Facial Expression Recognition Using Constructive Feedforward Neural Networks

Article *in* IEEE TRANSACTIONS ON CYBERNETICS · July 2004

DOI: 10.1109/TSMCB.2004.825930 · Source: PubMed

K. Khorasani
Concordia University Montreal

Facial Expression Recognition Using Constructive Feedforward Neural Networks

L. Ma and K. Khorasani

***Abstract*—A new technique for facial expression recognition is proposed, which uses the two-dimensional (2-D) discrete cosine transform (DCT) over the entire face image as a feature detector and a constructive one-hidden-layer feedforward neural network as a facial expression classifier. An input-side pruning technique, proposed previously by the authors, is also incorporated into the constructive learning process to reduce the network size without sacrificing the performance of the resulting network. The proposed technique is applied to a database consisting of images of 60 men, each having five facial expression images (neutral, smile, anger, sadness, and surprise). Images of 40 men are used for network training, and the remaining images of 20 men are used for generalization and testing. Confusion matrices calculated in both network training and generalization for four facial expressions (smile, anger, sadness, and surprise) are used to evaluate the performance of the trained network. It is demonstrated that the best recognition rates are 100% and 93.75% (without rejection), for the training and generalizing images, respectively. Furthermore, the input-side weights of the constructed network are reduced by approximately 30% using our pruning method. In comparison with the fixed structure back propagation-based recognition methods in the literature, the proposed technique constructs one-hidden-layer feedforward neural network with fewer number of hidden units and weights, while simultaneously provide improved generalization and recognition performance capabilities.**

***Index Terms*—Constructive neural networks, facial recognition, generalization, pruning strategies, two-dimensional (2-D) discrete cosine transform.**

I. Introduction

The computer-based recognition of facial expressions has been an active area of research in the literature for a long time. The ultimate goal in this research area is the realization of intelligent and transparent communications between human beings and machines. Several facial expression recognition methods have been proposed in the literature; see, for example, [4], [5], [12], and [27] and the references therein.

A well-known facial action coding system was developed by Ekman [5] for facial expression description. In facial action coding system, the face is divided into 44 action units, such as nose, mouth, eyes, etc. The movement of muscles of these feature-bearing action units are used to describe any human facial expression. This method requires a three-dimensional (3-D) measurement and may thus, be too complex for real-time processing. To overcome and remedy the drawbacks associated with the original facial action coding system, a modified system using only 17 relevant action units was proposed in [14] for facial expression analysis and synthesis. However, 3-D measurement is still needed. Although, the complexity of the modified facial action coding system is reduced when compared to the original system, certain information useful for facial expression recognition may be lost. In [6], a more accurate representation of human facial expressions (FAC+) is derived by using computer vision system to probabilistically characterize facial motion and muscle activation. In recent years facial expression recognition based on two-dimensional (2-D) digital images has received a lot of attention by researchers. In [24], a radial basis function neural network is proposed to recognize human facial expressions. The 2-D discrete cosine transform is used to compress the entire face image. The resulting lower-frequency 2-D discrete cosine transform coefficients are used to train a one-hidden-layer feedforward neural network in [27]. Very promising experimental results are also reported in [24]. A more detailed review on facial expression recognition can be found in [4].

The neural network-based recognition methods are found to be particularly promising [24], [27], since the neural networks can easily implement the mapping from the feature space of face images to the facial expression space. However, determining a proper network size has always been a frustrating and time consuming experience for neural network developers. This is generally dealt with through a series of long and costly trial-and-error simulations. Motivated by these limitations and drawbacks, in this paper, we propose to use a constructive feedforward neural network to overcome and remedy this problem. The constructive feedforward neural network can systematically determine a proper network size required by the complexity of a given problem, while reducing considerably the computational cost involved in network training when compared with the standard radial basis functions and back propagation-based training techniques. We are particularly interested in the constructive one-hidden-layer feedforward neural networks which are simple in structure and yield fairly good performances in many applications such as regression problems, image compression and facial expression recognition [18]–[20].

The organization of the remainder of this paper is as follows. In Section II, the main features of a constructive neural network are presented. A pruning technique is proposed and applied to our constructive neural network in Section II-C. In Section III, the application of our proposed constructive neural network to facial expression recognition is presented. Experimental results on a database consisting of images of 60 men, each having five facial expression images are also presented to demonstrate and illustrate the potential capabilities of our proposed technique. Conclusions are stated in Section IV.

II. Constructive Algorithms for Feedforward Neural Networks

Constructive learning alters the network structure as learning proceeds, producing automatically a network with an appropriate size. In this approach, one starts with an initial network of a "small" size, and then adds incrementally new hidden units and/or hidden layers until some prespecified error requirement is reached, or no performance improvement can be observed. The network obtained in this way is a "reasonably" sized one for the given problem at hand. Generally, a "minimal" or an "optimal" network size is seldom achieved by using this strategy, however a "subminimal/suboptimal" network can be expected [16], [17]. This problem has attracted a lot of attention by many researchers and several promising algorithms have been proposed in the literature. Kwok and Yeung in [16] surveys the major constructive algorithms in the literature. Dynamic node creation algorithm and its variants [1], [25], activity-based structure level adaptation [26], cascade-correlation algorithms [8], [23], and the constructive one-hidden-layer algorithms [15], [17] are among the most important constructive learning algorithms developed so far in the literature.

The major advantages of constructive algorithms over the other methods such as pruning algorithms [3], [21] and regularization-based techniques [2], [13] are as follows.

1) It is easier to specify the initial network architecture in constructive learning techniques, whereas in pruning algorithms

Manuscript received July 14, 2001; revised July 9, 2003. This work was supported in part by the Natural Science and Engineering Research Council of Canada (NSERC) under research Grant RGPIN-42515. This paper was recommended by Associate Editor V. Murino.

The authors are with the Department of Electrical and Computer Engineering, Concordia University, Montreal, H3G 1M8 QC, Canada (e-mail: liying@ece.concordia.ca; kash@ece.concordia.ca).

Digital Object Identifier 10.1109/TSMCB.2004.825930

one usually does not know *a priori* how large the original network should be.

2) Constructive algorithms tend to build small networks due to their incremental learning nature. A network is constructed that has a direct correspondence to the complexity of the given problem and the specified performance requirements, while excessive efforts may be rendered to trim the unnecessary weights in the network in pruning algorithms. Thus, constructive algorithms are generally more efficient (in terms of training time and network complexity/structure) than pruning algorithms.
3) In pruning algorithms and regularization-based techniques, one must specify or select several problem-dependent parameters in order to obtain an "acceptable" and "good" network yielding satisfactory performance results. This aspect could potentially reduce the applicability of these algorithms in real-life applications. On the other hand, constructive algorithms do not suffer from these limitations.

In the next section, we first give a simple formulation of the training problem for a constructive one-hidden-layer feedforward neural network in the context of a nonlinear optimization problem. The advantages and disadvantages of the constructive algorithms are also discussed.

A. Formulation of Constructive Feedforward Neural Network Training

Suppose a feedforward neural network is used to approximate a regression function whose input vector (or predictor variables) is indicated by the multidimensional vector $\mathbf{X}$ and without loss of any generality, its output (or response) is expressed by the scalar Y. A regression surface (input–output function) $g(\cdot)$ is used to describe the relationship between $\mathbf{X}$ and Y. A feedforward neural network is trained and used to realize or represent this relationship. The input samples are denoted by $(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^P)$, the output samples at each layer are denoted by $(\mathbf{y}_1^j, \ldots, \mathbf{y}_{l-1}^j, y_l^j), j = 1, \ldots, P$ and the corresponding target samples (or observations) are denoted by $(d^1, d^2, \ldots, d^P)$, which are the output data contaminated by an additive white noise vector $\mathbf{\Lambda} = (\epsilon^1, \epsilon^2, \ldots, \epsilon^P)$, where $l-1$ is the number of hidden layers, l denotes the output layer, and P is the number of patterns in the data set. The network training problem may be formulated as the following unconstrained least squares nonlinear optimization problem, shown in (1) and (2) at the bottom of the page, where $\mathbf{n} = (n_1, n_2, \ldots, n_{l-1})$ is a vector denoting the number of hidden units at each hidden layer, $\mathbf{f} = (f_1, f_2, \ldots, f_l)$ denotes the activation function of each layer, $f_1, f_2, \ldots, f_{l-1}$ are usually nonlinear activation functions, with f_l the activation function of the output layer selected to be linear for a regression problem, and $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_l$ are the weight matrices corresponding to each layer. The three levels of adaptation, that is, structure, functional, and learning levels are included in the above least squares nonlinear optimization problem. The performance index (1) is clearly too complicated to solve by existing optimization techniques. Fortunately, by fixing certain variables, the optimization problem will become tractable and easier to solve. However, only a suboptimal solution can then be obtained. As it turns out, a suboptimal solution suffices in many practical situations.

If on the other hand, a one-hidden-layer network is used to represent a mapping problem, its training is reduced to the following least-squares optimization problem, shown in (3) and (4) at the bottom of the page. It is not difficult to observe that even the above reduced least-squares nonlinear optimization problem is still not easy to solve. This is partly due to the freedom in selecting the activation functions and the number of hidden units that complicate the solution search space for the optimization problem.

Practically, one can solve (1) and (2) or (3) and (4), only through an incremental procedure. That is, by first fixing certain variables, say the activation functions and the number of hidden layers and units, and then by solving the resulting least-squares optimization problem with respect to the remaining variables. The process will be repeated until an acceptable solution or a network is obtained. The constructive feedforward neural network proposed in this paper provides also a suboptimal solution to (1)–(2) or (3)–(4).

B. Limitations of the Current Constructive Feedforward Neural Networks

Our motivation for applying a constructive learning algorithm as developed earlier by the authors in [19] and [20] may be justified due to the following rationale.

1) The one-hidden-layer feedforward neural network is simple and elegant in structure. The fan-in problem with the cascade correlation-type architectures is not present in this structure. Furthermore, as "deeper" the structure becomes, the more input-side connections for a new hidden unit will be required. This may give rise to degradation of generalization performance of the network, as some of the connections may become irrelevant to the prediction of the output.

$$\min_{l,\mathbf{n},\mathbf{f},\mathbf{w}} \sum_{j=1}^{P} \left(d^j - y_l^j\right)^2 \tag{1}$$

$$\text{subject to} \begin{cases} \mathbf{y}_1^j = f_1(\mathbf{w}_1 \mathbf{x}^j), & \mathbf{w}_1 \in \Re^{n_1 \times M}, \quad \mathbf{y}_1^j \in \Re^{n_1}, \quad \mathbf{x}^j \in \Re^M \\ \mathbf{y}_2^j = f_2\left(\mathbf{w}_2 \mathbf{y}_1^j\right), & \mathbf{w}_2 \in \Re^{n_2 \times n_1}, \quad \mathbf{y}_2^j \in \Re^{n_2} \\ \vdots & \\ y_l^j = f_l\left(\mathbf{w}_l \mathbf{y}_{l-1}^j\right), & \mathbf{w}_l \in \Re^{1 \times n_{l-1}}, \quad y_l^j \in \Re^1 \end{cases} \tag{2}$$

$$\min_{f_1, f_2, n_1, \mathbf{w}_1, \mathbf{w}_2} \sum_{j=1}^{P} \left(d^j - y_2^j\right)^2 \tag{3}$$

$$\text{subject to} \begin{cases} \mathbf{y}_1^j = f_1(\mathbf{w}_1 \mathbf{x}^j), & \mathbf{w}_1 \in \Re^{n_1 \times M}, \quad \mathbf{y}_1^j \in \Re^{n_1}, \quad \mathbf{x}^j \in \Re^M \\ y_2^j = f_2\left(\mathbf{w}_2 \mathbf{y}_1^j\right), & \mathbf{w}_2 \in \Re^{1 \times n_1}, \quad y_2^j \in \Re^1. \end{cases} \tag{4}$$

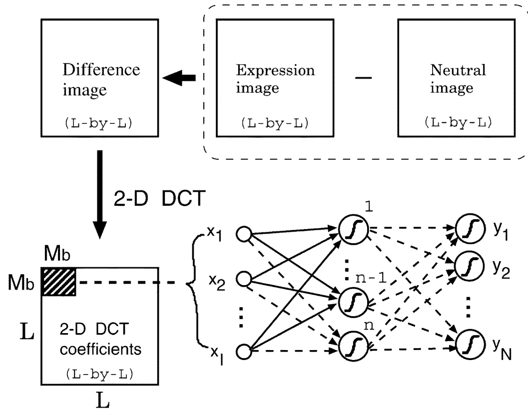

Fig. 1. Application of the constructive one-hidden-layer feedforward neural network to facial expression recognition.

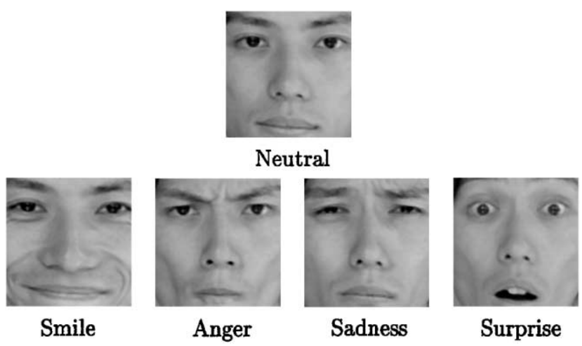

Fig. 2. Sample of nominal face images from the database.

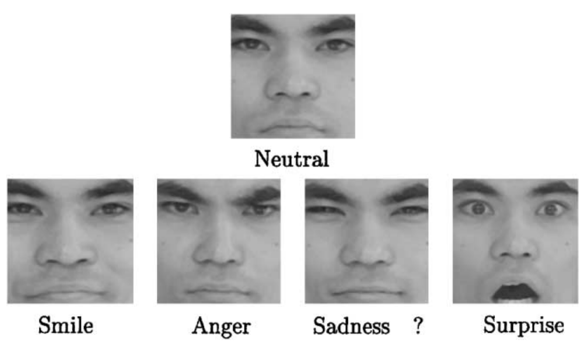

Fig. 3. Sample of face images from the database with the image registered as sadness being ambiguous.

2) The one-hidden-layer feedforward neural network is a universal approximator [10], [11]. Therefore, the convergence of constructive algorithms can be easily established [17].
3) The constructive learning process is simple and facilitates the investigation of training efficiency and development of other improved strategies.

The constructive feedforward neural network considered in this paper may yield improved approximation and representation capabilities as compared to fixed structure feedforward neural networks. Other architectures have also been developed in the literature, such as the stack learning algorithm [9] and adding-and-deleting algorithm [22]. The stack learning algorithm begins with a minimal structure consisting of input and output units only, similar to the initial network in cascade correlation algorithm. The algorithm then constructs a network by creating a new set of output units and by converting the previous output units into new hidden units. The new output layer has

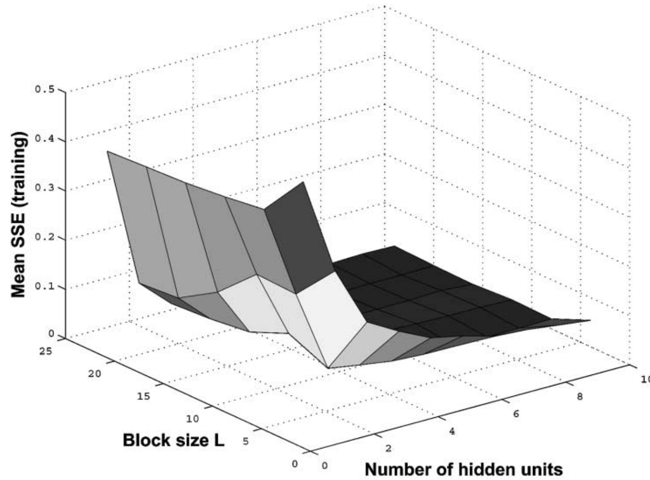

Fig. 4. Mean training SSEs versus the block size and the number of hidden units (training with pruning, 20 runs).

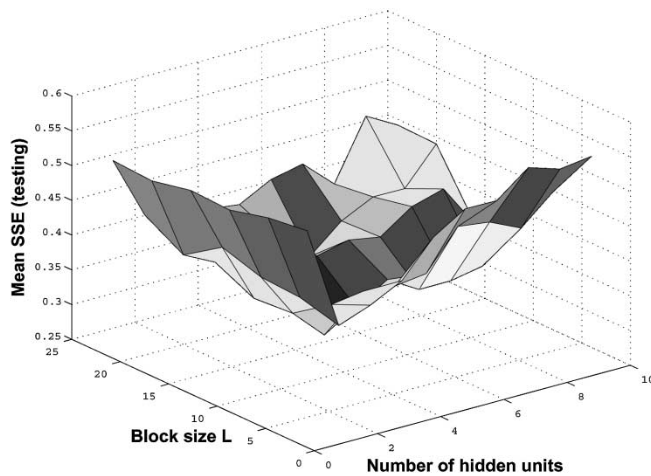

Fig. 5. Mean generalization SSEs versus the block size and the number of hidden units (training with pruning, 20 runs).

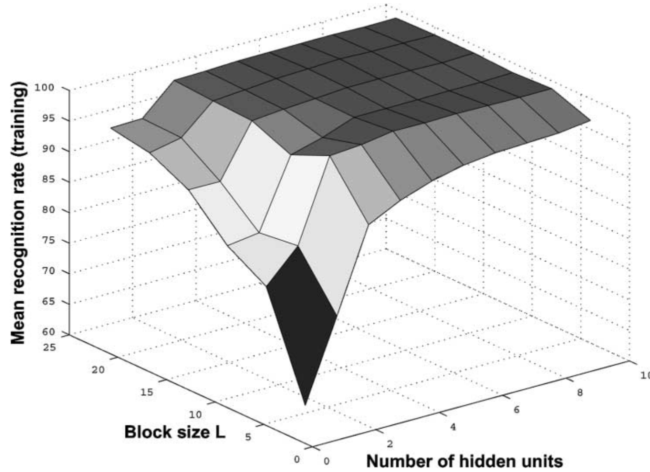

Fig. 6. Mean recognition rates versus the block size and the number of hidden units obtained during network training with pruning (20 runs).

connections to both the original input units and all the established hidden units. In other words, this algorithm generates a network that has a similar structure as in the cascade correlation-based networks, and hence has the same limitations as the cascade correlation. In the

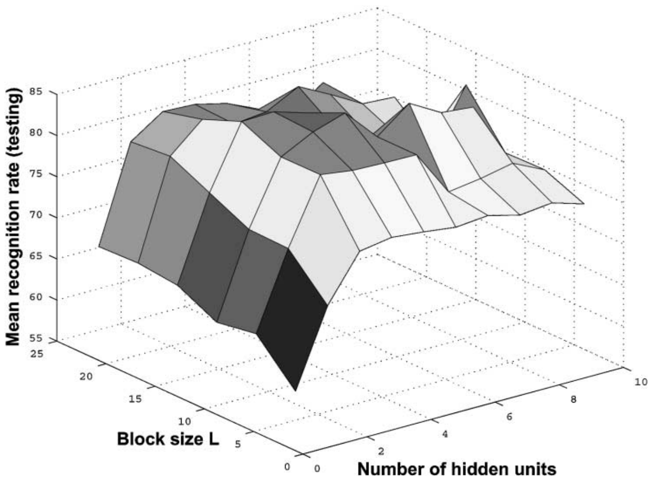

Fig. 7. Mean recognition rates versus the block size and the number of hidden units obtained during testing the networks trained with pruning (20 runs).

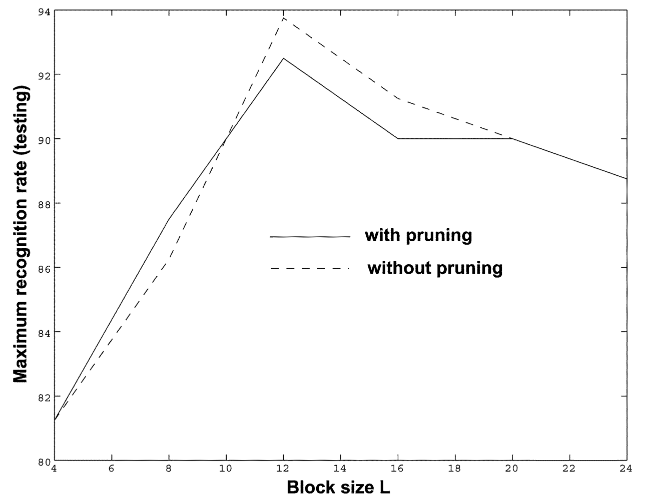

Fig. 9. Maximum recognition rates versus the block size obtained in testing for the networks trained with pruning and without pruning (20 runs).

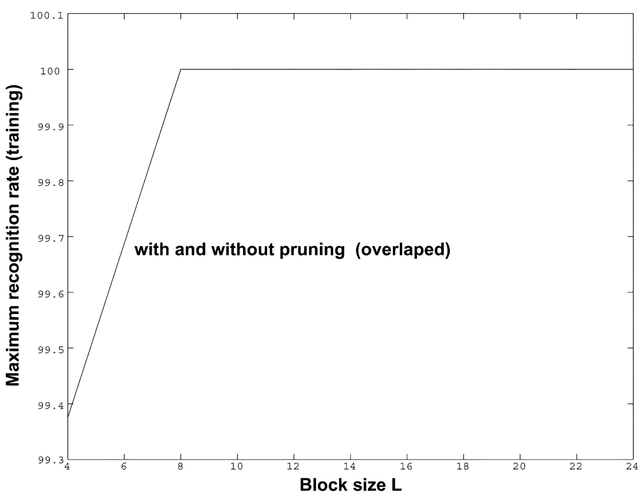

Fig. 8. Maximum recognition rates versus the block size obtained during network training with pruning and without pruning (20 runs).

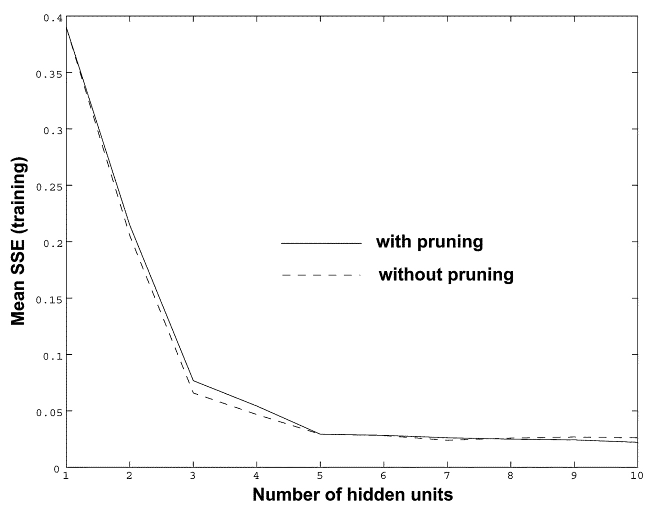

Fig. 10. Mean SSEs for training of the constructive feedforward one-hidden-layer-feedforward neural networks trained with pruning and without pruning ($M_b = 12$ and 20 runs).

adding-and-deleting algorithm, the network training is divided into two phases: addition phase and deletion phase. These two phases are controlled by evaluating the network performance. The so-called backtracking technique is used to avoid the "limit off" of the constructive learning. This algorithm may produce multilayered feedforward neural networks and it is computationally very intensive due to its lengthy addition-and-deletion process and the use of the BP-based training algorithm.

C. Input-Side Sensitivity-Based Pruning Strategy

In the input-side training, one can have one or a pool of candidates to train a new hidden unit. In the latter case, the neuron that results in the maximum objective function will be selected as the best candidate. This candidate is incorporated into the network and its input-side weights are frozen in the subsequent training process that follows. However, certain input-side weights may not contribute noticeably to the maximization of the objective function or indirectly to the reduction of the training error. These connections should first be detected and then removed through a pruning technique. Pruning these connections is expected to produce a smaller network without compromising the performance of the network. Note that the pruning operation is carried out "locally," and therefore, the generalization performance of the final network will not be improved significantly, since the conventional pruning-and-backfitting performed in standard fixed size network pruning is not implemented here. Below, we present a sensitivity function for the purpose of formalizing the input-side weight pruning process.

Suppose that the best candidate for the nth hidden unit to be added to the network results in an objective function $J_{\max,n}$. Then, sensitivity of each weight may be defined as follows:

$$S_{n,i} = J_{\max,n} - J_{\text{input}}(w_{n,i} = 0), \quad i = 1, \ldots, M \tag{5}$$

where $J_{\text{input}}(w_{n,i} = 0)$ is the value of the objective function when $w_{n,i}$ is set to zero, while other connections are unchanged. Note that the bias is usually not pruned. The above sensitivity function measures the contribution of each connection to the objective function. The largest value for the nth hidden unit sensitivity is denoted by $S_n^{\max}$. If $S_{n,i} \leq 0$, and/or is very small compared to $S_n^{\max}$, say 3% (pruning level $:= (S_n^{\max} - S_{n,i})/S_n^{\max}$) of it, then the weight $w_{n,i}$ is removed. After the

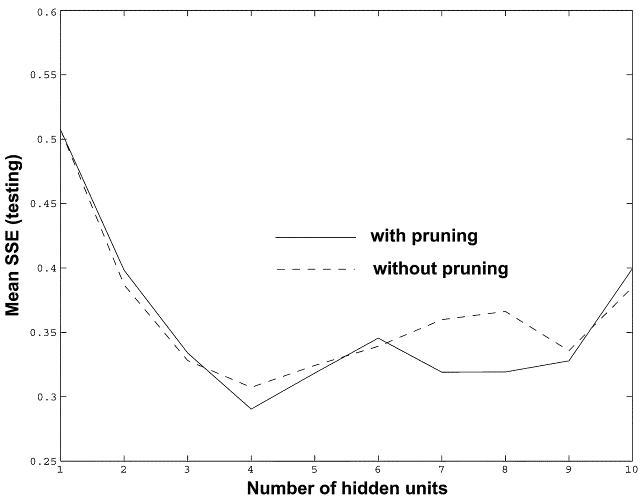

Fig. 11. Mean SSEs for generalization of the constructive one-hidden-layer-feedforward neural networks trained with pruning and without pruning ($M_b = 12$ and 20 runs).

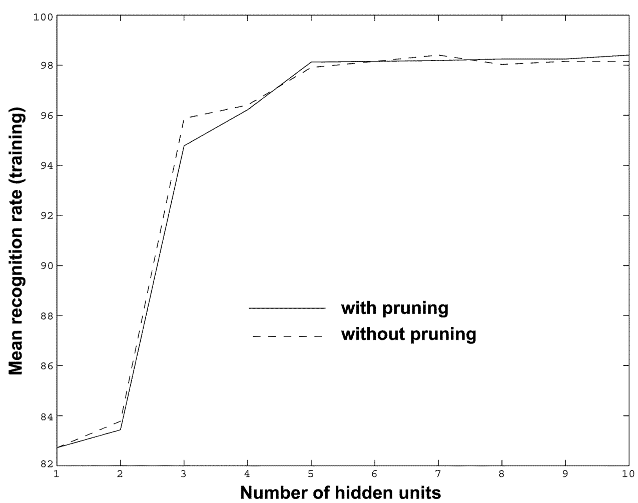

Fig. 12. Mean recognition rates for the constructive one-hidden-layer feedforward neural networks obtained during network training with pruning and without pruning ($M_b = 12$ and 20 runs).

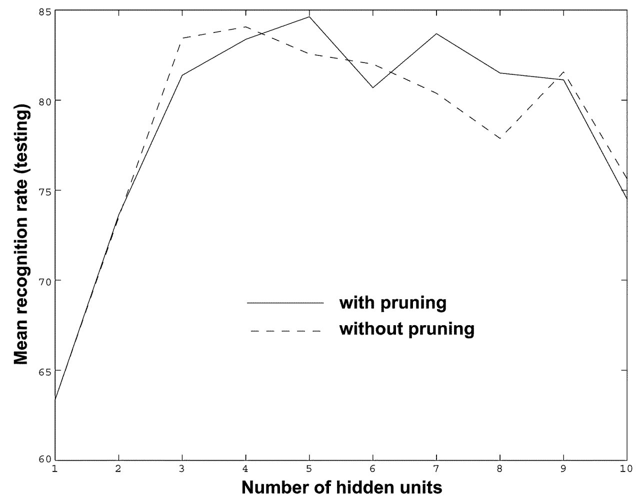

Fig. 13. Mean recognition rates for the constructive one-hidden-layer feedforward neural networks obtained for testing of the networks trained with pruning and without pruning ($M_b = 12$ and 20 runs).

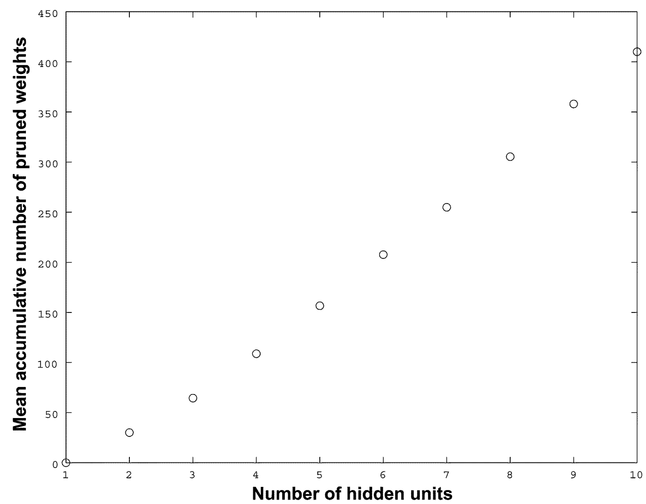

Fig. 14. Mean accumulative number of pruned input-side weights for the constructive one-hidden-layer feedforward neural networks with pruning ($M_b = 12$ and 20 runs).

pruning is performed, the output of the hidden unit $f(s_n^j)$ is reevaluated and the output-side training is performed one more time.

III. Application to Facial Expression Recognition

A. Theoretical Background

Fig. 1 describes the procedure in the application of the constructive one-hidden-layer feedforward neural network to the facial expression recognition problem. To recognize the facial expressions from a 2-D human face images, one generally needs to establish a feature detector that can capture the dominant characteristics of the face images, and a classifier that can categorize the facial expressions of interest. The features detected for each facial expression must be insensitive and not be influenced by the appearance of any individual human. Therefore, some preprocessing of the face images is generally needed. One may first obtain a difference image by subtracting a neutral image from a given expression image. The difference images are expected to have much less to do with the appearance of the human whose facial expressions are the subject of recognition. However, it is still very difficult for a classifier to recognize the facial expression from the difference images, as the difference image still has a large amount of data. To facilitate the recognition process, one needs to further compress the difference image in order to reduce the size of the data without sacrificing key attributes and features that play fundamental role in the recognition success. The 2-D discrete cosine transform (DCT) is frequently used in image compression as one viable tool for this purpose. The 2-D DCT can reduce the size of the data significantly by transforming an image from a spatial representation into the frequency domain where, in general, the lower frequencies are characterized by relatively large amplitudes while the higher frequencies have much smaller magnitudes. In other words, the higher frequency components can be ignored without significantly compromising the key characteristics of the original difference image, as far as the facial expression recognition problem is concerned. It is therefore argued that the 2-D DCT coefficients of the lower frequency modes, in principle, capture the most dominant and relevant information of the facial expressions.

A square (or block) of the lower frequency 2-D DCT coefficients is rearranged as an input vector $\mathbf{x}$ of dimension M_b^2 fed to a constructive one-hidden-layer feedforward neural network. The input-side training is performed by maximizing a correlation objective function based on the *quickprop* algorithm [7], [19], [20].

Output-side training is performed by using a Quasi-Newton algorithm due to the nonlinearity of the sigmoidal activation function of the output nodes. The pruning method in Section II-C is used during the network training to reduce the network size. The adaptively constructive trained one-hidden-layer is evaluated by not only the mean summed squared error (SSE), but also the recognition rate subject to these facial expression images that are used during training. The remaining images that are not presented to the network during training will be used to test the generalization capability of the trained network. Furthermore, the confusion matrix that is commonly used in pattern recognition, is also utilized here to assess the ability of the trained network to separate the four facial expressions being considered. The decision to select a particular facial expression category at the output of the network is achieved by the so-called winner-take-all policy. That is, a given image will be classified to the category whose corresponding output node yields the maximum output value.

B. Experimental Results

The constructive one-hidden-layer feedforward neural network discussed in previous section is applied to a database that consists of images of 60 men, each having five face images (neutral, smile, anger, sadness, and surprise). The database is already normalized. In the normalization process, the centers of the eyes and mouth are taken as the reference points, and two lines one connecting the centers of the eyes (line A), and the other starting from the center of the mouth and ending at the middle point of line A (line B) are considered. An affine transformation is used such that these two lines are orthogonal to each other in all the images. Furthermore, the length of line B is set to a prespecified constant value for all the images. All the images in this database are of size 128×128 having 256 gray levels (bite rate $= 8$ bits/pixel). Smile, anger, sadness, and surprise are the four specific facial expressions of interest. In the simulation experiments, the images of 40 men are used for network training, and the remaining 20 images are used for generalization and testing. Fig. 2 shows a set of sample of face images corresponding to the same man. The facial expression of each face image in this sample is quite clear to a human vision. These face images are used in network training. For comparison, in Fig. 3 we provide a sample of face images for another man. One can observe that the facial expression of the fourth image registered as "sadness" is not trivial even for a human to recognize.

Through numerous simulations it was determined that for network training and testing the block size (M_b) of the square of the lower frequency 2-D discrete cosine transform coefficients has a strong influence on the neural network performance. Therefore, experiments were conducted with different block size M_b. For each M_b, 20 runs with different initial weights were conducted to construct 20 one-hidden-layer feedforward neural networks that each has a maximum of ten hidden units (the number of ten hidden units was selected to simply demonstrate how the network performance is impacted as one changes the number of hidden units from 1 to 10). The networks performance were evaluated first by the images used during network training, and then by the remaining images that are not seen by the trained networks.

First, four figures (refer to Figs. 4–7) for the mean SSEs of training with pruning, the mean SSEs of generalization with pruning, the mean recognition rate for network training, and the mean recognition rate for testing with pruning are presented, respectively, as a function of the block size M_b and the number of hidden units. Similar results are also obtained for the 20 one-hidden-layer feedforward neural networks trained without pruning. Clearly, from Figs. 4 and 6, one can observe that our proposed technique performs quite satisfactorily in terms of the training SSEs for all the selected block sizes, and the training effectiveness saturates when more than three hidden units are added to the network. Figs. 5 and 7 indicate that networks with less than two and more than six hidden units will result in poor performance in terms of both generalization SSEs and recognition rates.

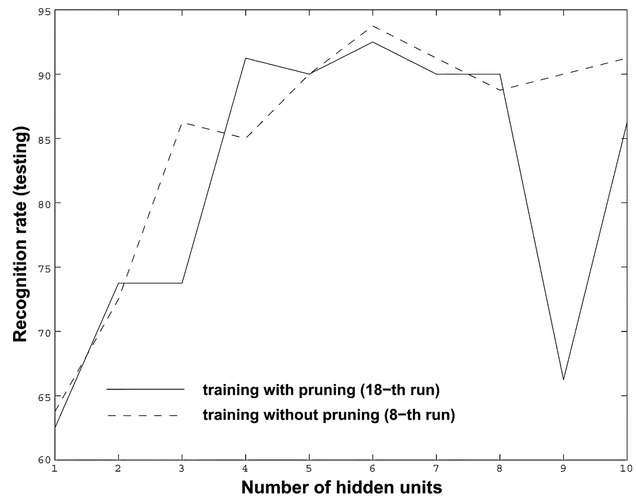

Fig. 15. Recognition rates versus the number of hidden units for two constructive one-hidden-layer feedforward neural networks yielding the best recognition rates in testing stage. These two networks are obtained in the 18th and 8th runs of network training with and without pruning, respectively ($M_b = 12$).

Next, we have selected the best recognition rates as far as network training and testing are concerned. The purpose here is to decide the most appropriate block size that leads to the highest recognition rates during testing. The results are plotted in Figs. 8 and 9. It follows that in this database for perfect training the block size needs to be equal to or greater than eight, while for generalizing and testing the block size of 12 yields the best results. These are based on the 20 runs of network training with and without pruning (40 one-hidden-layer feedforward neural networks). The best recognition rates obtained during the training and the testing stages with and without pruning are achieved in the 18th and 8th runs, respectively. These results will clearly vary if one increases the number of runs.

Finally, we take a closer look at the performance of the constructive one-hidden-layer feedforward neural network corresponding to the best block size selected above. In Figs. 10 and 11, we provide the mean SSE for training and generalization for the block size $M_b = 12$ which has resulted in the highest recognition rates in the generalizing stage. The mean recognition rates during training and testing are also plotted in Figs. 12 and 13, respectively. The mean accumulative number of pruned input-side weights is shown in Fig. 14. The recognition rates as a function of the number of hidden units for the two best one-hidden-layer feedforward neural networks are provided in Fig. 15, with one network trained with pruning and the other network trained without pruning. The confusion matrices corresponding to the training and the testing are given in Tables I and II for these two networks having six hidden units.

For sake of comparison with two other methods that are available in the literature, namely the Vector Matching algorithm in [12] and the fixed-size NN algorithm in [27], simulation results are provided in Tables III and IV. A comparative summary of the performance of our proposed algorithm and the above two algorithms are given in Table V.

From the above representative experimental results the following comments are now in order.

TABLE I
CONFUSION MATRICES OBTAINED BY ONE-HIDDEN-LAYER FEEDFORWARD NEURAL NETWORK WITH SIX HIDDEN UNITS FOR THE IMAGES USED DURING NETWORK TRAINING (LEFT TABLE) WITH PRUNING AND (RIGHT TABLE) WITHOUT PRUNING ($M_b = 12$)

	smile	anger	sadness	surprise
smile	40	0	0	0
anger	0	40	0	0
sadness	0	1	39	0
surprise	0	0	0	40
mean recognition rate 99.375%				

	smile	anger	sadness	surprise
smile	40	0	0	0
anger	0	40	0	0
sadness	0	0	40	0
surprise	0	0	0	40
mean recognition rate 100.00%				

TABLE II
CONFUSION MATRICES OBTAINED BY ONE-HIDDEN-LAYER FEEDFORWARD NEURAL NETWORK WITH 6 HIDDEN UNITS FOR THE IMAGES NOT SEEN BY THE TRAINED NETWORK (LEFT TABLE) WITH PRUNING AND (RIGHT TABLE) WITHOUT PRUNING ($M_b = 12$)

	smile	anger	sadness	surprise
smile	20	0	0	0
anger	1	17	2	0
sadness	0	3	17	0
surprise	0	0	0	20
mean recognition rate 92.50 %				

	smile	anger	sadness	surprise
smile	20	0	0	0
anger	0	19	1	0
sadness	1	3	16	0
surprise	0	0	0	20
mean recognition rate 93.75 %				

TABLE III
CONFUSION MATRICES BY VECTOR MATCHING (LEFT TABLE IS FOR TRAINING AND RIGHT TABLE IS FOR TESTING)

	smile	anger	sadness	surprise
smile	40	0	0	0
anger	2	33	5	0
sadness	1	10	28	1
surprise	0	1	1	38
Mean recognition rate =86.875%				

	smile	anger	sadness	surprise
smile	19	1	0	0
anger	1	16	3	0
sadness	1	5	15	0
surprise	0	0	0	20
Mean recognition rate =86.25%				

TABLE IV
CONFUSION MATRICES BY FIXED-SIZE NN (LEFT TABLE IS FOR TRAINING AND RIGHT TABLE IS FOR TESTING)

	smile	anger	sadness	surprise
smile	40	0	0	0
anger	0	40	0	0
sadness	0	0	40	0
surprise	0	0	0	40
Mean recognition rate =100%				

	smile	anger	sadness	surprise
smile	17	2	1	1
anger	0	19	0	1
sadness	0	2	18	0
surprise	0	0	0	20
Mean recognition rate =92.50%				

TABLE V
COMPARISON AMONG THE BEST RECOGNITION RESULTS OBTAINED BY THREE DIFFERENT RECOGNITION METHODS

Method	Block size $M_b \times M_b$	Hidden unit number	Training Recognition Rate (%)					Testing Recognition Rate (%)				
			smi	ang	sad	sur	mean	smi	ang	sad	sur	mean
Vector Matching [12]	7×7	—	100	82.50	70.00	95.00	**86.875**	95.00	80.00	70.00	100	**86.25**
Fixed-size NN [27]	16×16	25	100	100	100	100	**100**	85.00	95.00	90.00	100	**92.50**
Constructive NN	12×12	6	100	100	100	100	**100**	100	95.00	80.00	100	**93.75**

1) From Figs. 4–9, it can be concluded that constructive one-hidden-layer feedforward neural networks trained with and without pruning are capable of representing the training sample images considerably well, and recognizing the new facial expression images in surprisingly high recognition rates, as long as the block size M_b is properly selected. It is conceivable that there exists an optimal block size which leads to the highest recognition rate during generalization of the constructive one-hidden-layer feedforward neural network. For the database used in this paper, the optimal block size is found to be approximately 12. A significantly smaller or a larger block size will result in poor recognition performance.
2) It can be seen from Figs. 8–13, and 15 that our proposed network training with and without pruning results in very close training, generalization SSEs and recognition rates performances. However, by invoking pruning the number of input-side weights is reduced by approximately 30%, resulting in a much smaller network. The one-hidden-layer feedforward neural networks with four to eight hidden units are found to have sufficient computational capabilities to represent the mapping from the feature space to the facial expression space of images.
3) Tables I and II demonstrate that the confusion matrices corresponding to expressions "anger" and "sadness" clearly emphasize the challenges that the facial expression recognition system

is confronted with as compared to the recognition of expressions "smile" and "surprise."

4) In comparison with the backpropagation-based recognition algorithm shown in [27], the constructive technique proposed here generates one-hidden-layer feedforward neural networks with significantly fewer number of hidden units and reduced number of input-side weights, while simultaneously resulting in improved recognition performances. A one-hidden-layer feedforward neural network obtained with block size $M_b = 12$ and a maximum of 6 hidden units yields a recognition rate that is as high as 93.75% (i.e., 75 expression images being correctly recognized). This result is better than the best result of 94.7% provided by the backpropagation-based networks with block sizes of $M_b = 16$ and 25 hidden units but which is subject to a rejection rate of 5% (as reported in [27]), with actually 72 expression images being correctly recognized. Vector matching method [12] is very simple, but its recognition rate (86.25%), as shown in Table III, is much lower than the rates of our proposed algorithm at (93.75%) and the fixed-size backpropagation-based algorithm [27] at (92.50%), as shown in Table IV. These results are summarized in Table V for comparison. Based on the above experimental results, we have also found that the training time of our proposed algorithm is always lower than the backpropagation-based algorithm when implemented and coded in Matlab environment.

IV. Conclusion

In this paper, the application of an adaptive constructive one-hidden-layer feedforward neural network to facial expression recognition was considered. It was shown that the proposed constructive algorithm can produce one-hidden-layer feedforward neural networks with much reduced number of hidden units and input-side weights in comparison with the backpropagation-based neural network constructed in [27], while yielding an improved recognition rate. In all the experimental results presented, it was revealed that the input-side weight pruning technique proposed results in smaller networks, while simultaneously providing similar performances when compared to their fully connected network counterparts.

Acknowledgment

The authors would like to thank M. Oda, Ritsumeikan University [formerly with Advanced Telecommunications Research Institute International (ATR)], Kyoto, Japan, for providing them with the database used in this work.

References

[1] T. Ash, "Dynamic node creation in backpropagation networks," *Connection Sci.*, vol. 1, no. 4, pp. 365–375, 1989.

[2] Y. Chauvin, "A back-propagation algorithm with optimal use of hidden units," in *Advances in Neural Information Processing*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1990, vol. 2, pp. 642–649.

[3] Y. Le Gun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Advances in Neural Information Processing*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1990, vol. 2, pp. 598–605.

[4] G. Donato, M. S. Bartlett, J. C. Hager, P. Ekman, and T. J. Sejnowski, "Classifying facial actions," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 974–989, Oct. 1999.

[5] P. Ekman and W. Friesen, *Facial Action Coding System*. Palo Alto, CA: Consulting Psychologists Press , 1978.

[6] I. A. Essa and A. P. Pentland, "Coding, analysis, interpretation, and recognition of facial expression," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 757–763, July 1997.

[7] S. E. Fahlman, "An Empirical Study of Learning Speed in Back-Propagation Networks," Carnegie-Mellon Univ., Pittsburg, PA, Tech. Rep. CMU-CS-88-162, 1988.

[8] S. E. Fahlman and C. Lebiere, "The Cascade-Correlation Learning Architecture," Carnegie Mellon Univ., Tech. Rep. CMU-CS-90-100, 1991.

[9] W. Fang and R. C. Lacher, "Network complexity and learning efficiency of constructive learning algorithms," in *Proc. Int. Conf. Artificial Neural Networks*, 1994, pp. 366–369.

[10] G. Gybenko, "Approximation by superposition of sigmoidal functions," *Math. Contr., Signals Syst.*, vol. 2, no. 4, pp. 303–314, 1989.

[11] D. Husmeier, *Neural Networks for Conditional Probability Estimation, Perspectives in Neural Computation*. New York: Springer-Verlag, 1999.

[12] Y. Inada, Y. Xiao, and M. Oda, "Facial Expression Recognition Using Vector Matching of Special Frequency Components,", IEICE Tech. Rep. DSP2001, Oct. 2001.

[13] M. Ishikawa, "A Structural Learning Algorithm With Forgetting of Link Weights," Electrotechnical Lab., Tsukuba, Japan, Tech. Rep. TR-90-7, 1990.

[14] F. Kawakami, H. Yamada, S. Morishima, and H. Harashima, "Construction and psychological evaluation of 3-d emotion space," *Biomed. Fuzzy Human Sci.*, vol. 1, no. 1, pp. 33–42, 1995.

[15] T. Y. Kwok and D. Y. Yeung, "Bayesian regularization in constructive neural networks," in *Proc. Int. Conf. Artificial Neural Networks*, Bochum, Germany, 1996, pp. 557–562.

[16] ——, "Constructive algorithms for structure learning in feedforward neural networks for regression problems," *IEEE Trans., Neural Networks*, vol. 8, pp. 630–645, May 1997.

[17] ——, "Objective functions for training new hidden units in constructive neural networks," *IEEE Trans., Neural Networks*, vol. 8, pp. 1131–1148, Sept. 1997.

[18] L. Ma and K. Khorasani, "New pruning techniques for constructive neural networks with application to image compression," *Proc. SPIE*, vol. 4055, pp. 298–308, 2000.

[19] ——, "Facial expression recognition using constructive neural networks," *Proc. SPIE*, vol. 4380, pp. 521–530, 2001.

[20] ——, "Application of adaptive constructive neural networks to image compression," *IEEE Trans. Neural Networks*, vol. 13, pp. 1112–1126, Sept. 2002.

[21] M. C. Mozer and P. Smolensky, "Skeletonization: A technique for trimming the fat from a network via relevance assessment," in *Advances in Neural Information Processing*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1989, vol. 1, pp. 107–115.

[22] T. M. Nabhan and A. Y. Zomaya, "Toward generating neural network structures for function approximation," *Neural Networks*, vol. 7, no. 1, pp. 89–99, 1994.

[23] L. Prechelt, "Investigation of the cascor family of learning algorithms," *Neural Networks*, vol. 10, no. 5, pp. 885–896, 1997.

[24] M. Rosenblum, Y. Yacoob, and L. S. Davis, "Human expression recognition from motion using a radial basis function network architecture," *IEEE Trans. Neural Networks*, vol. 7, pp. 1121–1138, Sept. 1996.

[25] R. Setiono and L. C. K. Hui, "Use of a quasinewton method in a feedforward neural network construction algorithm," *IEEE Trans. Neural Networks*, vol. 6, pp. 273–277, Nov. 1995.

[26] W. Weng and K. Khorasani, "An adaptive structural neural network with application to eeg automatic seizure detection," *Neural Networks*, vol. 9, no. 7, pp. 1223–1240, 1996.

[27] Y. Xiao, N. P. Chandrasiri, Y. Tadokoro, and M. Oda, "Recognition of facial expressions using 2-d dct and neural network," *Electron. Commun. Jpn.*, vol. 82, no. 7, pp. 1–11, 1999.