2} \delta_{Y_{h^2}}(n) \frac{\partial netY_{h^2}(n)}{\partial Z_{h^1}(n\beta + j)} \frac{\partial Z_{h^1}(n\beta + j)}{\partial W_{h^1 h^0}(q;i)}, \qquad \text{(A.17)}$$

where $\varphi = \{(n;j) | n\beta + j = q\}$ is the set of all fan-outs of hidden node $Z_{h^1}(q)$. The remained partial terms are obtained by

$$\frac{\partial netY_{h^2}(n)}{\partial Z_{h^1}(n\beta + j)} = V_{h^2 h^1}(n;j), \qquad \text{(A.18)}$$

and

$$\begin{aligned}
\frac{\partial Z_{h^1}(n\beta + j)}{\partial W_{h^1 h^0}(q;i)} &= \frac{\partial Z_{h^1}(q)}{\partial W_{h^1 h^0}(q;i)} \\
&= a'(netZ_{h^1}(q)) X_{h^0}(q\alpha + i). \qquad \text{(A.19)}
\end{aligned}$$

The final result of Eq. (A.16) is

$$
\begin{aligned}
\frac{\partial E}{\partial W_{h^1 h^0}(q;i)} &= -\sum_{(n,j)\in\varphi}\sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2 h^1}(n;j) a'(netZ_{h^1}(q)) X_{h^0}(q\alpha+i) \\
&= -\delta_{Z_{h^1}}(q) X_{h^0}(q\alpha+i),
\end{aligned}
\tag{A.20}
$$

where

$$
\delta_{Z_{h^1}}(q) = \sum_{(n,j)\in\varphi}\sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2 h^1}(n;j) a'(netZ_{h^1}(q)). \tag{A.21}
$$

Collecting Eq. (A.21) for all h^1, we can get the error signals of the cell $\boldsymbol{Z}(q)$ as

$$
\begin{aligned}
\Delta_{\boldsymbol{Z}(q)} &= \begin{bmatrix} \delta_{Z_1}(q) \\ \vdots \\ \delta_{Z_{h^1}}(q) \\ \vdots \\ \delta_{Z_{H^1}}(q) \end{bmatrix} \\
&= \sum_{(n,j)\in\varphi} \begin{bmatrix} \sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2 1}(n;j) \\ \vdots \\ \sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2 h^1}(n;j) \\ \vdots \\ \sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2 H^1}(n;j) \end{bmatrix} \circ \begin{bmatrix} a'(netZ_1(q)) \\ \vdots \\ a'(netZ_{h^1}(q)) \\ \vdots \\ a'(netZ_{H^1}(q)) \end{bmatrix} \\
&= \sum_{(n,j)\in\varphi} \boldsymbol{V}^T(n;j) \cdot \Delta_{\boldsymbol{Y}(n)} \circ a'(\boldsymbol{netZ}(q)).
\end{aligned}
\tag{A.22}
$$

Therefore, the weight adjustment for the hidden layer is derived as

$$
\Delta \boldsymbol{W}(q;i) = \eta \Delta_{\boldsymbol{Z}(q)} \cdot \boldsymbol{X}(q\alpha+i)^T. \tag{A.23}
$$

Appendix B

Technical Notes on Bimodal Speech Recogntion System

Bibliography

[1] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 3, pp. 328-338, March 1989.

[2] K. Fukushima, S. Miyake, and T. Ito, "Neocognitron: a neural network model for a mechanism of visual pattern recognition," *IEEE Trans. Syst., Man., Cybern.*, vol. SMC-13, No. 5, pp. 826-834, Oct. 1983.

[3] O. Matan, C.J.C. Burges, Y. LeCun, and J.S. Denker, "Multi-digit recognition using a space displacement neural network," In *Advances in Neural Information Processing System 4*, pp. 488-495, 1992, San Mateo, CA: Morgan Kaufmann.

[4] J. Keeler and D.E. Rumelhart, "A self-organizing integrated segmentation and recognition neural net," In *Advances in Neural Information Processing System 4*, pp. 496-503, 1992, San Mateo, CA: Morgan Kaufmann.

[5] E. Sackinger, B.E. Boser, J. Bromley, Y. LeCun, and L.D. Jackel, "Application of the ANNA neural network chip to high-speed character recognition," *IEEE Trans. on Neural Networks*, vol. 3, No. 3, pp. 498-505, May 1992.

[6] T.S. Huang and A.N. Netravali, "Motion and structure from feature correspondences: A review," *Proceedings of the IEEE*, Vol. 82, No. 2, pp. 252-268, Feb. 1994.

[7] K. Rohr, “Toward model-based recognition of human movements in image sequences,” *CVGIP: Image Understanding*, Vol. 59, No. 1, pp. 94-115, Jan. 1994.

[8] C. Cedras and M. Shah, “A survey of motion analysis from moving light displays,” *Proc. 1994 IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 214-221, IEEE Comput. Soc. Press, 1994.

[9] J. Yamato, J. Ohya, and K. Ishii, “Recognizing human action in time-sequential image using hidden markov model,” *Proc. 1992 IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 379-385, IEEE Comput. Soc. Press, 1992.

[10] G.I. Chiou and J.N. Hwang, “Image sequence classification using a neural network based active contour model and a hidden markov model,” *Proc. ICIP-94*, pp. 926-930, IEEE Comput. Soc. Press, 1994.

[11] C. Bregler, S. Manke, H. Hild, A. Waibel, “Bimodal sensor integration on the example of 'speechreading',” *1993 IEEE International Conference on Neural Networks*, pp. 667-671, 1993.

[12] P. Duchnowski, M. Hunke, D. Busching, U. Meier, and A. Waibel, “Toward movement-invariant automatic lipreading and speech recognition,” *1995 International Conference on Acoustics, Speech, and Signal Processing*, pp. 109-112, 1995.

[13] R. Polana and R. Nelson, “Recognizing activities,” *Proc. of ICPR*, pp. 815-818, IEEE Comp. Soc. Press, 1994.

[14] R. Polana and R. Nelson, “Detecting activities,” *Journal of Visual Communication and Image Representation*, No. 5, Vol. 2, pp. 172-180, 1994.

[15] R. Polana and R. Nelson, "Low-level recognition of human motion (or how to get your man without finding his body parts)," *Porceedings of the 1994 IEEE Workshop on Motion of Non-Regid and Articulated Objects*, pp. 77-82, IEEE Comput. Soc. Press, 1994.

[16] S. Haykin, *Neural Networks : A comprehensive foundation*, pp. 498-515, 1994, Macmillan College Publishing Company, Inc.

[17] E.A. Wan, "Temporal backpropagation for FIR neural networks," *IEEE International Joint Conference on Neural Networks*, Vol. 1, pp. 575-580, 1990.

[18] E.A. Wan, "Temporal backpropagation: An efficient algorithm for finite impulse response neural networks," In *Proceedings of the 1990 Connectionist Models Summer School*, (D.S. Touretzky, J.L. Elman, T.J. Sejnowski, and G.E. Hinton, eds.), 1990, pp. 131-140, San Mateo, CA:Morgan Kaufmann.

[19] A. Back, E.A. Wan, S. Lawrence, A.C. Tsoi, "A unifying view of some training algorithms for multilayer perceptrons with FIR filter synapses," In *Proceedings of the 1994 IEEE workshop*, 1994, pp. 146-54.

[20] E. Petajan, B. Bischoff, N. Brooke, and D. Bodoff, "An improved automatic lipreading system to enhance speech recognition," *CHI 88*, pp. 19-25.

[21] K. Mase and A. Pentland, "Lip reading: automatic visual recognition of spoken words," *Image Understanding and Machine Vision 1989*, vol. 14, pp. 124-127, Technical Digest Series 1989.

[22] A.J. Goldschen, "Continuous automatic speech recognition by lipreading," Ph.D. thesis, George Washington Univ., Sept. 1993.