International Journal of Neural Systems, Vol. 9, No. 4 (August, 1999) 311–334

A SPACE-TIME DELAY NEURAL NETWORK FOR MOTION RECOGNITION AND ITS APPLICATION TO LIPREADING

CHIN-TENG LIN, HSI-WEN NEIN and WEN-CHIEH LIN
Department of Electrical and Control Engineering,
National Chiao-Tung University, Hsinchu, Taiwan, R.O.C.

Received March 1999
Revised July 1999
Accepted July 1999

Motion recognition has received increasing attention in recent years owing to heightened demand for computer vision in many domains, including the surveillance system, multimodal human computer interface, and traffic control system. Most conventional approaches classify the motion recognition task into partial feature extraction and time-domain recognition subtasks. However, the information of motion resides in the space-time domain instead of the time domain or space domain independently, implying that fusing the feature extraction and classification in the space and time domains into a single framework is preferred. Based on this notion, this work presents a novel Space-Time Delay Neural Network (STDNN) capable of handling the space-time dynamic information for motion recognition. The STDNN is unified structure, in which the low-level spatiotemporal feature extraction and high-level space-time-domain recognition are fused. The proposed network possesses the spatiotemporal shift-invariant recognition ability that is inherited from the time delay neural network (TDNN) and space displacement neural network (SDNN), where TDNN and SDNN are good at temporal and spatial shift-invariant recognition, respectively. In contrast to multilayer perceptron (MLP), TDNN, and SDNN, STDNN is constructed by vector-type nodes and matrix-type links such that the spatiotemporal information can be accurately represented in a neural network. Also evaluated herein is the performance of the proposed STDNN via two experiments. The moving Arabic numerals (MAN) experiment simulates the object's free movement in the space-time domain on image sequences. According to these results, STDNN possesses a good generalization ability with respect to the spatiotemporal shift-invariant recognition. In the lipreading experiment, STDNN recognizes the lip motions based on the inputs of real image sequences. This observation confirms that STDNN yields a better performance than the existing TDNN-based system, particularly in terms of 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.

1. Introduction

Space and time coordinate the physical world that surrounds us. Physical objects exist at some space-time point. Such objects may be idle or active, and their forms or behaviors may vary over time. Despite these distortions, people can inherently recognize the objects. To construct an ideal recognizer capable of dealing with natural patterns in daily life, e.g., speech, image, or motion, the recognizer should remain insensitive to the patterns' distortions in the time or space domain, or both.

Some criteria are available to assess the recognizer's tolerance for distortions of input patterns. For instance, the translation-invariant property of a recognizer implies that the recognizer can accurately recognize an object regardless of its proximity. Figure 1 summarizes these criteria for time-domain and space-domain distortions. Some physical analogies between time-domain and

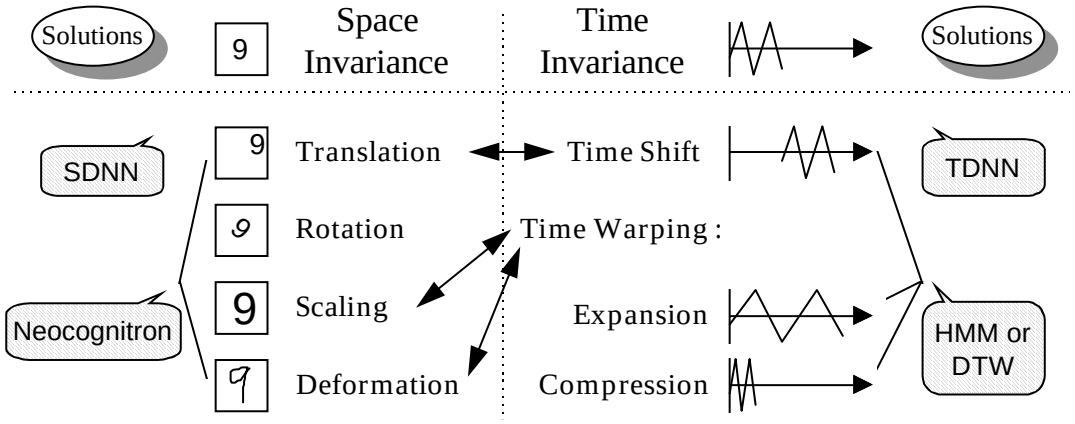

Fig. 1. Invariant criteria for an ideal recognizer.

space-domain criteria 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 criteria (in the time domain) corresponds to the scaling- and deformation-invariant criteria (in the space domain) as well.

Resolving these distortion problems either on the time domain or space domain[a] has received considerable interest. Typical examples include speech recognition and image recognition, which are on the time domain and space domain, respectively. Previous investigations have attempted to solve these 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. Figure 1 also contains some of their typical classifiers. According to this figure, in the space domain, Neocognitron[2] and Space Displacement Neural Network (SDNN) are used to recognize optical characters. The neocognitron overcomes the patterns' deformation and translation problem, while SDNN solves the patterns' translation problem. The SDNN referred to herein indeed represents a class of neural networks.[3–5] 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; in addition, weight sharing is applied to all the 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 are used for speech recognition. The TDNN overcomes the patterns' time-shift distortion, while the other classifiers eliminate the patterns' time-warping distortion.

However, integrated space-time-domain recognition has seldom been mentioned, particularly motion recognition. The dashed line between the time domain and space domain in Fig. 1 is just like a watershed that separates the research fields of space-domain and time-domain pattern recognition in the past. Previous experience in developing a bimodal speech recognition system that incorporates the image recognition techniques into a conventional speech recognition system allows us not only to acquire the rich sources from these two areas, but also to combine the space-domain recognition and time-domain recognition.

In the following, the problems to be solved are clarified since motion recognition is a diverse topic. First, we proceed with motion recognition on monocular images that are widely adopted in related investigations. Using monocular images is advantageous in that humans can only recognize a motion

[a]The space domain referred to herein is indeed the 2-D image space domain.

depending on an image sequence. Therefore, we believe that machine vision can also perform the same task; besides, in the monocular images, the problem of recovering 3-D geometry does not need to be recovered. Another merit of using monocular images is that the input data are real image sequences. Some earlier investigations attached markers on the actor to facilitate the analysis; however, this is impractical in real situations. This work focuses primarily on the main part of recognition and the local tracking of the actor.[b] Herein, global tracking is assumed to be supported by other systems. The motion types can be categorized as nonstationary motion and stationary motion. An object's motion is usually accompanied by a translation in position. Occasionally, the translation is much larger than our eyesight. For instance, the motion of a flying bird includes a motion of wings and a translation of body movement in 3-D space. This example typifies the case of nonstationary motion. On other occasions, the translation is within our eyesight, e.g., 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 deemed necessary required.

Most conventional approaches classify the motion recognition task into spatial feature extraction[c] and time-domain recognition. In such approaches, an image sequence is treated 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 performed afterward. However, spatial feature extraction is generally domain-dependent, and some delicate preprocessing operations may be reguired. The developed recognition system is subsequently restricted to one application, since redesigning the feature extraction unit for different applications would be too time-consuming. On the other hand, for motion recognition, information does not only exist in the space domain or time domain separately, but also exists in the integrated space-time domain. Distinguishing between feature extraction and recognition in the space domain and time domain would be inappropriate. Therefore, in this work feature extraction and classification are integrated in a single framework to achieve space-time-domain recognition.

Neural networks are adopted in the developed system herein, since their powerful learning ability and flexibility have been demonstrated in many applications. To achieve this goal, a model must be developed, capable of treating the space-time 3-D dynamic information. Restated, the model should be capable of learning the 3-D dynamic mapping that maps the input pattern varying in the space-time 3-D domain to a desired class. However, according to our results, the conventional neural network structure cannot adequately resolve this problem. 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 enhances the ability of the neural networks to learn 2-D dynamic mapping. The related development is curtailed since previous literature has not clarified the need for such models. A more important reason is that the ordinary constituents, which are used in MLP, TDNN, and SDNN, are difficult in terms of constructing the complex network that can represent and learn the 3-D, or higher dimensional dynamic information.

In light of the above discussion, this work presents a novel Space-Time Delay Neural Network (STDNN) that embodies the space-time-domain recognition and the spatiotemporal feature extraction in a single neural network. 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 constructing every layer into a spatiotemporal cuboid, STDNN preserves the

[b]The actor referred to herein implies the area of interest in a motion. This actor may be a human, an animal, an object, or any other object that performs the motion.

[c]The feature extraction referred to herein includes all the processes needed to transform an image into a feature vector.

inherent spatiotemporal relation in motion. In addition, the size of the cuboids shrinks increasingly 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 on the basis of the backpropagation rule.

The proposed STDNN possesses the shift-invariant property in the space-time domain because it inherits the TDNN's shift-invariant property and SDNN's translation-invariant property. STDNN's spatiotemporal shift-invariance ability implies that accurate tracking in the space-time domain is unnecessary. Once the whole motion occurs in the image sequence, the STDNN can handle it. The actor does not need to be located in the centroid; nor must the actor start his/her motion in the beginning of the image sequence.

The rest of this paper is organized as follows. Section 2 reviews previous work on the motion recognition problem. Sections 3 and 4 describe STDNN's structure and learning algorithms, respectively. Section 5 presents two recognition experiments: moving Arabic numerals (MAN) and lipreading. The former exhibits the STDNN's spatiotemporal shift-invariant property, while the later shows the practical application of the STDNN. Concluding remarks are finally made.

2. Previous Work

Research involving motion in computer vision has, until recently, focused primarily on geometry-related issues, either the three-dimensional geometry of a scene or the geometric motion of a moving camera.[6] A notable example that is closely related to motion recognition is the modeling of human motion. In this area, many researchers are attempting to recover the three-dimensional geometry of moving people.[7,8] Of particular interest is, the interpretation of moving light displays (MLDs),[8] which has received considerable attention. In the experiments of MLDs, some bright spots are attached to an actor dressed in black; the actor then moves in front of a dark background. The results of MLDs depend heavily on the ability to solve the correspondence problem and accurately track joint and limb positions.

Motion recognition has received increasing attention as of late. Yamato *et al.*[9] used HMM to recognize different tennis strokes. In their scheme, the image sequence was transformed into feature (mesh feature) vector sequence and, then, was vector quantized to a symbol sequence. The symbol sequences were used as input sequences for HMM in training and testing. Although the temporal invariance was accomplished by HMM, spatial invariance was not fulfilled since the mesh feature is sensitive to the position-displacement as described in their experiments. Chiou and Hwang[10] also adopted HMM as the classifier in the lipreading task; however, the features fed into HMM were extracted by a neural-network-based active contour model. Waibel *et al.*[11] developed a bimodal speech recognition system, in which a TDNN was used to perform lipreading, and downsampled images or 2D-FFT transformed images were used as features. In the last two cases of lipreading,[10,12] the image sequences are color images and the color information was used to locate 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] emphasized 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 investigators measured the period of an object's motion using a Fourier transform.[14,15] By assuming that the object producing periodic motion moves along a locally linear trajectory and the object's height does not vary over time, it achieves translation and scale invariance in the space domain. In the recognizing stage, a template matching the 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. In general, Polana and Nelson achieved most of the invariant criteria in the time and space domains for periodic motions. However, assuming that the entire image sequence must consist of at least four cycles of the periodic motion may be unrealistic under some circumstances. In some human motions, such as running and walking, which are examples used in their experiments, this assumption is appropriate. However, this assumption is unrealistic for cases such as open-the-door action,

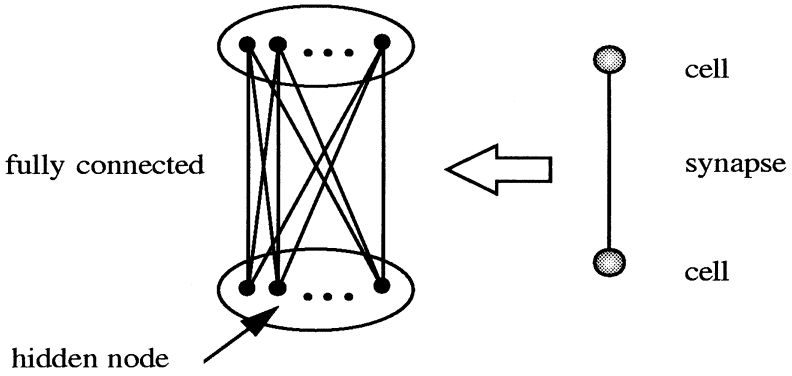

Fig. 2. The cells and synapses in the STDNN.

sit-down action, facial expressions, hand gestures, or lipreading.

3. Space-Time Delay Neural Network (STDNN)

The motion recognition problem involves processing information of time, space, and features. This high-dimensional information, however, is not easily represented explicitly by ordinary neural networks because the basic units and interconnections used in these neural networks are frequently 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 STDNN proposed herein 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. Therefore the time and space relation can be represented as a 3-D box, i.e., the manner in which we visualize the time and space information. Figure 2 illustrates the cell and synapse, as well as the hidden nodes and interconnections inside them.

Figure 3 depicts a three-layer STDNN. 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 case. However in other cases, where the elements of input data may be a vector, such a cell could contain as many nodes according to the length of the vector. Based on the contextual relation of time and space, the cells arranged along the t-axis, y-axis, and x-axis form a 3-D cuboid in each layer. The cells in the hidden layer generally contain multiple hidden nodes, so that a sufficient dimension space is available 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 instance, if four classes of inputs are to be classified, the number of hidden nodes should be four. According to Fig. 3, the information is concentrated layer by layer; the final output is obtained by summing up the outputs of cells in the output layer and taking the average. The final 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 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 net inputs from the activation values of cells covered in the receptive box. The net input of the cell Z at layer l with location (q_t, q_y, q_x) can be expressed as:

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

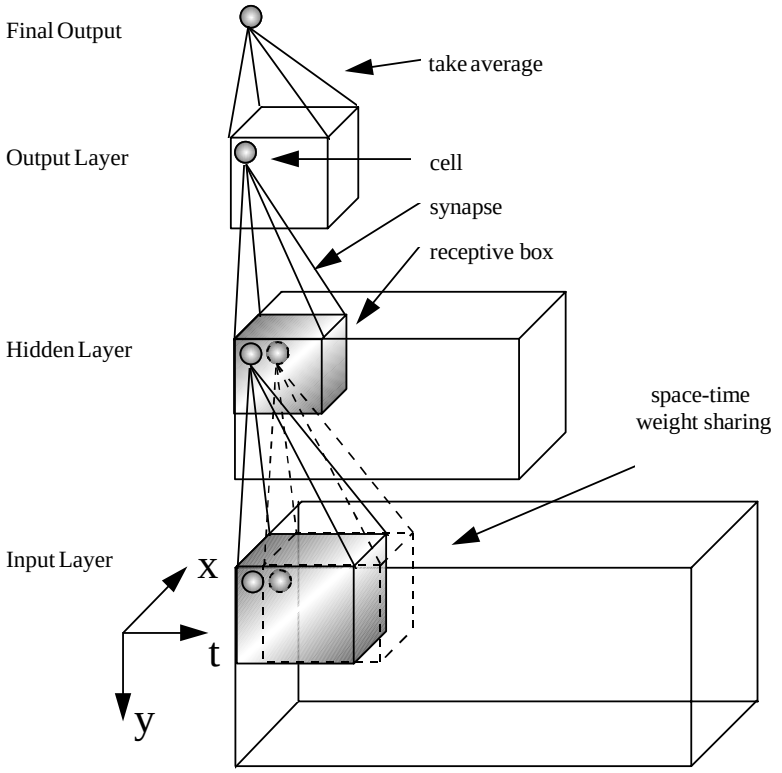

Fig. 3. A three-layer STDNN.

where $\boldsymbol{NetZ}(\cdot) \in \Re^{H^l \times 1}$ denotes the net input of cell Z, $\boldsymbol{X}(\cdot) \in \Re^{H^{l-1} \times 1}$ represents 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:

$$\boldsymbol{Z}(q_t, q_y, q_x) = a(\boldsymbol{NetZ}(q_t, q_y, q_x)), \tag{2}$$

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 Eqs. (1) and (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 the 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 the t-axis, y-axis, and x-axis, respectively.

Figure 4 displays the practical view of Eqs. (1) and (2). When the cell Z is located 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 is the local coordinates that are only relative to the

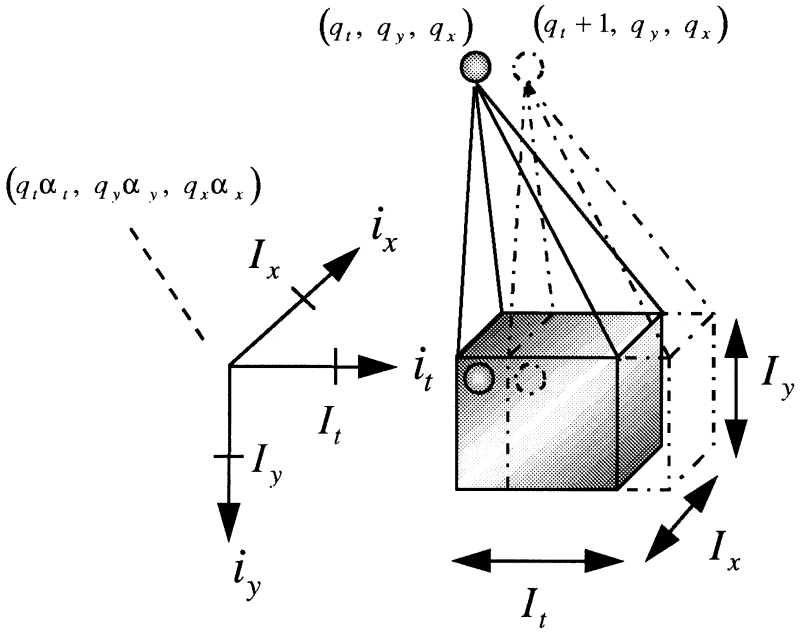

Fig. 4. Practical view of Eqs. (1) and (2).

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

To clarify the mathematical symbols, the notations of 3-D indexes are redefined as follows:

$$\begin{aligned} \boldsymbol{q} &\equiv (q_t, q_y, q_x)\,, \\ \boldsymbol{i} &\equiv (i_t, i_y, i_x)\,, \\ \boldsymbol{I} &\equiv (I_t, I_y, I_x)\,, \\ \boldsymbol{\alpha} &\equiv (\alpha_t, \alpha_y, \alpha_x)\,, \end{aligned} \tag{3}$$

In addition, Eqs. (1) and (2) can be rewritten as:

$$\boldsymbol{NetZ}(\boldsymbol{q}) = \sum_{\boldsymbol{i}=\boldsymbol{0}}^{\boldsymbol{I}-\boldsymbol{1}} \boldsymbol{W}(\boldsymbol{i})\boldsymbol{X}(\boldsymbol{q}\circ\boldsymbol{\alpha}+\boldsymbol{i}) + \boldsymbol{B}(\boldsymbol{q})\,, \tag{4}$$

$$\boldsymbol{Z}(\boldsymbol{q}) = \boldsymbol{a}(\boldsymbol{NetZ}(\boldsymbol{q}))\,, \tag{5}$$

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 is frequently used herein.

The operations of STDNN can be viewed in another way. The receptive box travels in a spatiotemporal hyperspace and reports its findings to the corresponding 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 the cells in the next layer are filled, and the searching in the next layer starts. In this manner, the locally spatiotemporal features in every receptive box are extracted and fused layer by layer untill the final output is generated. In other words, STDNN gathers the locally spatiotemporal features that appear at any of the regions in hyperspace to make the final decision. Hence, the STDNN possesses the shift-invariant ability in both time and space domains.

4. Learning Algorithms for the STDNN

The training of 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). The FIR MLP has been disscussed. Owing to the difference in the manner in which the gradients are computed and the error function is defined, many different forms of training algorithms for the FIR MLPs have been derived.[16–19]

Unfortunately, these training algorithms for FIR MLP cannot be directly applied to train STDNN

because of the different constituents of STDNN and FIR MLP. In FIR MLP, the synapse is treated as a FIR filter in which every coefficient represents a weight value on a specific delay link. In constrast to 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.

Herein, we derive the training algorithms of the STDNN from a different viewpoint. Unlike the FIR MLP that embeds the time-delay information into a FIR filter, the STDNN embeds the feature information into a cell such that time-space information can easily be visualized. Consequently, the training algorithms for the STDNN are derived from the perspective 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 first used in the training of the TDNN.[1] In this method, a static equivalent network is constructed by unfolding the STDNN in time and space; the standard backpropagation algorithm is then 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 are referred to herein as the *static method*, and *instantaneous error method*, respectively.

4.1. *Static method for training STDNN*

For clear explanation, a learning algorithm for the 1-D degenerative case of STDNN is discussed first. This degenerative network is referred to herein as 1-D STDNN. According to Fig. 5, 1-D STDNN is actually a TDNN; however, the hidden nodes in TDNN originally lined up in each vertical axis are grouped in a cell.

Figure 6 depicts a three-layer 1-D STDNN unfolded in time. According to this figure, the cell in the input layer is denoted as $\boldsymbol{X}(m) \in \Re^{H^{L-2}\times 1}$, where m represents 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 $\boldsymbol{V}(n;\, j) \in \Re^{H^{L}\times H^{L-1}}$ connects the cell $\boldsymbol{Y}(n)$ and $\boldsymbol{Z}(n\beta+j)$, where β denotes the offset each time the receptive field moves, and j represents the number of unit-delays, ranging from 0 to $J-1$. The parameter J, i.e., the total number of unit-delays, can also be practically viewed as the length of the receptive field. As Fig. 6 indicates, 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 $\boldsymbol{Y}(n+1)$.

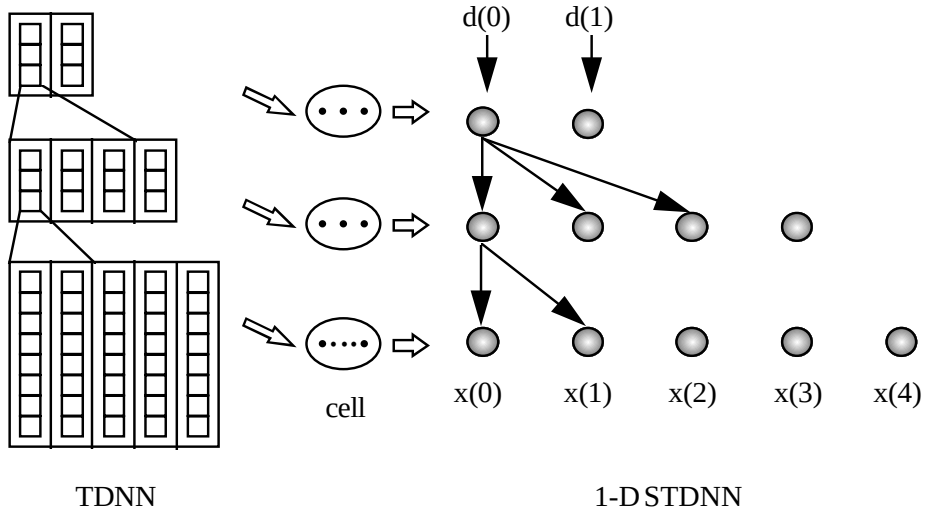

Fig. 5. An 1-D STDNN and TDNN.

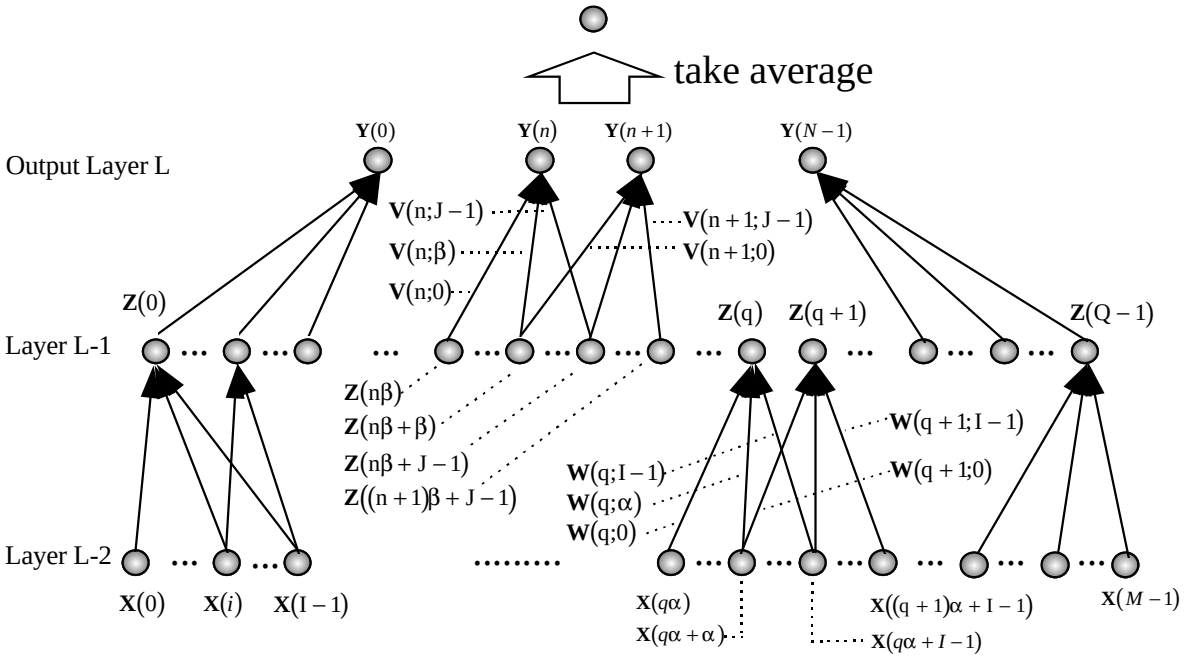

Fig. 6. An 1-D STDNN unfolded in time.

In the unfolded 1-D STDNN, many synapses are replicated. To monitor which of the 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; however, 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 denotes the offset each time the receptive field moves, and i represents the number of unit-delays.

According to Eq. (5), the output of these cells can be expressed by:

$$\boldsymbol{Y}(n) = a\left(\sum_{j=0}^{J-1} \boldsymbol{V}(n;\, j)\boldsymbol{Z}(n\beta + j) + \boldsymbol{C}(n)\right), \qquad n = 0, \ldots, N-1\,, \quad (6)$$

$$\boldsymbol{Z}(q) = a\left(\sum_{i=0}^{I-1} \boldsymbol{W}(q;\, i)\boldsymbol{X}(q\alpha + i) + \boldsymbol{B}(q)\right), \qquad q = 0, \ldots, Q-1\,, \quad (7)$$

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,

$$\boldsymbol{O} = \frac{1}{N}\sum_{n=0}^{N-1} \boldsymbol{Y}(n)\,. \quad (8)$$

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

$$E = \frac{1}{2}\,(\boldsymbol{d} - \boldsymbol{O})^{\boldsymbol{T}}(\boldsymbol{d} - \boldsymbol{O})\,. \quad (9)$$

Applying the gradient-descent method to minimize the above error function, leads to:

$$\Delta \boldsymbol{V}(n;\, j) = -\eta\,\frac{\partial E}{\partial \boldsymbol{V}(n;\, j)}\,, \quad (10)$$

$$\Delta \boldsymbol{W}(q;\, i) = -\eta\,\frac{\partial E}{\partial \boldsymbol{W}(q;\, i)}\,. \quad (11)$$

Thus, the synapse updating rule of the output layer and hidden layer can be obtained by differentiating the error function of Eq. (9) with respect to the matrix $\boldsymbol{V}(n;\, j)$, and $\boldsymbol{W}(q;\, i)$. Only the resulting equations are listed herein. 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:

$$\Delta \boldsymbol{V}(n;\, j) = \eta \Delta_{\boldsymbol{Y}(n)} \cdot \boldsymbol{Z}(n\beta + j)^T\,, \qquad (12)$$

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

$$\Delta_{\boldsymbol{Y}(n)} = \frac{1}{N}\,(\boldsymbol{d} - \boldsymbol{O}) \circ a'(\boldsymbol{netY}(n))\,, \qquad (13)$$

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

$$\boldsymbol{netY}(n) = \sum_{j=0}^{J-1} \boldsymbol{V}(n;\, j)\boldsymbol{Z}(n\beta + j)\,. \qquad (14)$$

Weight updating rule for the hidden layer

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

$$\Delta \boldsymbol{W}(q;\, i) = \eta \Delta_{\boldsymbol{Z}(q)} \cdot \boldsymbol{X}(q\alpha + i)^T\,, \qquad (15)$$

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

$$\Delta_{\boldsymbol{Z}(q)} = \sum_{(n,j)\in\varphi} \boldsymbol{V}(n;\, j)^T \cdot \Delta_{\boldsymbol{Y}(n)} \circ a'(\boldsymbol{netZ}(q))\,, \qquad (16)$$

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

$$\boldsymbol{netZ}(q) = \sum_{i=0}^{I-1} \boldsymbol{W}(q;\, i)\boldsymbol{X}(q\alpha + i)\,. \qquad (17)$$

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

$$\Delta V(j) = \frac{1}{N} \sum_{n=0}^{N-1} \Delta V(n;\, j)\,, \qquad (18)$$

$$\Delta W(i) = \frac{1}{Q} \sum_{q=0}^{Q-1} \Delta W(q;\, i)\,, \qquad (19)$$

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 manner, a cell's bias values are adjusted 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 neglect the fact that each node considered herein is a vector and the weight link is a matrix, but also 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. Restated, only the timing indexes, number of unit-delays, and offsets by Eq. (3) need to be changed.

4.2. *Instantaneous error method for training STDNN*

The instantaneous error method is derived by unfolding the STDNN in another way, which is originally used for the online adaptation of the FIR MLP. From the experience of deriving of the first learning algorithm, we begin with the derivation of the second one from the 1-D STDNN, owing to its clarity and ease of generalization.

Figure 7 illustrates the difference between these two unfolding methods. Figure 7(a) displays a three-layer 1-D STDNN, in which the moving offset of the receptive field in each layer is one cell for each time. Figures 7(b) and 7(c) depict its static equivalent networks unfolded by the first and second methods, respectively. According to Fig. 7(c), many smaller subnetworks are replicated. The number of subnetworks equals the number of cells in the output layer of 1-D STDNN. The instantaneous outputs are subsequently generated by these subnetworks whenever sufficient input data are coming. For example, as shown in Fig. 7(c), $\boldsymbol{Y}(0)$ is generated by the first subnetwork when the input sequence from $\boldsymbol{X}(0)$ to $\boldsymbol{X}(3)$ is present. 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 $\boldsymbol{X}(4)$.

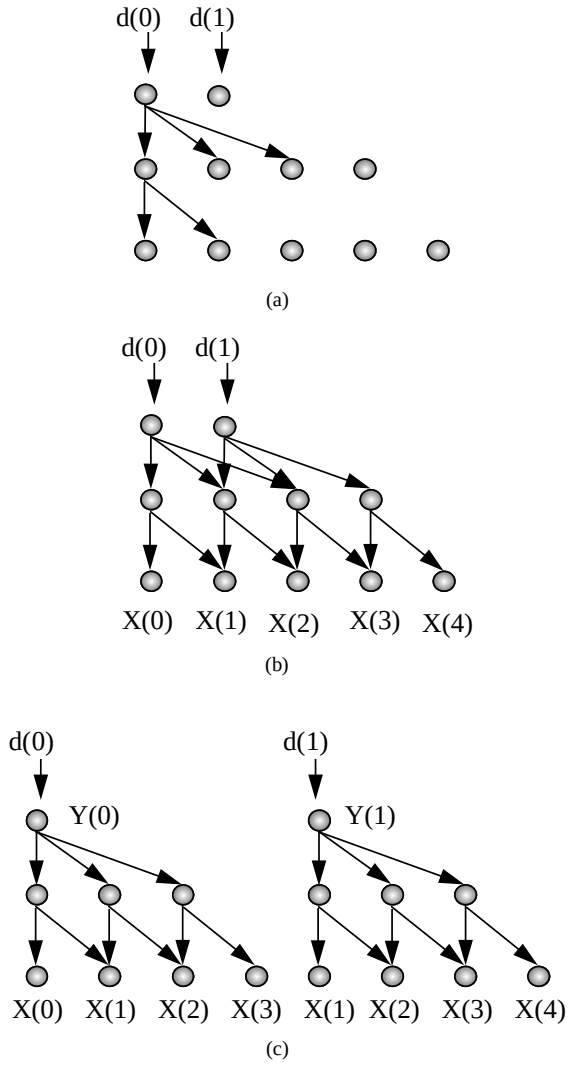

Fig. 7. Illustration of different unfolding methods of the 1-D STDNN.

In the online adaptation of FIR MLP, the synapses of the subnetwork are adjusted at each time step. Restated, every time the output $\boldsymbol{Y}(n)$ is produced, the synapses are immediately adjusted according to the desired output $\boldsymbol{d}(n)$. In the 1-D STDNN, however, the synapses are not immediately adjusted at each time step. In contrast, the changes of synapses are accumulated until the outputs of all the 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 adopt the accumulated-change manner to update the synapses. As generally known, the number of unfolded subnetworks of the STDNN generally exceeds that of the FIR MLP, accounting for why the online adaptation on every subnetwork takes much longer time in the STDNN. Therefore, when considering the training speed, the changes of the synapses are accumulated, and the synapses are updated once the output at the last time step is generated.

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

$$E(n) = (\boldsymbol{d}(n) - \boldsymbol{Y}(n))^T (\boldsymbol{d}(n) - \boldsymbol{Y}(n)) \,. \qquad (20)$$

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:

$$\Delta \boldsymbol{V}(n;\, j) = \eta \Delta_{\boldsymbol{Y}(n)} \cdot \boldsymbol{Z}(n\beta + j)^T \,, \qquad (21)$$

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

$$\Delta_{\boldsymbol{Y}(n)} = (\boldsymbol{d}(n) - \boldsymbol{Y}(n)) \circ a'(\boldsymbol{netY}(n)) \,. \qquad (22)$$

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:

$$\Delta \boldsymbol{W}(q;\, i) = \eta \Delta_{\boldsymbol{Z}(q)} \cdot \boldsymbol{X}(q\alpha + i)^T \,,$$
$$q = 0 \cdots R^{L-1} - 1 \text{ and } i = 0 \cdots I^{L-1} - 1 \,, \quad (23)$$

where the range of timing index q bounded by R^{L-1} denotes 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. (16).

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

$$R^{l-1} = R^l \times I^{l-1}\,, \quad R^L = 1 \text{ and } l = L,\, L-1, \ldots,\, 0\,. \tag{24}$$

For example, we have $N = 2$, $L = 2$, $R^2 = 1$, $I^1 = 3$, and $R^1 = 1 \cdot 3 = 3$ in Fig. 7(c).

Owing to that the number of replicated sets of synapses has changed; the weight average equations, Eqs. (18) and (19), are modified as:

$$\Delta V(j) = \frac{1}{N \cdot R^L} \sum_{n=0}^{N-1} \Delta V(n;\, j)\,, \tag{25}$$

$$\Delta W(i) = \frac{1}{N \cdot R^{L-1}} \sum_{n=0}^{N-1} \sum_{q=0}^{R^{L-1}} \Delta W(q;\, i)\,, \tag{26}$$

where N is the number of total subnetworks.

5. Recognition Experiments

Two groups of experiments are set up to evaluate the performance of the STDNN. 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.

5.1. *Moving arabic numerals*

A good neural network should have adequate generalization ability to deal with conditions that it has never learned before. The following experiment displays STDNN's generalization ability with respect to input shifting in space and time using the minimum training data. Although the experimental patterns may be rather simpler than those in a real situation, the designed patterns provide a good criterion for testing the shift-invariant capability.

In this experiment, an object's action appearing in different spatiotemporal regions is simulated by a man-made image sequence. Such an appearance 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. Man-made image sequences are used primarily for two reasons. First, many motions can be produced in a short time, thereby allowing for the experiments to be easily repeated. Second, some types of motions are difficult or even impossible for a natural object to perform; however, they can be easily performed by simulation. Each of the man-made images is generated by a 32×32 matrix using a C-language program. Each element of the matrix represents one pixel and each value of the elements is either 255 (white pixel) or 0 (black pixel). Thus, the resolution of the man-made image data is 32×32 pixels. In addition, the man-made bi-level image sequences are normalized to $[-1, +1]$ and the normalized image sequences are used for the input data of the STDNN.

According to Fig. 8, an image sequence consists of 16 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. Changing the order in which the numerals appear in the sequence leads to the formation of three classes of motion. The three classes are all constituted by eight basic actions (numerals); the first one and the second one act in

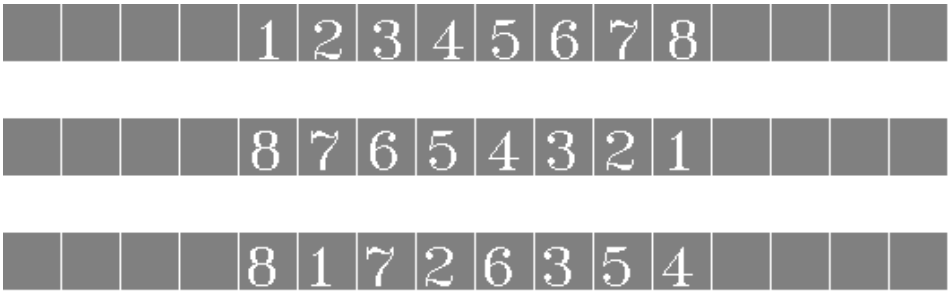

Fig. 8. All training patterns used in the MAN experiment.

the reverse order; meanwhile, the last one acts as the mixture of the other two. These motions can be viewed as watching a speaker pacing a platform. Under this circumstance, we easily concentrate our attention to his/her facial expressions or hand gestures however, the background he/she is situated in is neglected.

The experiment is performed 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. Figure 8 depicts all of the training patterns. The learning algorithm used for training STDNN is the static method described in Subsec. 4.1. It took 195 epoch until convergence. After training, the patterns with the numeral's (x, y, t) coordinates randomly moved are used to test the performance of STDNN.

Three sets of motions are tested according to the degree of freedom that numerals are allowed to move in a frame. The numerals in the first set are moved in a block manner, i.e., eight numerals are tied together, and their positions are changed in the same manner. In the second set of motion, the numerals are moved in a slowvarying manner, in which only the 1st, 3rd, 5th, and 7th numerals are randomly moved, and the positions of the rest are bound 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 Figs. 9, 10, and 11, respectively.

The recognition rates of STDNN on the three test sets, in which each class contains 30 patterns, are 85%, 80%, and 70%, respectively. Several correctly-recognized patterns are shown in Figs. 9 through 11. Table 1 lists STDNN's configuration used in this experiment, in which H represents the number of hidden nodes of the cell in each layer; (I_t, I_y, I_x) is 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 cells along the t-axis, y-axis, and x-axis.

Although the recognition rate in the MAN experiment is not very good, it is reasonable if we realize the disproportion between the hyperspace spanned

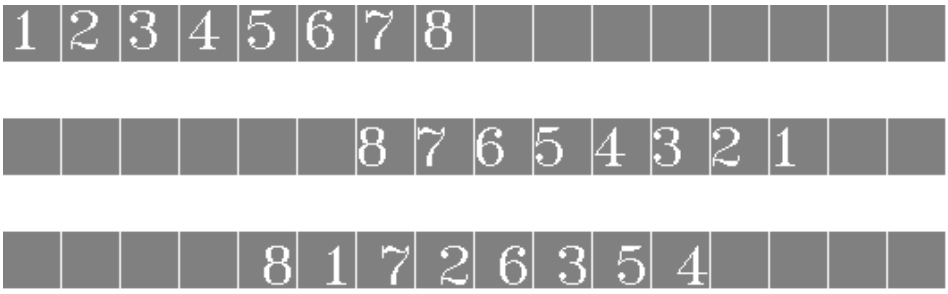

Fig. 9. Some stationary motion patterns in the first test set.

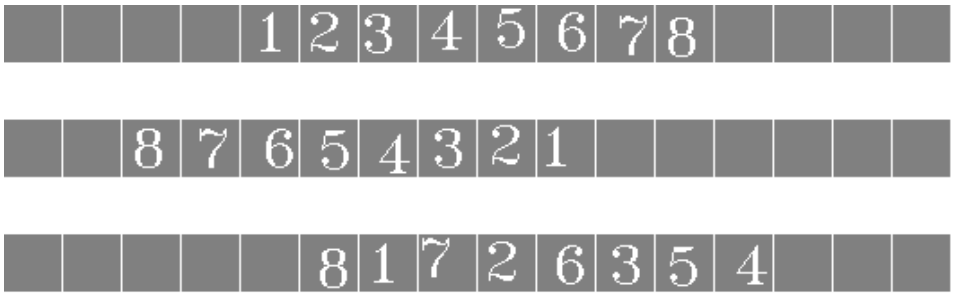

Fig. 10. Some nonstationary motion patterns in the second test set.

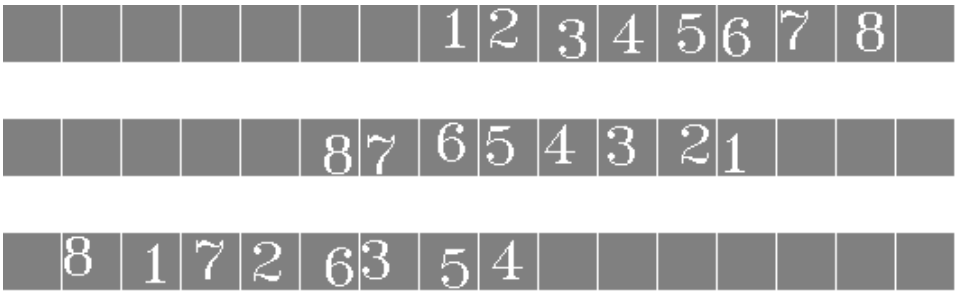

Fig. 11. Some highly nonstationary motion patterns in the third test set.

Table 1. The STDNN's network configuration in the MAN experiment.

	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	—	—	$(4, 3, 4)$

by only three training patterns and that spanned by 90 test patterns. Variations in the test image sequences include translations in space and time, as well as changes of the numerals' 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; however, this is not a desired situation. While considering a real situation, e.g., the pacing speaker mentioned earlier, it is impossible to gather a sufficient number of training patterns to cover the whole hyperspace for the speaker to pass through all of the time. Therefore, in this work we can merely 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 slightly under some realistic conditions. Every numeral in the final test set is freely moved; however, under some circumstances, the object might not move so fast. In particular, the human motion in the consecutive image sequence moves smoothly as usual. Therefore, the second set of test patterns, in which the moving of numerals is slowvary, is the closest to the real case.

5.2. *Lipreading*

The following experiment demostrates the feasibility of applying the system to lipreading. In particular, the performance of STDNN is compared with that of the lipreading recognition system proposed by Waibel *et al.*[11,12] 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, i.e., down-sampled images and 2D-FFT transformed images, are used. STDNN should be compared with Waibel's system for two reasons. First, STDNN and TDNN are neural-network-based classifiers. Second, TDNN has the time-shift invariant ability and 2D-FFT features are spatial-shift invariant: both of which are shift-invariant criteria of 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. The following experiments compare the performance of STDNN with that of Waibel's lipreading system based on the standpoints of the classifier or the whole recognition system. Besides, the comparison of the two learning algorithms for the STDNN is also discussed here. Most researchers involving FIR MLP learning algorithms compare the performance of their algorithms using simple prediction problems. Herein, an attempt is made to make such comparisons in terms of the real lipreading application, which is a more complex case.

5.2.1. *Experimental conditions*

The lip movements are recorded by a bimodal recognition system developed.[20] During the 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-lit condition and the speaker is seated before the camera at an appropriate distance. An image sequence contains 16 frames, and each frame is of size, 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; in addition, 13×13 FFT coefficients in a low-frequency area are selected from the magnitude spectrum of the 64×64 2D-FFT transformed image.

Whether the spatial-shift invariant ability is necessary if an AOI extraction unit is adopted must be addressed. Such an ability is still deemed necessary as the extracted lip region is not always accurate. Therefore, the recognition with spatial-shift invariant ability is still necessary. Moreover, the inherent nature of the speech hinders the lipreading, owing to the shift or scale distortions in the time domain compared with other kinds of motions. Therefore, in addition to the spatial-shift invariant ability, the time-shift invariant ability is also necessary in the lipreading task.

In this experiment, two Chinese vocabulary words are used for lipreading. One contains the

(a) Chinese digit: 1.

(b) Chinese digit: 2.

(c) Chinese digit: 3.

Fig. 12. Chinese digits: 1 to 6.

(d) Chinese digit: 4.

(e) Chinese digit: 5.

(f) Chinese digit: 6.

Fig. 12 (*cont'd*)

Chinese digits from 1 to 6, and the other contains acoustically confusable words consisting of 19 Chinese words.[d] These two vocabulary words are referred here as *Chinese digits* and *confusable words*, respectively. Figure 12 presents the six image sequences of Chinese digits.

Adhering to the same belief mentioned in the MAN experiment, we attempt to use as few training patterns as possible to test the generalization ability of TDNN and STDNN. In our experiment, five training patterns are used for each word. In the testing, two sets of patterns are used. One is recorded at the same time as for the training patterns, *but not used for training*. The other is recorded on another day. These two sets are referred to herein as the *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 13 displays the example patterns of Chinese digits recorded at different time intervals.

Two test sets are used to compare the performance of the systems under varying recording

[d]Some words can be easily confused acoustically; however, they are distinct in lip movements. With the auxiliary of lipreading, the bimodal speech recognition system can enhance the recognition rate of these words.

(a) Pattern in the training set.

(b) Pattern in the test set.

(c) Pattern in the new test set.

Fig. 13. Patterns of Chinese digits in the training set, test set, and new test set.

conditions. Acquired from the experience in developing an online bimodal speech recognition system,[20] the training data recorded at the same time are frequently quite uniform, so that the condition in the training set does not always correspond to the condition in an online environment. Although the recording condition can be maintained as uniform as possible, it cannot remain constant all of the time. This problem can be solved by enlarging the training set. However, constructing a large training set is frequently time consuming. A more practical and efficient means of solving this problem is to improve the learning ability of the classifier or select more robust features, or both.

5.2.2. *Results of experiment*

Tables 2 and 3 list the network configurations used for Chinese digits and confusable words recognition, respectively. By combining different types of input data and neural networks, four networks can be constructed for each vocabulary word. For all the 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.

According to these tables, TDNN is in fact a special case of 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 are passed to the hidden layer.

Table 2. Network configurations used for Chinese digit recognition.

	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 3. Network configurations used for confusable-word recognition.

	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)

For Chinese digits, Tables 4 and 5 summarize the experimental results for the networks trained by the static method and the instantaneous error method, respectively. Tables 6 and 7 summarize the results for confusable words.

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, the results of the experiment are compared with those of the classifier and the whole recognition system.

From the perspective of the classifiers, STDNN is better than TDNN according to the following observations. In this case, STDNN and TDNN are compared on the basis of the same input features. For the down-sampled image sequences, STDNN's recognition rate is 15%∼30% higher than that of the TDNN in terms of Chinese digits, and 0%∼30% in confusable words. This result is not surprising because only TDNN has the time-shift invariant recognition ability. TDNN cannot overcome the possible spatial shift in the down-sampled images. In contrast, if the input data are the 2D-FFT transformed images, STDNN's improvement in recognition rate is not so much. Owing to the space-shift invariant property of 2D-FFT, TDNN would not greatly suffer the space-shift problem. Nevertheless, the recognition rate of STDNN is still 5%∼12% higher than

Table 4. Recognition rates of Chinese digits, trained by the static method.

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 5. Recognition rates of Chinese digits, trained by the instantaneous error method.

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 6. Recognition rates of confusable words, trained by the static 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 7. Recognition rates of confusable words, trained by the instantaneous error method.

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

that of the TDNN in all the new test sets. This is due to the STDNN's high generalization ability as discussed later.

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

Of particular concern is the comparison of performance of the whole recognition system, since a unified structure of the motion recognition system,

that embodies a feature extraction unit and a classifier, is one of STDNN's goals. From the perspective of classifier the performance comparison attempts to isolate the effect of different features used in these systems. Experimental results indicate that STDNN possesses a better ability to extract more robust features, but also to learn more information from the training data.

For other lipreading-related research, the recognition rates vary significantly due to the different specifications and experimental conditions of their systems. Besides, most lipreading systems are auxiliary to a speech recognition system and, thus, the recognition rate of the whole system is of primary concern. Petajan's speech recognition system[21] is the first documented system that contains a lipreading subsystem. According to the results of that study, the bimodal speech recognition system achieved a recognition rate of 78%, while the speech recognizer alone has only a recognition rate of 65% for a 100-isolated words, speaker-dependent system. In our available reports on the recognition rates of the lipreading systems, Pentland[22] achieved a recognition rate of 70% for English digits in a speaker-independent system. Goldschen[23] achieved a recognition rate of 25% for 150 sentences in a speaker-dependent system. Waibel *et al.*[12] achieved a recognition rate of 30%∼53% for 62 phonemes or 42 visemes.[e] Generally speaking, although the lipreading system alone does not work very well, whether the goal of lipreading is auxiliary to speech recognition for achieving more robust recognition rate is irrelevant.

On the other hand, comparing the two different learning algorithms reveals that the instantaneous error method converges more rapidly than the static method by three- to fivefold. In particular, the training epochs are reduced from 2680 to 110 in the experiment of Chinese digits with down-sampled images.[f] Moreover, in most of our experiments, the network trained by the instantaneous error method has a higher recognition rate than that trained by the static method.

6. Conclusion

This work demonstrates STDNN's space-time-domain recognition ability. In the MAN experiment, STDNN possesses the spatiotemporal shift-invariant ability learned from the minimum training data. While in the lipreading experiment, STDNN has a higher recognition rate and better generalization ability than the TDNN-based system.[12] STDNN has several merits. First, it is a general approach to motion recognition since no *a priori* knowledge on the application domain is necessary. Second, the feature extraction capacity is embedded, therefore requiring minimum preprocessing. The input data are nearly the original image sequences except for the downsampling preprocessing. Third, STDNN possesses shift-invariant ability both in space and time domains such that the recourse for the tracking preprocess can be eliminated. Nevertheless, STDNN still has some drawbacks. For instance, a significant amount of time is required to process data, particularly in training. In training the STDNN on the downsampled image sequence, which is the worst case, it takes approximately one hour to run 6∼8 epochs on Sparc-20.[g] STDNN's processing time depends on its configurations. If the size and moving offset of the receptive box is small, searching through the spatiotemporal cuboids requires more time. However, the cost in processing time gains a higher generalization ability. Therefore, a trade-off occurs between the recognition rate and recognition speed. Another shortcoming is that the other invariant criteria in space and time such as rotation and scaling invariance, have not been treated in this research yet.

To increase the processing speed, STDNN's scale must be reduced, particularly in the input layer. As is generally known, humans do not see a moving pixel but a moving region, accounting for why some unsupervised learning mechanisms can be used to replace the input layer so that the redundancy can be reduced precedently. On the other hand, to treat other invariant criteria, the shape of the receptive box can be adapted in the learning stage so that the

[e]A distinct characteristic of lipreading is that the number of distinguishable classes is not directly proportional to the number of words in speech. For instance, /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 1-to-n mapping.

[f]The motivation for deriving the instantaneous error method is partially attributed due to the long training time taken in the static method.

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

deformation in the space-time domain can be overcome.

The novel constituents of STDNN provide further insight into its potential applications. The STDNN proposed herein suggests a neural network model capable of dealing with the 3-D dynamic information. This network can classify the patterns described by the multidimensional information which varies dynamically in a 3-D space. Besides, for the multidimensional signal processing, STDNN can be treated as a nonlinear 3-D FIR filter. On the other hand, the cells and synapses provide a viable means of constructing a more complex neural network that can treat higher-dimensional dynamic information.

Appendix A
Derivation Details of Learning Algorithms

We proceed with the derivation by expanding Eqs. (6)–(9) 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 Sec. 4. All of the denotations used here follow those used in Sec. 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 to in Eqs. (6)–(9) are:

$$\boldsymbol{Y}(n) = \begin{bmatrix} Y_1(n) \\ \vdots \\ Y_{h^2}(n) \\ \vdots \\ Y_{H^2}(n) \end{bmatrix} \qquad \boldsymbol{Z}(q) = \begin{bmatrix} Z_1(q) \\ \vdots \\ Z_{h^1}(q) \\ \vdots \\ Z_{H^1}(q) \end{bmatrix} \qquad \boldsymbol{X}(m) = \begin{bmatrix} X_1(m) \\ \vdots \\ X_{h^0}(m) \\ \vdots \\ X_{H^0}(m) \end{bmatrix} \tag{27}$$

and the expansions of the synapses are:

$$\boldsymbol{V}(n;\, j) = \begin{bmatrix} V_{11}(n;\, j) & \cdots & V_{1h^1}(n;\, j) & \cdots & V_{1H^1}(n;\, j) \\ \vdots & & \vdots & & \vdots \\ V_{h^2 1}(n;\, j) & \cdots & V_{h^2 h^1}(n;\, j) & \cdots & V_{h^2 H^1}(n;\, j) \\ \vdots & & \vdots & & \vdots \\ V_{H^2 1}(n;\, j) & \cdots & V_{H^2 h^1}(n;\, j) & \cdots & V_{H^2 H^1}(n;\, j) \end{bmatrix}, \tag{28}$$

$$\boldsymbol{W}(q;\, i) = \begin{bmatrix} W_{11}(q;\, i) & \cdots & W_{1h^0}(q;\, i) & \cdots & W_{1H^0}(q;\, i) \\ \vdots & & \vdots & & \vdots \\ W_{h^1 1}(q;\, i) & \cdots & W_{h^1 h^0}(q;\, i) & \cdots & W_{h^1 H^0}(q;\, i) \\ \vdots & & \vdots & & \vdots \\ W_{H^1 1}(q;\, i) & \cdots & W_{H^1 h^0}(q;\, i) & \cdots & W_{H^1 H^0}(q;\, i) \end{bmatrix}. \tag{29}$$

With these expansions, the scalar form of Eqs. (6)–(9) are written as follows:

$$net Y_{h^2}(n) = \sum_j \sum_{h^1} V_{h^2 h^1}(n;\, j) Z_{h^1}(n\beta + j)\,, \tag{30}$$

$$Y_{h^2}(n) = a(net Y_{h^2}(n))\,, \tag{31}$$

$$net Z_{h^1}(n) = \sum_i \sum_{h^0} W_{h^1 h^0}(q;\, i) X_{h^0}(q\alpha + i)\,, \tag{32}$$

$$Z_{h^1}(q) = a(net Z_{h^1}(q))\,, \tag{33}$$

$$O_{h^2} = \frac{1}{N} \textstyle\sum_n Y_{h^2}(n)\,, \tag{34}$$

$$E = \frac{1}{2} \sum_{h^2} (d_{h^2} - O_{h^2})^2\,. \tag{35}$$

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:

$$\frac{\partial E}{\partial V_{h^2h^1}(n;\, j)} = \frac{\partial E}{\partial O_{h^2}} \frac{\partial O_{h^2}}{\partial Y_{h^2}(n)} \frac{\partial Y_{h^2}(n)}{\partial netY_{h^2}(n)} \times \frac{\partial netY_{h^2}(n)}{\partial V_{h^2h^1}(n)} \,. \tag{36}$$

Let

$$\delta_{Y_{h^2}}(n) \equiv -\frac{\partial E}{\partial O_{h^2}} \frac{\partial O_{h^2}}{\partial Y_{h^2}(n)} \frac{\partial Y_{h^2}(n)}{\partial netY_{h^2}(n)} = (d_{h^2} - O_{h^2}) \frac{1}{N} a'(netY_{h^2}(n)) \,, \tag{37}$$

and

$$\frac{\partial netY_{h^2}(n)}{\partial V_{h^2h^1}(n)} = Z_{h^1}(n\beta + j) \,. \tag{38}$$

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

$$\Delta V_{h^2h^1}(n;\, j) = -\eta \frac{\partial E}{\partial V_{h^2h^1}(n;\, j)} = \eta \delta_{Y_{h^2}}(n) Z_{h^1}(n\beta + j) \,. \tag{39}$$

By assembling $\delta_{Y_{h^2}}(n)$ in Eq. (37) for all h^2, we get,

$$\begin{aligned} \Delta_{\boldsymbol{Y}(n)} &= \begin{bmatrix} \delta_{Y_1}(n) \\ \vdots \\ \delta_{Y_{h^2}}(n) \\ \vdots \\ \delta_{Y_{H^2}}(n) \end{bmatrix} \\ &= \frac{1}{N} \begin{bmatrix} d_1 - O_1 \\ \vdots \\ d_{h^2} - O_{h^2} \\ \vdots \\ d_{H^2} - O_{H^2} \end{bmatrix} \circ \begin{bmatrix} a'(netY_1(n)) \\ \vdots \\ a'(netY_{h^2}(n)) \\ \vdots \\ a'(netY_{H^2}(n)) \end{bmatrix} \\ &= \frac{1}{N} (d - O) \circ a'(netY(n)) \,, \end{aligned} \tag{40}$$

where $\circ$ is the operator of one-by-one array multiplication defined in Eq. (5). Thus, the synapse adjustment in the output layer can be obtained as:

$$\Delta V(n;\, j) = \eta \Delta_{\boldsymbol{Y}(n)} \cdot \boldsymbol{Z}(n\beta + j)^T \,. \tag{41}$$

For the hidden layer, we have,

$$\frac{\partial E}{\partial W_{h^1h^0}(q;\, i)} = \sum_{h^2} \frac{\partial E}{\partial O_{h^2}} \sum_{(n,j)\in\varphi} \frac{\partial O_{h^2}}{\partial Y_{h^2}(n)} \times \frac{\partial Y_{h^2}(n)}{\partial netY_{h^2}(n)} \frac{\partial netY_{h^2}(n)}{\partial Z_{h^1}(n\beta + j)} \times \frac{\partial Z_{h^1}(n\beta + j)}{\partial W_{h^1h^0}(q;\, i)} \,. \tag{42}$$

Using the definition of $\delta_{Y_{h^2}}(n)$ in Eq. (37) and changing the order of summation, Eq. (42) is reduced to:

$$\frac{\partial E}{\partial W_{h^1h^0}(q;\, i)} = -\sum_{(n,j)\in\varphi} \sum_{h^2} \delta_{Y_{h^2}}(n) \times \frac{\partial netY_{h^2}(n)}{\partial Z_{h^1}(n\beta + j)} \times \frac{\partial Z_{h^1}(n\beta + j)}{\partial W_{h^1h^0}(q;\, i)} \,, \tag{43}$$

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

$$\frac{\partial netY_{h^2}(n)}{\partial Z_{h^1}(n\beta + j)} = V_{h^2h^1}(n;\, j) \,, \tag{44}$$

and

$$\frac{\partial Z_{h^1}(n\beta + j)}{\partial W_{h^1h^0}(q;\, i)} = \frac{\partial Z_{h^1}(q)}{\partial W_{h^1h^0}(q;\, i)} = a'(netZ_{h^1}(q)) X_{h^0}(q\alpha + i) \,. \tag{45}$$

The final result of Eq. (42) is:

$$\frac{\partial E}{\partial W_{h^1h^0}(q;\, i)} = -\sum_{(n,j)\in\varphi} \sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2h^1}(n;\, j) \times a'(netZ_{h^1}(q)) X_{h^0}(q\alpha + i) = -\delta_{Z_{h^1}}(q) X_{h^0}(q\alpha + i) \,, \tag{46}$$

where

$$\delta_{Z_{h^1}}(q) = \sum_{(n,j)\in\varphi} \sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2h^1}(n; j) a'(netZ_{h^1}(q)) \,. \tag{47}$$

Collecting Eq. (47) for all h^1, we can get the error signals of the cell $\boldsymbol{Z}(q)$ as:

$$\Delta_{\boldsymbol{Z}(q)} = \begin{bmatrix} \delta_{Z_1}(q) \\ \vdots \\ \delta_{Z_{h^1}}(q) \\ \vdots \\ \delta_{Z_{H^1}}(q) \end{bmatrix}$$

$$= \sum_{(n,j)\in\varphi} \begin{bmatrix} \sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2 1}(n;\, j) \\ \vdots \\ \sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2 h^1}(n;\, j) \\ \vdots \\ \sum_{h^2} \delta_{Y_{h^2}}(n) V_{h^2 H^1}(n;\, j) \end{bmatrix}$$

$$\circ \begin{bmatrix} a'(netZ_1(q)) \\ \vdots \\ a'(netZ_{h^1}(q)) \\ \vdots \\ a'(netZ_{H^1}(q)) \end{bmatrix}$$

$$= \sum_{(n,j)\in\varphi} \boldsymbol{V}^T(n;\, j) \cdot \Delta_{\boldsymbol{Y}(n)} \circ a'(\boldsymbol{netZ}(q)) \,. \tag{48}$$

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

$$\Delta \boldsymbol{W}(q;\, i) = \eta \Delta_{\boldsymbol{Z}(q)} \cdot \boldsymbol{X}(q\alpha + i)^T \,. \tag{49}$$

References

1. A. Waibel, T. Hanazawa, G. Hinton, K. Shikano and K. J. Lang 1989, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoust., Speech, Signal Processing*, **37**(3), 328–338.
2. K. Fukushima, S. Miyake and T. Ito 1983, "Neocognitron: A neural network model for a mechanism of visual pattern recognition," *IEEE Trans. Syst., Man., Cybern.*, **SMC-13**(5), 826–834.
3. O. Matan, C. J. C. Burges, Y. LeCun and J. S. Denker 1992, "Multi-digit recognition using a space displacement neural network," in *Advances in Neural Information Processing System* **4**, 488–495 (Morgan Kaufmann, CA: San Mateo).
4. J. Keeler and D. E. Rumelhart 1992, "A self-organizing integrated segmentation and recognition neural net," in *Advances in Neural Information Processing System* **4**, 496–503 (Morgan Kaufmann, CA: San Mateo).
5. E. Sackinger, B. E. Boser, J. Bromley, Y. LeCun and L. D. Jackel 1992, "Application of the ANNA neural network chip to high-speed character recognition," *IEEE Trans. on Neural Networks* **3**(3), 498–505.
6. T. S. Huang and A. N. Netravali 1994, "Motion and structure from feature correspondences: A review," *Proceedings of the IEEE* **82**(2), 252–268.
7. K. Rohr 1994, "Toward model-based recognition of human movements in image sequences," *CVGIP: Image Understanding*, **59**(1), 94–115.
8. C. Cedras and M. Shah 1994, "A survey of motion analysis from moving light displays," *Proc. 1994 IEEE Conf. on Computer Vision and Pattern Recognition*, 214–221 (IEEE Comput. Soc. Press).
9. J. Yamato, J. Ohya and K. Ishii 1992, "Recognizing human action in time-sequential image using hidden markov model," *Proc. 1992 IEEE Conf. on Computer Vision and Pattern Recognition*, 379–385 (IEEE Comput. Soc. Press).
10. G. I. Chiou and J. N. Hwang 1994, "Image sequence classification using a neural network based active contour model and a hidden markov model," *Proc. ICIP-94*, 926–930 (IEEE Comput. Soc. Press).
11. C. Bregler, S. Manke, H. Hild and A. Waibel 1993, "Bimodal sensor integration on the example of 'speechreading'," *1993 IEEE International Conference on Neural Networks*, 667–671.
12. P. Duchnowski, M. Hunke, D. Busching, U. Meier and A. Waibel 1995, "Toward movement-invariant automatic lipreading and speech recognition," *1995 International Conference on Acoustics, Speech, and Signal Processing*, 109–112.
13. R. Polana and R. Nelson 1994, "Recognizing activities," *Proc. of ICPR*, 815–818 (IEEE Comp. Soc. Press).
14. R. Polana and R. Nelson 1994, "Detecting activities," *J. Visual Comm. Image Repres.* **5**(2), 172–180.
15. R. Polana and R. Nelson 1994, "Low-level recognition of human motion (or how to get your man without finding his body parts)," *Proceedings of the 1994 IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, 77–82 (IEEE Comput. Soc. Press).
16. S. Haykin 1994, *Neural Networks : A Comprehensive Foundation*, 498–515 (Macmillan College Publishing Company, Inc.).
17. E. A. Wan 1990, "Temporal backpropagation for FIR neural networks," *IEEE International Joint Conference on Neural Networks* **1**, 575–580.

18. E. A. Wan 1990, "Temporal backpropagation: An efficient algorithm for finite impulse response neural networks," in *Proceedings of the 1990 Connectionist Models Summer School*, eds. D. S. Touretzky, J. L. Elman, T. J. Sejnowski and G. E. Hinton (Morgan Kaufmann, CA: San Mateo), pp. 131–140.
19. A. Back, E. A. Wan, S. Lawrence and A. C. Tsoi 1994, "A unifying view of some training algorithms for multilayer perceptrons with FIR filter synapses," in *Proceedings of the 1994 IEEE Workshop*, pp. 146–54.
20. W. C. Lin 1996, "Bimodal speech recognition system," M.S. thesis, Dept. Control Eng., National Chiao-Tung Univ., Hsinchu, Taiwan.
21. E. Petajan, B. Bischoff, N. Brooke and D. Bodoff 0000, "An improved automatic lipreading system to enhance speech recognition," *CHI* **88**, 19–25.
22. K. Mase and A. Pentland 1989, "Lip reading: Automatic visual recognition of spoken words," *Image Understanding and Machine Vision 1989* **14**, 124–127, Technical Digest Series.
23. A. J. Goldschen 1993, "Continuous automatic speech recognition by lipreading," Ph.D. thesis, George Washington Univ.