DIMENSIONALITY REDUCTION IN HYPERSPECTRAL IMAGE CLASSIFICATION

Huiwen Zeng and H. J. Trussell

Dept. of Electrical and Computer Engineering
North Carolina State University
Raleigh, NC 27695-7911

ABSTRACT

Hyperspectral images provide a vast amount of information about a scene. However, much of that information is redundant as the bands are highly correlated. For computational and data compression reasons, it is desired to reduce the dimensionality of the data set while maintaining good performance in image analysis tasks. This work presents a method of dimensionality reduction based on neural networks. A novel penalty function is presented and shown to successfully reduce the number of active neurons, which corresponds to the dimensionality of the data for the task of interest.

1. INTRODUCTION

Hyperspectral images, which have 30 to over 200 bands of data, provide a huge amount of information about the scene under investigation. This information allows improved detection and characterization of objects within that scene. However, for efficient computation, data storage and transmission, the amount of data must be reduced. Principal component analysis (PCA), using the Karhunen-Loeve transformation is a common method [1]. While PCA gives optimal dimensionality reduction while maintaining fidelity of the image in a mean square error sense, it is not optimal with respect to any particular image analysis task, such as target detection or classification. Other dimension reduction techniques include Isomap [2], Multidimensional Scaling (MDS) [3] and clustering algorithms [4].

Neural networks have been used frequently for detection and classification problems. For the two-class hyperspectral classification problem, only one output neuron will be needed, and the network can be represented by (assume only one hidden layer is used)

$$y = g\left(f(\boldsymbol{x}^T\mathbf{W}_I + \boldsymbol{b}_I)\,\boldsymbol{w}_O + b_O\right), \qquad (1)$$

where $\boldsymbol{x}$ and y are the input vector and output value of the ANN (Artificial Neural Networks). The weight matrix and biases connecting the input with the neurons on the hidden layer are denoted as $\mathbf{W}_I$ and $\boldsymbol{b}_I$, respectively, where the output weights, weights connecting hidden neurons with output neuron, are represented by vector $\boldsymbol{w}_O$. The variable b_O is the bias for the output neuron. Common transfer functions f and g include linear, tangent sigmoid and logarithm sigmoid functions.

The weighted sum of the input vector (hyperspectral image) that is the input value to a hidden layer neuron can be considered a projection of the image onto a one-dimensional space defined by the weight vector. The number of neurons required to perform the task of interest gives the dimension of the data required for this task. Reducing the number of neurons is equivalent to reducing the dimensionality of the problem. Several researchers claim that smaller networks provide greater generalization capabilities [5][6][7]. There are many methods in the literature for pruning the network and reducing the number of hidden neurons such as penalty term methods which add a complexity penalty term to the performance measure [5][8][9]. The gain competition technique prunes neural net by having hidden neurons' gains compete according to similarities between nodes [7]. Other pruning algorithms include two-stage procedure [10] and iterative pruning [11].

2. PRUNING AND DIMENSIONALITY REDUCTION

The basis of many pruning methods is penalty function. This is added to the error function that is minimized by the neural network optimization methods.

$$C(\boldsymbol{w}) = C_s(\boldsymbol{w}) + \lambda C_p(\boldsymbol{w}) \qquad (2)$$

The first term, $C_s(\boldsymbol{w})$, is the error measurement which depends on both the network and the input data and the second term $C_p(\boldsymbol{w})$ is the penalty function which depends on the network itself alone. The parameter λ adjusts the weight of the penalty term in the total cost function.

0-7803-8554-3/04/$20.00 ©2004 IEEE.

Common penalty functions include weight decay which reduces the ANN by minimizing the square of the 2-norm of the weights [5][12].

$$C_p(\boldsymbol{w}) = \sum_{i \in net} w_i^2 \ , \tag{3}$$

where w_i is all the synaptic weights in the network.

Another widely used complexity penalty is weight elimination [8].

$$C_p(\boldsymbol{w}) = \sum_{i \in net} \frac{\left(w_i / w_0\right)^2}{1 + \left(w_i / w_0\right)^2} \ , \tag{4}$$

where w_0 is the preassigned parameter, when $| w_i | << w_0$, the cost of the penalty function will approach zero, which means that particular weight is unimportant in the learning process.

However, our experience has shown that none of the penalty term methods perform consistently on the hyperspectral problem of interest here.

3. MIXED-NORM PENALTY FUNCTIONS

A novel penalty function called *mixed-norm penalty* is introduced in the paper, which minimizes the 1-norm of the output weights while keeping the 2-norm of them to be constant.

$$C_p(\boldsymbol{w}) = \|\boldsymbol{w}_O\|_1 + | \ \|\boldsymbol{w}_O\|_2 - 1 \ | \ , \tag{5}$$

where, $\boldsymbol{w}_O$ is the output weight vector as in (1).

Unlike weight decay and weight elimination which uses all the weights in the network, this *mixed-norm penalty* only depends on the output weights. When $w_O(i)$ (the weight connecting the i^{th} neuron on the hidden layer with the output neuron) approaches zero, the output of that particular neuron

$$f(\boldsymbol{x}^T \boldsymbol{w}_I + \boldsymbol{b}_I)\, w_O(i) \ , \tag{6}$$

will be small, regardless of the values of the weights between the input and the neurons on the hidden layer or the bias of that neuron. Because the number of output weights is usually much smaller than the number of input weights, especially for the hyperspectral classification problems, the *mixed-norm* penalty function in (5) considerably reduces the computation time.

Consider two $N \times 1$ normalized weight vectors

$$\left(1/\sqrt{N} \quad 1/\sqrt{N} \quad \cdots \quad 1/\sqrt{N} \quad 1/\sqrt{N}\right)^T$$

and

$$\left(0 \quad \cdots \quad 1 \quad \cdots \quad 0\right)^T \ ,$$

where N is the number of neurons on the hidden layer. The number of non-zero values in the output weight vector represents the actual dimension for solving the problem, i.e. the second vector is much more desirable than the first one in the sense of dimension reduction. Both of the vectors have unit 2-norm, however, the first one has larger 1-norm for $N > 1$, easy to see that minimizing the 1-norm of constant 2-norm weight vectors will reduce dimension.

Instead of solving the ANN problem using conventional backpropagation, we treat the problem as a general optimization problem. The error to minimize is the difference in the computed and desired output, plus the penalty function to reduce dimension. This allows the use of a great variety of solution methods.

Common complexity regularization approaches for pruning ANN minimize the summation of the total cost which is the summation of the performance measure and the complexity penalty as in (2). These approaches treat both $C_s(\boldsymbol{w})$ and $\lambda C_p(\boldsymbol{w})$ equally. For inappropriate values of λ, the performance of the classification will be poor which ruins the goal of low classification error. A more reasonable way of solving the problem is to put the error rate requirement as the constraint in the model while leaving (5) as the objective function as the optimization goal. Bi-level optimization method can be utilized to solve problems like this [13]. The general form of BLO (Bi-Level Optimization) is

$$\begin{aligned} &\text{minimize } f_2(x) \\ &\text{subject to } x \in \text{argmin } \{f_1(x)\} \end{aligned} \tag{7}$$

where f_1 and f_2 are two objectives, and f_1 is more important than f_2. For the hyperspectral image classification problem, f_1 will be the error measure, i.e. the dimension reduction will only be considered for those weight vectors which minimize the error of classification. As all the variables in (1) are continuous, there exists a continuous path of optimal solutions, i.e. all the points on the path minimizes the error measurement. Thus, equation (7) minimizes *mixed-norm penalty* along optimal path and low error rate is guaranteed.

4. RESULTS AND EXAMPLES

This approach is tested by training the network to learn the XOR mapping and classify target and clutter in a hyperspectral image.

The classical XOR function can be described mathematically in 2-space, examples with coordinates (0,0) and (1,1) belong to the same class and examples with

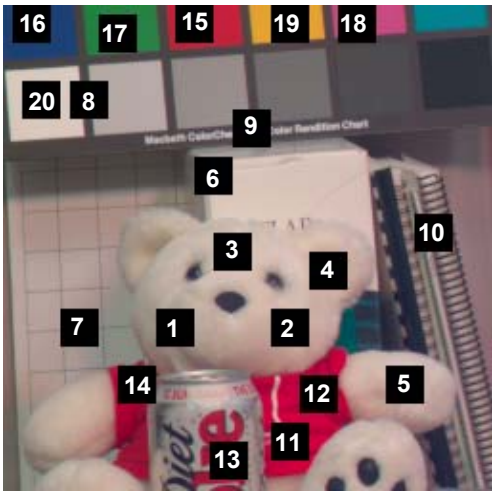

Figure 1 Original Image with Training Set

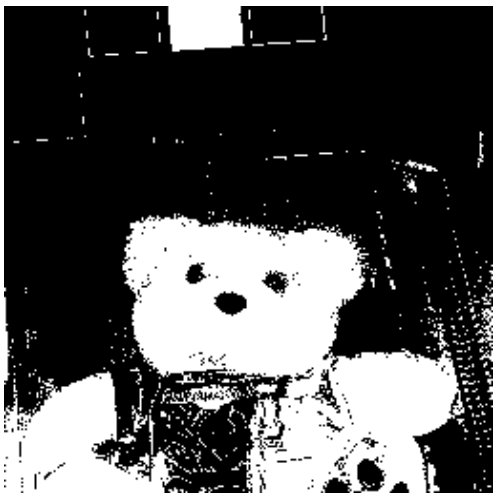

Figure 2 Results of Test 1 in Table 1 (mixed-norm)

coordinates (1,0) and (0,1) belong to another class. As is known, at least two hyperplanes, i.e. two neurons on the hidden layer, are needed to classify the XOR mapping.

Four neurons are used in the hidden layer for the XOR problem and 50 tests with different initial values of weights are performed. Out of the 50 tests we have run, 47 of them successfully reduced four neurons to two and 3 tests kept three significant weights. The average ratio of eliminated weights to the significant weights is 10^{-4}. Simulations with six neurons on the hidden layer also indicate that the dimension can be finally reduced to two.

Since true unclassified hyperspectral data is difficult to obtain, the 31-band image used for color reproduction was used [14]. A three band rendering (sRGB) of the image is shown with training sets indicated in Figure 1. In order to show the power of hyperspectral information, we declared the fur of the bear to be the target class and all the other objects to be the clutter. It seems reasonable to believe that since the fur is made of significantly different material than the other objects, that it would have a spectral characterization that would allow it to be distinguished from the paper white.

Six tests with different initial values are done on the hyperspectral image. Tests 1-3 are trained using only fur and white clutter, i.e. the input to the network are the 31 bands of all pixels in squares 1-10 in Figure 1. Tests 4-6 use fur, white clutter, as well as color pixels (squares 1-20 in Figure 1). In Figure 2, all the pixels detected as target are displayed in white, while all the black pixels are detected as clutter. Most of the error in Figure 2 appeared in the color region, such as the red jacket of the bear and a color checker chart at the top which indicates the lack of color information in training. Figure 3 shows the test error after color pixels are added, and the color pixels are detected correctly.

Figure 3 Results of Test 4 in Table 1 (mixed-norm)

Six neurons are used in the ANN for solving the hyperspectral separation problem and the output weights are compared in Table 1. Weights with magnitudes near zero ($<0.05w_{max}$, where w_{max} is the largest absolute value of the weight) can be removed from ANN. Successive neuron elimination tests have shown that one neuron will separate fur from white clutter while two neurons are need with colored clutter. All Tests except the last one correctly find the minimum dimension for optimal classification; and the last one also reduced the dimension considerably. We have run the method until it converges and the simulations indicate that the method is consistently successful but it may be slow to converge. Given sufficient iterations, we believe that it will converge to a very small dimension which equals or is close to the minimum. From Table 1, we can also see that weight elimination also reduces dimensionality to some extent,

Table 1
Output Weights of Six Tests on Hyperspectral Image

Test	Penalty Function	Error Rate	$\|w_O(1)\|$	$\|w_O(2)\|$	$\|w_O(3)\|$	$\|w_O(4)\|$	$\|w_O(5)\|$	$\|w_O(6)\|$
1	Mixed-Norm	0.0053	9.2e-8	2.6e-7	**0.9997**	8.6e-8	5.6e-7	5.4e-8
	Weight Elimination	0.0037	**40.772**	1.0379	1.7e-5	0.7996	1.0278	0.4894
2	Mixed-Norm	0.0045	9.3e-6	1.4e-5	2.5e-5	**0.9982**	5.2e-6	9.1e-6
	Weight Elimination	0.0063	**6.6924**	**3.6860**	1.0383	0.9904	**44.254**	**4.9768**
3	Mixed-Norm	0.0035	8.4e-6	1.2e-5	1.6e-6	1.2e-5	4.3e-5	**0.9971**
	Weight Elimination	0.0053	0.3718	1.0009	**23.506**	1.9e-6	0.2363	**140.65**
4	Mixed-Norm	0.0043	0.0275	**0.9371**	0.0011	0.0129	3.8e-5	**0.9658**
	Weight Elimination	0.0053	**0.8282**	2.2e-6	**12.444**	**1.0497**	0.2143	0.3780
5	Mixed-Norm	0.0031	**0.9191**	0.0111	0.0055	0.0074	**0.8064**	0.0058
	Weight Elimination	0.0057	**1.2440**	**2.2183**	**1.5425**	**12.671**	**0.9133**	**14.145**
6	Mixed-Norm	0.0026	**0.0455**	**0.8386**	0.0167	**0.7731**	0.0003	7.6e-6
	Weight Elimination	0.0067	**10.020**	0.9805	**83.244**	**12.641**	1.0163	0.9662

(*Bode type denotes Significant weights. Experiments with the same test number are trained with the same initial weights*)

but it does not work as well as mixed-norm penalty.

5. CONCLUSIONS

A novel penalty function for pruning ANN is presented in this paper. The 1-norm of the output weight vector is minimized while the 2-norm is kept constant. When the computation converges, neurons that are redundant will be have near zero output weights and leave only the important nodes.

The proposed penalty function appears to be very efficient comparing with the conventional penalties which uses all the weights in the ANN.

6. REFERENCE

[1] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, 1995.

[2] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, 290(5500):2319-2323, December 2000.

[3] I. Borg, and J. Lingoes, *Multidimensional Similarity Structure Analysis*, Springer-Verlag, New York, 1987.

[4] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd Edition, Academic Press, New York, 1990.

[5] G.E. Hinton, "Connectionist learning procedures," Artificial Intell., vol. 40, no. 1, pp. 143-150, 1989.

[6] E.B. Baum, "What size of neural net gives valid generalization?" *Neural Computation*, 1(4):1, pp. 51-160, 1989.

[7] J.K. Kruschke and J.R. Movellan, "Benefits of Gain: Speeded Learning and Minimal Hidden Layers in Back-propagation Networks," *IEEE Transactions on Systems, Man and Cybernetics*. 21(1), pp. 273-280, 1991.

[8] A.S. Weigend, D.E. Rumelhart, and B.A. Huberman "Generalization by weight elimination with application to forecasting," in *Advances in Neural Information Processing Systems 3*, R.P. Lippmann, J.E. Moody, and D.S. Touretzky, eds. Morgan Kaufmann, San Mateo, CA, pp. 875-882, 1991.

[9] J. Moody and T. Rögnvaldsson, "Smoothing Regularizers for Projective Basis Function Networks," in *Advances in Neural Information Processing Systems 9*, pp. 585-591, M.C. Mozer, M.I. Jordan and T. Petsche, eds. MIT Press, Cambridge, 1997.

[10] J. Sietsma and R.J.F. Dow, "Neural net pruning - Why and how? m, in *Proc. 1EEE Int. Conf. on Neural Network*, vol. 1, San Diego, CA, pp. 325-332, 1988.

[11] G. Castellano, A.M. Fanelli and M. Pelillo, "An iterative pruning algorithm for feedforward neural networks," *IEEE Transactions on Neural Networks*, 8(3):519--531, 1997.

[12] M.Y. Chow and J. Teeter, "An Analysis of Weight Decay as a Methodology of Reducing Three-Layer Feedforward Artificial Neural Networks for Classification Problems," *Proceedings of IEEE International Conference on Neural Network*, Orlando, FL, pp. 600-605, 1994.

[13] N. Alexandrov and J.E. Dennis, "Algorithms for Bilevel Optimization," *AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analyis and Optimization*, Florida, pp. 810—816, 1994.

[14] http://color.psych.upenn.edu/hyperspectral/bearfruitgray/bearfruitgray.html