Classification of Multispectral Remote Sensing Data Using a Back-Propagation Neural Network

Philip D. Heermann and Nahid Khazenie, *Member, IEEE*

Abstract—**The suitability of a back-propagation neural network for classification of multispectral image data is explored. Methodology is developed for selection of both training parameters and data sets for the training phase. A new technique is also developed to accelerate the learning phase. To benchmark the network, the results are compared to those obtained using three other algorithms: a statistical contextual technique, a supervised piecewise linear classifier, and an unsupervised multispectral clustering algorithm. All three techniques were applied to simulated and real satellite imagery. The simulated data allowed study of the functionality of each method over a diverse but controlled environment for a more accurate assessment of performance. The real-world data was included to study the implementation details and the real-world feasibility of the new method. Results from the classification of both Monte Carlo simulation and real imagery are summarized.**

Keywords—**Neural network, back-propagation, multispectral, remote sensing, training acceleration, classification.**

I. Introduction

MODERN remote sensing satellites generate multispectral data from a variety of sensors. Timely analysis of this plethora of data already presents a formidable challenge. As the spatial, spectral, and temporal resolution of the instruments increase, the need for advanced and efficient techniques escalates.

The challenge in dealing with these problems lies in the specification and development of both hardware and software systems which can handle the data rates at down-link or near down-link speeds. This process is commonly evolutionary in nature with software being developed to efficiently utilize the hardware, and, where possible, the hardware being modified to assist the software. Normally, however, the software development is the critical path in the project. The software development is usually a two-phase process with algorithms first developed on serial Von Neumann style computers and then later tailored to a hardware model with accelerated performance. The tailoring may involve complete redesign of the algorithm.

In contrast to this development process, the development of an artificial neural network technique starts with an inherently parallel processing technique. The development requires the selection of an appropriate neural network architecture and tuning of the network to solve the problem most efficiently and correctly. The neural network techniques generally use a learning method to "program" the network by use of examples. This changes the solution process from finding ways to understand and represent the problem in a computer language to a task of providing examples of the problem for the network to learn.

One of the goals of this research was to examine the operation of one type of neural network technique, back-propagation, under the conditions encountered in processing remote sensing data. Most studies of neural network techniques concentrate on problems with small data sets of usually less than 100 patterns stored. This study emphasized the analysis of larger data sets with the number of training patterns numbering 4200 and 22 000 from an image numbering four million pixels. In terms of applications, the study focused on classification of vegetation types from multichannel image data. The study includes both simulated data and two Landsat-TM scenes of the region surrounding the Chernobyl nuclear power plant.

The back-propagation network was trained on representative areas of the image and then the resulting neural network was used to classify the entire image. It was necessary to develop new techniques to facilitate general application of a back-propagation network for classification of remotely sensed images. To allow comprehensive evaluation of the neural network technique, it was compared to several different statistical methods. These results are reported in the final section of this paper.

II. Fundamentals of Neural Networks

Artificial neural networks have been investigated by scientists in a diverse range of disciplines including computer science, psychology, biology, and organic chemistry. Although the motivation for study varies, the main idea, computing using methods inspired by biological systems, remains constant.

Back-propagation, which is also known as the generalized delta rule [1], [2] is one of the most popular and widely investigated methods for training neural networks. The most common network topology is multiple layers with connections only between nodes in neighboring layers. There are no connections between nodes located in a common layer. Information is passed in one direction through the network; starting at one side of the network and moving through successive layers. This type of network is known as a feedforward network and is presented in Fig. 1.

The edge layer, where information is presented to the network, is the input layer. The layer on the far side, where the processed information is retrieved, is known as the output

Manuscript received April 9, 1991.

The authors are with the Mechanical Engineering Department and Center for Space Research, The University of Texas at Austin, Austin, TX 78712.

N. Khazenie is also with the University Corporation for Atmospheric Research, Boulder, CO 80302.

IEEE Log Number 9103943.

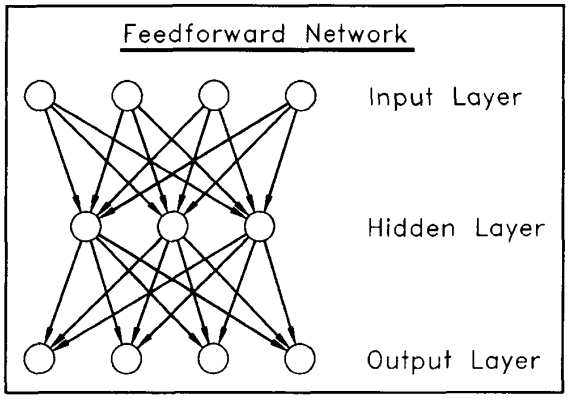

Fig. 1. Three-layer feedforward network.

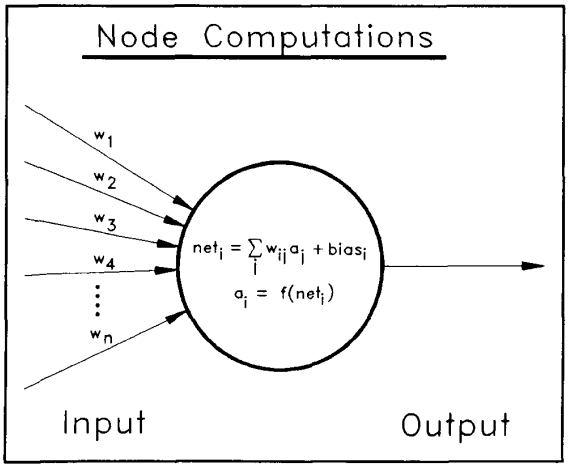

Fig. 2. Node computations.

layer. All layers in the middle are known as hidden layers. Fig. 1 illustrates a network with only one layer of hidden units, but more hidden layers are possible and are commonly used.

All nodes in the network (Fig. 2), except the input nodes , perform the same two functions; collecting the activation of nodes in the previous layer and setting an output activation. The input nodes activations are determined by the input data. The collection function [2] used in this study is

$$net_{pi} = \sum_j w_{ij} a_{pj} + bias_i \tag{1}$$

where variable w_{ij} represents the connection strength for the current node i to a node j in the previous layer, a_{pj} is the activation of node j for pattern p plus a node bias, $bias_i$, which can be considered a connection to a node which is always at full activation. The result of the collection function, net_{pi}, is passed to the output section which determines the node's output activation, a_{pi}. The output function is a nonlinear function which allows a network to solve problems that a linear network cannot solve [2], [3].

In this study the sigmoid function [2] given in (2) is used to determine the output state.

$$a_{pi} = \frac{1}{1 + e^{-net_{pi}}}. \tag{2}$$

A back-propagation network is trained by example. A set of representative input and output patterns is selected. As each input pattern is presented, the connections of the network are adjusted so that the activation of the output nodes more closely match the desired output pattern. All the patterns are repeatedly presented to the network until the network "learns" the patterns. The foundation of the back-propagation learning algorithm is the nonlinear optimization technique of gradient descent [5] on the sum of the squared differences between the activation, O_{pi}, of the nodes in the output layer and the desired output t_{pi}. The objective is to minimize:

$$E = \sum_p \sum_i (t_{pi} - O_{pi})^2 \tag{3}$$

where p indexes the training patterns and i indexes the output nodes of the network. By adjusting the network connection strengths, w_{ij}, the above function is minimized and the network "learns" the patterns.

Application of the gradient descent method [3] yields the following iterative weight update rule:

$$\Delta w_{ij}(n+1) = \varepsilon(\gamma_{pi} a_{pj}) + \alpha \Delta w_{ij}(n) \tag{4}$$

where w_{ij} is the connection strength from node i to node j, γ_{pi} is the node i error for pattern p, a_{pj} is the activation of node j for pattern p, and ε is a parameter known as the learning rate and the parameter α controls the momentum term. The node error, γ_{pi}, for an output node is then given as

$$\gamma_{pi} = (t_{pi} - a_{pi}) a_{pi} (1 - a_{pi}) \tag{5}$$

where the first term is the error between an output node's activation and the target pattern. The other terms in (5) are the result of the derivative of the activation function.

The error at an arbitrary hidden node i is

$$\gamma_{pi} = a_{pi}(1 - a_{pi}) \sum_k \gamma_{pk} w_{ki}. \tag{6}$$

The summation in (6) collects the errors from the layer below and the other terms are the derivative of the activation function. For details of the back-propagation learning algorithm including derivation of the equations see [2].

III. Methodology

Bendiktsson *et al.* [3] demonstrated by comparison with statistical methods that a three-layer back-propagation network showed good potential for processing multisource remote sensing and geographic data. However, the main drawback to the network method was the slow learning phase. In addition, it was concluded that the proper selection of training data is a crucial step in achieving best results.

To address the main issues indicated above, the back-propagation network was evaluated by comparing the network operation with statistical techniques and by studying the network on a large real data set. The goal of the comparisons with

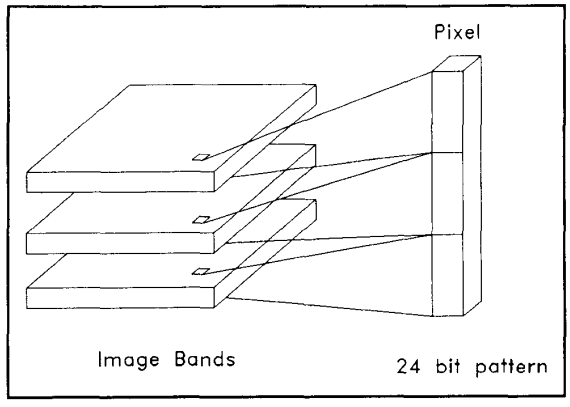

Fig. 3. Construction of input patterns for back-propagation network.

statistical methods was to access the effect of training data on network operation and to study the generalization from image to image. Feasibility of using a back-propagation network under "real world" conditions was the focus of the work with large real images to ensure that problems of significant scale would be present.

A. Data Sets

To study the neural network, three data sets were selected for study, two simulated data sets and one set of actual satellite data. The first simulated data set consisted of three bands of data. The size of the image was 64×64 pixels and contained three classes. The image had a signal to noise ratio (SNR) of 1.2 and a first-order spatial correlation of 0.3.

The second simulated data set was a temporal series of fourteen images of size 40×40 pixels. The data consisted of two bands, or channels, with the first containing a SNR of 0.7, and the second band had a SNR of 1.2. The first-order spatial correlation was 0.55 and the temporal correlation was 0.65, respectively. Again, the data represented three separate ground classes.

The third data set consisted of two Landsat Thematic Mapper images from the region surrounding the Chernobyl nuclear power plant in the Soviet Union. Each 2048×2048 pixel image consisted of three 8-b bands (Landsat bands 2, 4, and 7); Each image consisted of 12 megabytes of data. The images were taken on two dates, May 24, 1987 and April 29, 1986. Both images were classified into five types of ground cover.

To present the data to the back-propagation network, an input pattern was constructed by concatenating the data from all three bands at each pixel location. This process is illustrated in Fig. 3.

Since each band of the Landsat data was 8 b deep this process resulted in a 24-b pattern for each pixel.

The simulated data was slightly different than the real data. Each channel contained 32-b floating point values rather than 8-b data. Therefore, each floating point value was digitized to 8-b resolution and concatenated with the other band(s) to form a pattern. This was done because the training set preparation and neural network software was designed for binary data. It is important to note that this is not the only way to present data to the network as the floating point values can be normalized to the zero-to-one range and presented directly to the network.

B. Preparation of Training Sets

The selection of a training set from the real data consists of two steps referred to as "picking" and "packing." In the picking phase, an unsupervised clustering algorithm is applied to the multiband data. Then several small homogeneous regions inside the clusters are selected by hand to represent each corresponding category. The picking software displays the three channels of data in 24-b RGB color with zooming capabilities to provide as much information as possible during selection of the training regions. Using the picking software, subsets of pixels are picked from the full color image and placed in classification categories. Each subset is a small region containing a variable number of 24-b pixels.

The result of the picking is many small subsets of the image representing examples of the desired classifications. Each class contains multiple small regions from the overall image. The regions were selected to incorporate variations in the data within a given class. For example, in the center of the Landsat image, taken in May 1987, there is a slight decrease in the intensity of all pixels. Therefore, regions were included from this area to provide information to the network about the blemish. This technique could also be applied to mosaic images where slight variations between composite images are common.

The packing software processes the training set data for efficient storage and "cleans up" the data in order to maximize the information extracted. The training data is cleaned up in two ways: a) only one example of each pixel is allowed, and b) a given pixel value is accepted in only one class. This ensures that each training pixel is unique and all pixels contain new information with no conflicting data.

To accomplish this task, the packing software first reads the picked values from the three separate image bands and loads the bits into a single 32-b integer by shifting and adding the 8-b values. Then each pixel (integer) is compared to all the pixels which are already packed. If the test pixel value is equal to any already packed pixel, it is discarded, otherwise, the new value is added to the pool of packed pixels. Throughout the entire process the class membership, which was determined by the picking, is maintained. The result is many pixels in each class, but each pixel is unique to the union of all the groups.

C. Network Selection

The number of input nodes is specified by the dimension of the input patterns. For the Landsat data the input pattern is from one pixel consisting of three bands at a precision of 8 b/band to give 24 b per input pattern. The data is presented with one input node for each bit in the pattern. Similarly, the output nodes are determined by the number of categories to be classified or the desired output mapping.[1] The hidden nodes, however, are not specified by the problem. To select

[1]Two output mappings were studied: a) one output node per class, and b) a binary mapping where the first class was mapped into a binary 1 (001), the second class mapped to a binary 2 (010), the third class to a binary 3 (011), etc.

the proper number of hidden nodes requires some knowledge of the purpose of the middle layer.

The middle layer forms an n-dimensional space where n is the number of nodes in a given hidden layer. The hidden layer allows the network to form its own internal representation of the data. The internal representation is the foundation on which the decision boundaries are formed.[2] If too few middle nodes are selected, the network may not contain sufficient degrees of freedom to form a representation. The learning algorithm, since it is based on a heuristic, may also improperly train hidden nodes. Extra middle nodes provide room for improperly trained nodes to be ignored by later layers rather than taking the time to retrain the erroneous node.

The downside to a large number of middle nodes is the increased computation required and the loss in generalization ability of the network. Therefore, a balance must be drawn. We have found that it is best to start with a relatively large number of hidden nodes, usually at least as many nodes as in the input layer. As experience is gained with the network, the middle layer may be reduced. The advantage of starting with many hidden nodes is that the network will probably solve the problem, albeit slowly, while if too few hidden nodes are selected the learning algorithm may oscillate or simply not learn the classifications. Oscillation of the network does not necessarily indicate that the hidden layer is too small. The most common source of oscillation or instability is the improper selection of the learning rate, ε in (4).

D. Learning Rate Selection

To address the problem of slow training, attention was devoted to small details during the development of the back-propagation software. As suggested by McClelland [4], the weights were updated once every complete cycle through the patterns rather than after each pattern. This process is known as batch training or weight update by epoch. This reduces the computations required at each step.

The back-propagation algorithm, like other numerical algorithms, can become unstable if the steps are too large. McClelland [4] recommends $1/n$ as the proper size for the learning rate ε, where n is the total number of nodes in the network.

However, for batch processing this selection of ε is too large resulting in saturation of the weights in less than ten steps. This is a result of the large number of training patterns. The smallest training set for the Landsat data was 4200 patterns with other training sets growing to 22 000 patterns. The sum of the errors from all the patterns was simply too large and induced instability in the gradient descent algorithm. Based on this information we added the term $1/p$ where p is the number of patterns, to the equation for calculating the learning rate, ε. This, however, was found to be too conservative so a multiplicative factor was incorporated to improve the estimate. The resulting equation, where $C_0 = 10$, is

$$\varepsilon = C_0 \times \frac{1}{p} \times \frac{1}{n}. \tag{7}$$

[2]Due to the use of a sigmoidal activation function the decision boundaries are gradual transitions rather than threshold like hyperplane decision boundaries. The network both positions the decision boundaries and determines the sharpness of the transition.

The choice of C_0 is based on the experience to date and will probably change as development continues. The other controlling parameter in (5) is known as the momentum term α.

The purpose of the momentum term is to damp out oscillations which occur when traversing a sloping but steep walled canyon. The momentum term cancels the side-to-side oscillations without disturbing the motion along the canyon. Thus, the momentum term allows the learning rate to be larger without instability which speeds training of the network.

Back-propagation is based on gradient descent method, which is one of many techniques for nonlinear optimization. There are other methods of nonlinear optimization which are more efficient than gradient descent [4]. The other methods, however, would require a redesign of the learning algorithm. Therefore, it was decided to examine a minor modification which was suggested by Vogl *et al.* [5]. The method is an adaptive method of adjusting the learning rate and controlling the momentum [6]. The adaptive back-propagation algorithm is presented in Appendix A.

The main idea is to increase the learning rate if the last step results in a decrease in the total sum squared error. If the error increases, then disable the momentum term and reduce the learning rate. The momentum is simply pushing the search in the wrong direction and the learning rate should be reduced to find the proper direction to move. Once the error begins to decrease again, the momentum term is included and the learning rate is increased with each good step. The algorithm incorporates some of the ideas found in the faster nonlinear optimization methods. The most notable feature is an increasing step size, which performs a function similar to a line search, which is prevalent in other techniques [7].

The algorithm is different from the method suggested by Vogl *et al.* by the addition of the minimum learning rate. For this application this was required to keep the learning rate from decreasing to zero. The minimum learning rate is the rate defined by (7).

IV. Empirical Results

The performance of the back-propagation network is affected by many factors. Therefore, several experiments were conducted to gain insight into the network operation. Tests were conducted to study the effect of training acceleration, training sets, network accuracy, effect of hidden units, generalization ability, and speed against the various other methods.

A. Comparison with Statistical Methods

To assess the accuracy and generalization capability of the back-propagation network, it was compared to three other classification techniques, an unsupervised multistage clustering algorithm, a linear classifier, and a contextual statistical spatial–temporal method.

The unsupervised method was a Markov random field method [8]. This multistage algorithm makes use of spatial contextual information in a hierarchical clustering procedure for image segmentation. This method uses a Markov random field model to enforce local spatial smoothness, and utilizes a maximum entropy principle to quantify global smoothness in the image.

The linear classifier was a supervised method [9]. This approach is based on a committee classification technique in the form of seniority decision logic. The method implements piecewise linear decision surfaces of the multiband data where the number of necessary hyperplane decision vectors, i.e., committee members, is automatically determined.

The supervised contextual classification algorithm incorporates a spectral, spatial, and temporal correlation model for the underlying process. It models the spatial class configuration of the patterns using a geometric probability model [10].

Back-propagation was studied for generalization across the temporal series of 14 similiar images. The network was compared to the linear classifier and the spatial–temporal technique. The results are presented in Fig. 4. Although the network performed with less overall accuracy than both methods, this is probably an exception since back-propagation generally performs better than linear techniques, since the network is a superset of a linear discriminator. The linear classifier had the advantage of training on 25% of the data, while the network trained on only 20%.[3] The network was also handicapped because the simulated data was 32–b floating point data which was digitized to only 8 b per channel for input to the network. The main goal of this test was to investigate the generalization ability of the neural network which was generated by our back-propagation simulator which was optimized for the real images. In general, however, a back-propagation network is not limited to 8-b data.

The contextual spatial–temporal technique had the best performance on the two-band data set. The technique uses information from the neighborhood of pixels, both spatially and between temporal images. The network can easily be expanded to include the additional neighboring pixels. Both of the extensions are underway and empirical results will soon be available.

B. Results of the Neural Network Method

The adaptive back-propagation method had the same error rate as conventional back-propagation, but the learning time was reduced by five to ten times. Training time for the Landsat data was reduced from 3 weeks to 3 days on a Hewlett–Packard 9000–835 workstation. The degree of improvement in training speed varied between data sets, but the gains were always at least a factor of five. The adaptive algorithm was also tested against standard back-propagation on common neural network test problems of parity, exclusive or (XOR), and character recognition with no major differences in final networks.

The selection of training patterns has an important effect on the accuracy of the network. Fig. 5 illustrates the performance comparison of a 24/24/3 network (24 input nodes, 24 hidden nodes, 3 output nodes) trained on the 64×64 pixel simulated data with a 24/24/5 network trained on the satellite data. The network classifying the simulated data was trained on

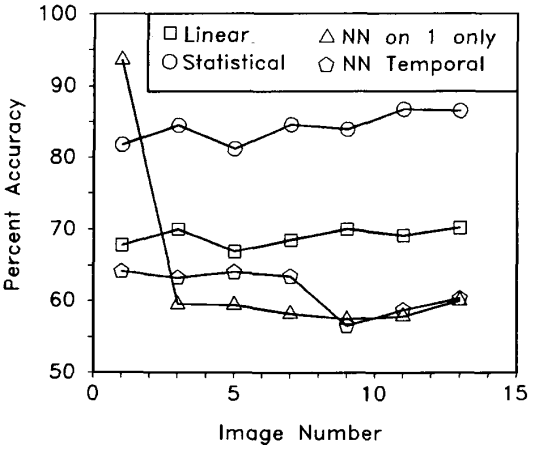

Fig. 4. Comparison on simulated data of Linear, Statistical Spatial–Temporal, Neural Network trained on first image, and Neural Network trained on 20% of the first seven images (networks trained on 1600 and 1604 patterns, respectively).

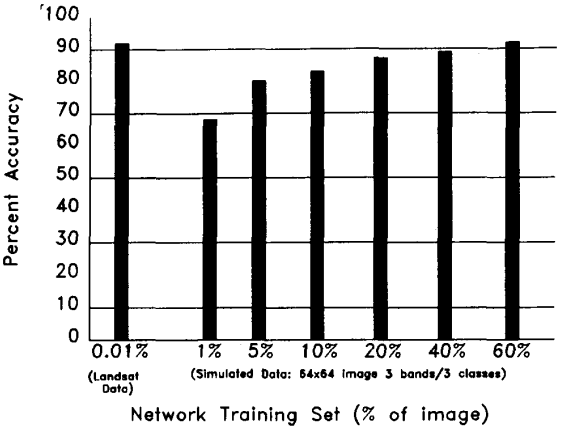

Fig. 5. Effect of training set size.

a randomly selected subset of image. The size of the training set varied from 1 to 60% of the entire image. The accuracy of the network on the entire image increased as with the larger training sets, but with diminishing returns. The network also performed reasonably well with few training patterns. For example, using only 1% of the data (40 randomly selected patterns), the network correctly classified 68% of the image.

The Landsat image, trained on only 0.1% (approximately 4200 patterns) of the data, yielded an accuracy around 90%.[4] The Landsat image tests with larger training sets, up to 22 000 patterns, showed little improvement in accuracy over the 4200 patterns. This suggests that the absolute size of the training set may be important. The small simulated training sets may be statistically insufficient to characterize the underlying process. It also suggests that to properly study neural network

[3] Due to computer time constraints, we were unable to train the network on 25% of the entire temporal series of images. However, the network trained on 40% of the first image yielded 69.5% accuracy on the first image and across the remaining temporal series of images averaged 52.3% accuracy with a standard deviation of 1.2%.

[4] Ground truth data was not available on the Landsat data. Therefore, the results of network were carefully studied for misclassification. After meticulous review, the only major area of misclassification was the cloud boundaries. The error indicated is the the sum of the unknown pixels and the cloud misclassifications.

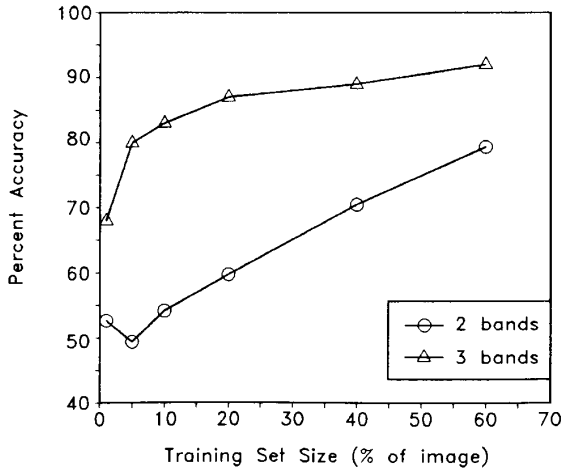

Fig. 6. Effect of data set and training sample size on classification accuracy using simulated data.

techniques, large scale data should be studied because the small images will probably not be indicative of the network performance on larger imagery. The method of selecting the training sets also affects the classification accuracy.

All the training sets were processed to remove any repeated or cross-classified patterns and to remove extra and confusing patterns. The main difference between the simulated data and real image training sets is the selection of patterns. The real data training patterns were selected systematically rather than randomly selected. The regions were selected to be representative of each class, so the training sets were higher in quality than the random selection.

Overall, the study indicates a positive correlation between the classification accuracy and the size of the training set. The exact number, however, varies from image to image. Fig. 6 illustrates this concept by comparing the effect of training sets on classification accuracy for independent simulated images.

Both curves show the characteristic increase in accuracy as the sample size grows. The two-band data set, however, is more difficult to learn so more data is required. This result was expected since the two-band data set is simulated according to a very complex distribution model. The data was generated such that considerable correlation existed between spectral bands, spatial neighbors, and classes. This three-dimensional correlation structure models the worst possible case of real data and has proven to be the most difficult case for every classification technique.

C. Real Data Set Analysis

The results of the Landsat data are presented in Table I and Table II. All the networks were trained using patterns from the May data until at least 99% of the training set was learned. The real data was studied for the effect of hidden nodes, output nodes, and ability of the network to generalize both within and between images.

No ground truth was available for the satellite data so Tables I and II allow the different network structures to be compared against each other. The results of 24/24/5 network on the May data were carefully checked by visual inspection. The only major errors found were in perimeters of the clouds which were classified as urban rather than cloud. This resulted in 0.5% of the image being placed in the urban class rather than the cloud classification.

TABLE I
CLASSIFICATION RESULTS (PERCENTAGE OF TOTAL IMAGE IN EACH CLASS)

Date	May 24, 1987			
Class	Network			
	24/24/5	24/30/5	24/37/5	24/24/3
Water [H_2O]	3.84%	3.93%	3.80%	5.74%
Cloud [Cld]	1.28%	1.28%	1.29%	3.23%
Urban [Urb]	3.25%	3.70%	3.23%	3.71%
Forest [For]	45.11%	41.63%	39.15%	37.66%
Agriculture [Ag]	36.93%	40.81%	41.77%	49.54%
Urb + H_2O	*	0.01%	0.01%	*
Urb + For	*	0.01%	0.05%	—
Ag + H^2O	—	—	0.02%	0.01%
Ag + For	1.09%	0.78%	3.04%	—
For + H_2O	—	—	—	0.11%
For + Cld	0.01%	*	—	—
Unknown	8.49%	7.85%	7.65%	—

* less than 0.01%
— not possible or no classifications

TABLE II
CLASSIFICATION RESULTS (PERCENT OF TOTAL IMAGE IN EACH CLASS)

Date	April 29, 1986			
Class	Network			
	24/24/5	24/30/5	24/37/5	24/24/3
Water [H_2O]	5.36%	5.63%	5.38%	6.86%
Cloud [Cld]	0.11%	0.11%	0.11%	3.68
Urban [Urb]	10.76%	10.00%	8.23%	16.75%
Forest [For]	46.22%	42.15%	48.73%	45.86%
Agriculture [Ag]	15.56%	19.07%	16.61%	26.78%
Urb + H_2O	*	*	*	0.02%
Urb + For	0.04%	*	0.16%	—
Ag + H_2O	—	—	—	0.01%
Ag + For	0.03%	0.09%	0.18%	*
For + H_2O	—	—	—	0.04%
For + Cld	*	*	—	—
Unknown	21.92%	22.95%	20.60%	—

* less than 0.01%
— not possible or no classifications

Varying the number of hidden nodes appears to have little effect on the result. The only effect noticed was a larger hidden layer allowed the network to learn more rapidly. The study of the hidden layer was not exhaustive and more empirical work needs to be performed. The output mapping, however, has a significant effect.

The 24/24/3 network in Tables I and II was designed to map the classification to a binary output rather than one output node per class. The goal of the binary mapping was to force

TABLE III
APPROXIMATE EXECUTION TIME ON 2.0 MFLOP COMPUTER

Method	Training Time		Classification Time	
	40 x 40	2048 x 2048	40 x 40	2048 x 2048
Linear Classification	—	—	3.0 min. [1]	60 days
Unsupervised Method	—	—	0.63 min. [1]	5 days
Statistical Method	4.4 min. [1]	8.0 days	2.6 min. [1]	4.7 days
Neural Network	~5 days [1.2]	~5 days [1.3]	0.15 min. [1]	0.1 days [1]

1. Actual Test
2. 1600 Training Patterns
3. 4200 Training Patterns

the network to classify all the pixels without allowing any unknown pixels. This mapping, however, performed poorly and was inferior to the one-class to one-node mapping in all tests.

The generalization ability of the network was very good. The network was able to accommodate the slight fade in pixel intensity in the center of the May 1987 image. Training patterns from this region were included in the training set with the result of the network showing no degradation of classification performance in the faded region. The network also performed reasonably on generalizing the classification to the image of April 1986. The unknown classifications jumped to around 20%, but major features such as a full river from spring runoff and fields without crops were easily identified.

The main classification errors for the network are the perimeters of clouds. In all cases, the cloud boundaries were confused with urban area.[5] The use of more data bands or spatial information may correct the problem since the signatures of the different categories would be better distinguished.

The execution time of the back-propagation network was compared to the linear classifier, the unsupervised method, and the spatial temporal classifier. Table III presents the results of the timing studies. The execution times are presented in terms of the time required on an Hewlett–Packard 9000–835 workstation.

The main disadvantage of the network technique is the slow training phase. The adaptive back-propagation method greatly reduced this problem. For the real image the computations time was reduced from 400–800 h to less than 100 h on the Hewlett–Packard workstation. Training time varied widely depending on the difficulty of the problem and the size of the training set. The small training sets were learned in as short as 2 min while the training on the actual satellite data required as long as 100 h.

Network training does not proceed linearly. Most of the training set is learned during the initial portion of the training phase with the rate of improvement decreasing as learning progresses. The exact percentages, of course, vary from image to image. For example, the 40×40 temporal data proved to be difficult to learn. Learning 1600 patterns of this simulated data was as difficult as 4200 patterns from the Landsat data. The 40×40-pixel simulated data set was representative of worst case data. For this case the training time appears to depend as strongly on the difficulty of the problem as the size of the training set.

A back-propagation network is slow to train, but it is fast in the classification stage. The network is also compact which would allow classification information to be distributed as a network rather than another image channel.

Classification time depends linearly on image size, so classification of large images like an entire continent are possible. For example by extrapolating from the 2.0 million floating point operations per second (MFLOP) performance of the Hewlett–Packard workstation, a small supercomputer with 80 MFLOP performance could learn the 4200 patterns from the satellite image in less than 3 h. The classification time for the entire 2048×2048 image would require less than 5 min. Because the classification phase is so fast and the classification time increases linearly with the image size, the network technique could be used to classify the Earth globe. The training time would depend on the number of training patterns, but classifying the entire Earth globe with 1 km resolution using three bands could be accomplished in approximately 13 h on an 80 MFLOP supercomputer.

V. DISCUSSION

The neural network technique of back-propagation appears to be feasible for classifying satellite images. The adaptive back-propagation algorithm reduces the main drawback, the training time, to a reasonable level. Neural network hardware is also progressing such that real time or near real time learning will be possible [11].

Back-propagation is easily modified to accommodate more channels or to include spatial and temporal information. The input layer can simply be expanded to accept the additional data. Expansion of the input increases the computation on the order of N^2, If the size of the input is doubled, the required computations will be four times as great. Therefore, new

[5]The urban classification was actually more of an industrial or soil classification since the training samples were taken from the power plant grounds, which are devoid of vegetation.

channels of information should not be added indiscriminately. However, if each node of the network is placed in a separate processor, the input expansion is again of order N. Back-propagation is inherently parallel, so all that is required is a highly parallel machine. No major modifications in the algorithm would be necessary.

The classification time is linearly proportional to the size of the image and each pixel is calculated independently of its neighbors. Therefore, course-grain multiprocessor machines could speed the classification by splitting the large image into smaller sets.

The network generalizes fairly well from image to image, but the network can also be trained through time. The retraining should be easier than the initial training since the network needs to make only small corrections to the connection strengths rather than learn the problem from the beginning.

Although the study of neural network techniques for classifying multispectral and multisource satellite data is still in its infancy, a back-propagation network appears to be a feasible classifier for very large multichannel images.

APPENDIX A
ADAPTIVE BACK PROPAGATION ALGORITHM

```
Loop while( error > desired error )
   begin
   Perform forward pass and
   sum errors for all patterns

   If( error < 1.02*lasterror )
      begin
       Update weights
       If( error < lasterror )
          begin
           Increase learning rate by 5%
           Turn momentum on
          end
      end

   Else
      begin
       Reduce Learning rate by 30%
       If( momentum = on )
          begin
           Remove last weight update
           Turn momentum off
          end
       If( learning rate < min learning
             rate )
          begin
           Set Learning rate to minimum
             learning rate
           Update weights
          end
       end
       lasterror = error
end
```

REFERENCES

[1] D. E. Rumelhart and J. L. McClelland, Eds., *Parallel Distributed Processing*, vol. 3. Cambridge, MA: MIT Press, 1986.

[2] J. L. McClelland and D. E. Rumelhart, Eds., *Parallel Distributed Processing*, vol. 1. Cambridge, MA: MIT Press, 1986.

[3] J. A. Bendiktsson, P. H. Swain, and O. K. Ersoy, "Neural network approaches versus statistical methods on classification of multisource remote sensing data," *IEEE Trans. Geosci. Remote Sensing*, vol. 28, pp. 540–552, July 1990.

[4] D. G. Luenberger, *Linear and Nonlinear Programming* (2nd ed.). Menlo Park, CA: Addison-Wesley, 1984.

[5] T. P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, and D. L. Alkon, "Accelerating the convergence of the back-propagation method," *Biological Cybernetics*, vol. 59, pp. 257–263, 1988.

[6] P. D. Heermann and N. Khazenie, "Analysis of large multi-dimensional data with a back-propagation neural network," presented at the Proc. INNS-90 San Diego, International Joint Conference on Neural Networks, San Diego, CA, 1990.

[7] D. F. Shanno, "Recent advances in numerical techniques for large scale optimization," *Neural Networks for Control*, W. Miller III, R. S. Sutton, and P. J. Werbos, Eds. Cambridge, MA: MIT Press, 1990.

[8] S. Lee and M. M. Crawford, "Statistical based unsupervised hierarchical image segmentation algorithm with a blurring corrector," in *Proc. IGARSS 89, 12th Canadian Symposium on Remote Sensing*, Vancouver, Canada, vol. 2, 1989.

[9] N. Khazenie, "Contextual classification of remotely sensed data based on a spatial-temporal correlation model," Ph.D. dissertation, The University of Texas at Austin, 1987.

[10] N. Khazenie and M. M. Crawford, "Spatial-temporal autocorrelated model for contextual classification," *Proc. IGARSS 89, 12th Canadian Symposium on Remote Sensing*, Vancouver, Canada, vol. 2, 1989.

[11] D. Hammerstrom, "A VLSI architecture for high-performance, low cost, on-chip learning," presented at the Proc. INNS-90 San Diego, International Joint Conference on Neural Networks, San Diego, CA, 1990.

Philip D. Heermann received the B.S. degree in mechanical engineering from Colorado State University, Fort Collins, CO in 1985 and the M.S. in mechanical engineering from the University of Texas at Austin, Austin, TX in 1987.

Currently, he is completing his doctoral dissertation at the University of Texas at Austin. His research interests include control of nonlinear systems, remote sensing, and artificial neural network applications in remote sensing and adaptive control. He is employed by ALFA Engineering, Austin, TX and spent ten months working with the Neural Network group at the Microelectronics and Computer Technology Corporation in 1990 and 1991.

He is a student member of the American Society of Mechanical Engineers and the International Neural Network Society. He is also a member of Pi Tau Sigma, where he served as President from 1984–1985 and Tau Beta Pi.

Nahid Khazenie (M'86) received the B.S.E.E. degree in 1979 from Michigan Technological University, Houghton, MI and the M.S.E.E. degree in 1980, the M.A. degree in mathematics, and the Ph.D. degree in mechanical engineering in 1987, all from the University of Texas at Austin, Austin, TX.

During 1987 and 1988 she was a lecturer/research scientist at the Departments of Mechanical Engineering and Aerospace Engineering at the University of Texas at Austin. From 1988 to 1989 she held a joint appointment between the Department of Mathematics, as a visiting assistant professor, and the Center for Space Research, as a research scientist. In 1990 she joined the Naval Oceanographic and Atmospheric Research Laboratory in Monterey, CA as a University Corporation for Atmospheric Research Senior Scientist. Her research interests include neural networks, statistical classification, edge detection, and textural analysis of remotely sensed images.