The Effect of Neural-Network Structure on a Multispectral Land-Use/Land-Cover Classification

Justin D. Paola and Robert A. Schowengerdt

Abstract

While neural networks are now an accepted alternative to statistical multispectral classification techniques for remote sensing image classification, the network approach presents both unique challenges and abilities. The size of the hidden layer must be determined by trial and error, and the random initial weight settings result in different paths for the training procedure, making the network a non-deterministic classifier. For the sample classification presented here, it was found that there was a range of optimal hidden layer sizes below which the accuracy decreased and above which the training time increased. However, it was also found that, for a fairly wide range, the hidden layer size made little difference to the final classification accuracy. Initial weight randomization was as much of a factor as hidden layer size. Using 3 by 3 windows of data in each band was found, despite increased training time per iteration, to achieve similar accuracy with less overall training time, although with less consistency.

Introduction

Neural-network classifiers[1] are non-parametric and therefore may be more robust when distributions are strongly non-Gaussian. During training, the network is capable of forming arbitrary decision boundaries in the feature space. This ability gives it an advantage over statistical classifiers because the decision regions are adjusted iteratively by the training algorithm to fit the intrinsic distributions of the classes, whether they are Gaussian, multi-modal, or (in the case of four-layer networks) even disjointed (Lippmann, 1987).

Supervised application of the neural-network classifier is much like that of a standard statistical classifier. The differences are in the details of the training and classification algorithms. The network training phase is analogous to the class mean and covariance matrix calculations of the maximum-likelihood statistical classifier. Instead of a one-time calculation of statistical measures, however, the network is trained in an iterative fashion, typically by the *backpropagation* algorithm, until some targeted minimal error is achieved between the desired output (the training classes) and actual output values of the network. For the classification phase, instead of calculating discriminant functions on the basis of the distributions determined from the training data, as in maximum-likelihood, the network is used in a feed-forward mode like a hard-wired circuit. The entire image is fed into the net pixel-by-pixel, and a simple metric, such as the maximum network response at the output stage, is used to make a class selection for each pixel. The neural-network classification method has been compared to maximum-likelihood and other traditional techniques in numerous studies (Paola and Schowengerdt, 1995a; Blonda *et al.*, 1994; Fierens *et al.*, 1994; Yoshida and Omatu, 1994; Kanellopoulos *et al.*, 1993; Bischof *et al.*, 1992; Heermann and Khazenie, 1992; Liu and Xiao, 1991; Benediktsson *et al.*, 1990; Key *et al.*, 1990; Key *et al.*, 1989).

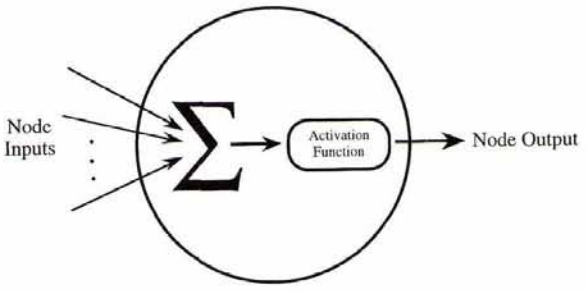


Figure 1. Internal structure of a neural-network processing node. Node inputs are summed and passed through an activation function which is usually a sigmoid.

Network Structure and Setup

The basic element of a neural network is the processing node (Figure 1). Each processing node sums the values of its inputs. This sum is then passed through an arbitrary activation function to produce the node's output value. The processing nodes are organized into layers, each generally fully interconnected to the following layer. There are no interconnections within a layer, however. In addition, there is an input layer that serves as a distribution structure for the data being presented to the network. No processing is done at this layer. One or more actual processing layers follow the input layer. The final processing layer is called the output layer. Layers in between the input and output layers are termed hidden layers. Figure 2 shows the generic structure for a commonly used configuration, the three-layer neural network. The interconnections between each node have an associated weight. When a value is passed down that interconnection, it is mul-

[1]We restrict our analysis to the widely-used Multi-Layer Perceptron (MLP) type of network.

Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721 (paola@ece.arizona.edu).

Photogrammetric Engineering & Remote Sensing, Vol. 63, No. 5, May 1997, pp. 535–544.

0099-1112/97/6305-535$3.00/0

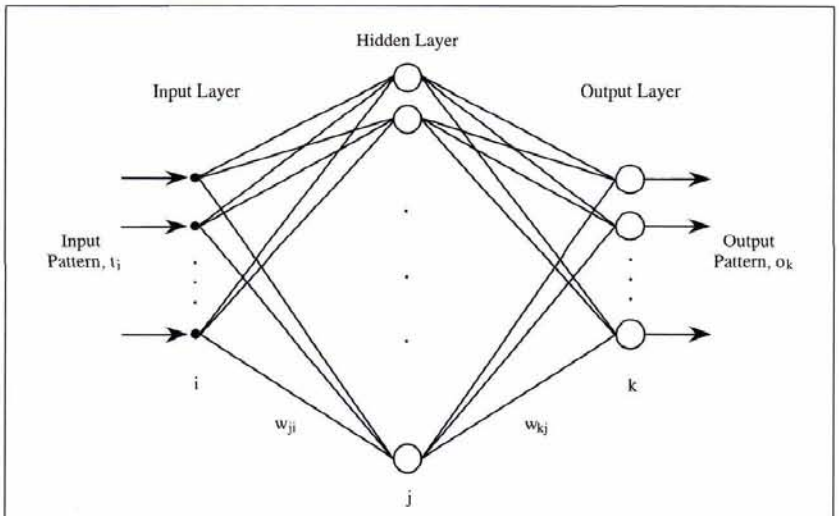


Figure 2. A generic three-layer neural network structure. The weights w_{ji} and w_{kj} are adjusted during backpropagation so that the error between the desired (as determined by the training data) and actual output pattern, o_k, is minimized.

tiplied by the weight. After training, these weight values contain the distributed learned information of the network.

Training a neural network involves setting several initial parameters. The first step is to determine the training data and corresponding desired outputs for that training data. Then overall network structure must be defined. When the training process begins, all of the weights of the network must be set to random values; otherwise, the network might not converge to a minimum training error (Rumelhart *et al.*, 1986). Then the learning rate and momentum parameter must be set. An adaptive learning rate can be used to avoid trial and error at this stage. One adaptive strategy is to automatically adjust the learning rate downward after some training interval if the overall training error has increased and upward if the overall error has decreased (Heermann and Khazenie, 1992). Therefore, the initial learning rate is not crucial to the success of the training, and training speed is increased because the learning rate is adjusted to the highest value that does not cause instability.

The final required parameter is the training convergence threshold, which must be determined experimentally. Only in simple cases is it possible to train the network to zero training error. Thus, some criterion for terminating the training process must be established, such as a threshold on the mean square error between the desired and actual output values. When this criterion is met, the training is complete and the network may be used as a feed-forward classifier. The convergence threshold controls the degree of generalization versus specialization: if the network is trained too well on the training data, it might not function accurately on the rest of the image; on the other hand, if it is not trained well enough, it will not be able to separate the classes, even in the training data, to an acceptable degree. The convergence threshold also controls the total training time.

Data Description

An urban land-use/land-cover classification of a Landsat Thematic Mapper (TM) image of Tucson, Arizona was selected as the test application for network configuration. The image was acquired on 1 April 1987 (Figure 3). The six non-thermal bands were used for the classification, which was presented in the context of a comparison of the neural network and maximum-likelihood classifiers in an earlier paper (Paola and Schowengerdt, 1995a). The classification categories (Table 1) are similar to the Level I and II land-use categories proposed by Anderson *et al.* (1976). The maximum-likelihood test site accuracy of 89.5 percent is used as a baseline for some comparisons of the various network training runs presented here. Different images and classification schemes will provide different challenges to the neural-network algorithm. A single example is explored in this paper to illustrate the technique and provide some guidelines for subsequent projects.

The test sites were well-characterized areas that were not used in the training process. However, they were rather small and limited in number (a typical problem in classification of urban regions). These areas (as well as the training sites) were determined from a manual interpretation of the study area, including site visits and aerial photography. The classification performed here is truly a land-use/land-cover classification; a number of the classes (see Table 1 and class key in Figure 7) consist of a mix of land-cover characteristics. The benefits of a neural-network approach to this type of classification are discussed in Paola and Schowengerdt (1995a). While the overall test site accuracy was on the order of 90 to 95 percent for most runs, this is not the only basis for our evaluation of different network configurations. A relative difference measure was also used in the analysis. This measure is the average overall difference in the classification maps (i.e., in the class labels) between each pair in a set of neural network runs, expressed as a percentage of the total number of pixels in the image. This value provides an indicator of the stability of the neural-network method, and is completely independent of the classification accuracy based on test sites.

Network Implementation

We used two types of input structures in our experiments. The first was the commonly used one-pixel-per-network input node method (see Paola and Schowengerdt (1995b) for a description of this and several alternative input methods). The second was a mapping of 3- by 3-pixel neighborhoods in each band to nine input nodes (for a total of 54 inputs). One node per class was used for the output of the network, with

Figure 3. Band 4 (near infrared) of the Landsat Thematic Mapper image of Tucson, Arizona. Acquired 1 April 1987.

target values of 0.1 for nodes not representing a particular class and 0.9 for the node that does represent the class. The sigmoid activation function was used in each processing node.

With the input and output structures fixed, the configuration of the middle portion of the network (the number of hidden layers and number of nodes per hidden layer) must be defined. From our survey of earlier work, it was apparent that a three-layer neural network (one hidden layer), with full interconnection between layers, would be sufficient for this type of classification. Determining the number of nodes in the single hidden layer is the focus of this experiment.

The final parameters are the learning rate and momentum. In initial training runs, these values were fixed, and it was found that too high a learning rate would lead to unstable training. To avoid this problem, the learning rate had to be set so low that training took an excessively long time. The solution was to use an adaptive learning rate and momentum. After a specified number of training iterations, the mean square error is compared to that of the previous iteration. If the error has increased, the learning rate and momentum are halved. If the error has decreased, the learning rate and momentum are increased by 20 percent. This allows for accelerated convergence when the error is steadily decreasing.

In the experiments described below, the size of the hidden layer and the type of input (single pixel or 3 by 3 window) were varied. All other network starting parameters were kept constant. The range of initial random weight values was specified to be the interval -0.1 to -0.0001 and $+0.0001$ to $+0.1$ (avoiding values too close to zero). The learning rate and momentum were 0.001 and 0.00005, re-

TABLE 1. QUALITATIVE DESCRIPTION OF THE 12 LAND-USE/LAND-COVER CATEGORIES

Class	Description
residential	single-family homes, yards, driveways (components mixed at sub-pixel resolution)
building	larger public buildings, multi-family buildings with high reflectance roofs
foothills natural vegetation	natural desert area, medium density vegetation
shaded foothills natural vegetation	within shadow natural desert area, medium density vegetation
desert scrub	natural desert area, sparse vegetation
urban	high density, large buildings, no vegetation (components mixed at sub-pixel resolution)
riparian	dense vegetation along seasonal watercourses (mostly cottonwood trees)
grass	dense, cultured grass (such as golf courses)
bare soil	exposed soil, no vegetation
asphalt	spectrally uniform material, low reflectance
concrete	spectrally uniform material, high reflectance
sand	natural material, high reflectance, no vegetation

TABLE 2. AVERAGES OF TRAINING AND TEST SITE CLASSIFICATION ACCURACIES AND MEAN SQUARE ERROR OF THE TRAINING PATTERNS FOR MULTIPLE RUNS OF THE NEURAL-NETWORK TRAINING WITH DIFFERENT NUMBERS OF HIDDEN-LAYER NODES. ALL TIMES ARE FOR A SINGLE USER PROCESS RUNNING ON THE DIGITAL IMAGE ANALYSIS LAB'S SUN SPARCSTATION 10. THE 20,000 ITERATION RUNS WERE DONE SEVERAL TIMES FOR EACH HIDDEN-LAYER SIZE, WHILE THE 50,000 ITERATION RUNS WERE PERFORMED TWICE FOR EACH SIZE.

Nodes in hidden layer	1	2	3	6	9	12	15	18	24	30	36
Mean of train site accuracy for 20,000 iterations (%)	33.3	79.0	83.6	**95.3**	**95.9**	**95.9**	**96.0**	**96.0**	**95.9**	**95.9**	**95.6**
Mean of test site accuracy for 20,000 iterations (%)	24.6	73.7	80.5	**92.1**	**93.0**	**92.8**	**93.1**	**92.5**	**92.9**	**92.4**	**92.1**
Std. Dev. of test site accuracy for 20,000 iterations	(0.15)*	(2.67)*	3.04	2.16	1.74	0.85	0.76	0.84	0.65	**0.55**	0.64
Average overall percentage of pixels with different labels (between different runs) at 20,000 iterations	(1.2)*	(12.5)*	26.7	12.3	11.0	9.2	8.7	7.4	**6.4**	**6.2**	**6.4**
Mean square error of training for 20,000 iterations	0.0042	0.0315	0.0243	0.0115	0.0097	0.0094	0.0092	**0.0089**	0.0091	0.0093	0.0093
Mean of train site accuracy for 50,000 iterations (%)	33.3	78.1	85.3	95.6	**96.4**	**96.5**	**96.7**	**96.3**	**96.5**	**96.5**	**96.5**
Mean of test site accuracy for 50,000 iterations (%)	24.5	72.8	82.9	91.1	93.4	**94.9**	**93.8**	93.0	**94.7**	**93.7**	93.1
Seconds per training iteration	0.158	0.214	0.267	0.427	0.589	0.748	0.911	1.072	1.396	1.719	2.044
Classification time (sec)	120	136	152	199	245	294	339	385	480	574	668

*The low accuracy achieved with one or two hidden-layer nodes makes the test site accuracy standard deviation and overall classification difference values unreliable measures of performance in these cases.

spectively, with adaptation every four training iterations. The neural-network classifier was implemented using the C language on a Sun SPARCstation 10 by the authors in the Digital Image Analysis Laboratory in the Department of Electrical and Computer Engineering at the University of Arizona.

Experiments

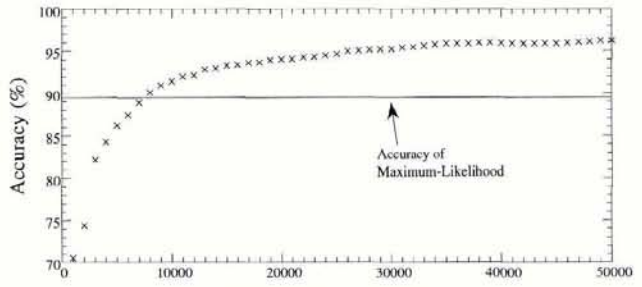
Dependence on Hidden Layer Size

Multiple training runs were made for hidden layer sizes ranging from 1 to 36 nodes. At least seven runs were done for each configuration to 20,000 iterations and two runs to 50,000 iterations. Table 2 shows the averaged accuracies and mean square error of training, with the best averages shown in bold. Clearly, networks with hidden layers of three or fewer nodes were not able to differentiate the classes as well as networks with six or more hidden layer nodes. In fact, for these cases the network often reached a relatively high minimum mean square error well before 20,000 training iterations (Figure 4c). Also, for three or fewer hidden layer nodes, the test and training site accuracies often did not increase monotonically with time as was observed with the larger nets.

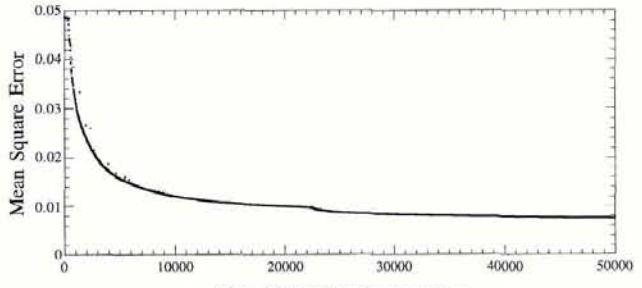
The best single-run test site accuracies were obtained with nine hidden layer nodes (95.3 percent) for 20,000 iterations and 12 hidden layer nodes (96.2 percent) for 50,000 iterations, although the network achieved a very high average over the wide range of 6 to 36 hidden layer nodes. Figure 4a shows the accuracy of the best neural-network training run (12 hidden layer nodes) as a function of iteration number. In this case, as with all the neural-network runs produced with this data set, the mean square error (Figure 4b) decreased steadily with increased training time (except when fewer than three nodes were used — see Figure 4c). It can be seen, however, that there were jumps in the error which were rectified immediately in every case by the adaptive learning rate. Theoretically, the change to the network weights at each training iteration should be infinitesimal in order to ensure an overall decreased mean square error. However, to decrease training time, the learning rate is increased slowly as long as the mean square error decreases. At a certain point, however, the learning rate becomes too large and the neural-network algorithm takes too big of a step in its gradient descent to the minimal error and the mean square error makes a sharp jump upwards. On the next learning rate adaptation iteration, the learning rate is cut in half and the mean square error returns to its original, steadily decreasing curve. Figure 5 shows how the learning rate changes as a function of iteration number in the early portion of the training. It can be seen that the initial learning rate setting of 0.001 was inconsequential as it automatically adjusted to a more useful level soon after training began. This behavior was similar for all neural-network training runs.

Figure 4 indicates that the test site accuracy increases at nearly the same rate that the mean square error decreases – quickly at first, and then slowly, but steadily right through the end. This is encouraging because it links the minimization of mean square error of the neural-network representation of the training patterns with the maximization of classification accuracy of areas not used in the training. Thus, the network has enough generalization capability to extend what it has learned about the training patterns to the rest of the image. Had the test site accuracy not increased proportionally to the decrease in mean square error, the network would have been specializing too much on the training patterns and would have been useless as a classifier for such a widely varying image.

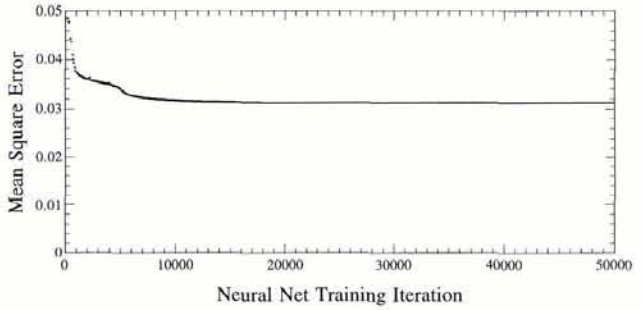
Table 3 shows the overall difference between the resulting classification maps (as a percentage of the total number of pixels) of each of the most accurate (according to test sites) runs of different hidden layer size. In the middle and upper end of the hidden layer size range, the maps differ in about 5 to 10 percent of the pixels (class labels). These are greater differences than those indicated by the test site accuracies. A visual comparison (Figures 7b through 7g) of the classification maps produced by all these networks reveals two things. First, the low accuracy of the one to three hidden-layer node networks is clear. Even in the three-node case, however, a more accurate classification is beginning to show. The second observation is that, for six nodes or more, the classifications are remarkably similar. Thus, the small differences in the test site accuracy measurements are indicative of similar classifications, though some of the class outli-



a)



b)



c)

Figure 4. One 12-node hidden-layer neural-network classifier achieved a higher test site accuracy (96.2 percent) for 50,000 iterations than any other net. The training site accuracy was near that of most of the other nets at 96.4 percent. (a) Test site accuracy vs. training iteration number, 1000 iteration intervals. The maximum-likelihood test site accuracy of 89.5 percent is shown for comparison. (b) Mean square error vs. training iteration number, 20 iteration intervals. (c) As a comparison, the mean square error vs. training iteration number (at 100 iteration intervals) is shown for a two-node hidden-layer network. This net achieves a relatively high minimum mean square error fairly early in the training process. The 12-node net, as shown in (a), on the other hand, is still decreasing in MSE at the 50,000 iteration point.

ers (which are not represented in the well-characterized test sites) will be classified with more variance from one image to the next, as indicated by the higher overall differences.

Dependence on Initial Weights

Because classifications produced by these widely different network structures are so similar, it seems that the initial randomization of the network weight values should not cause much difference from one run to the next, given a fixed hidden-layer size. However, even though resultant differences in accuracy are small, the final classifications are different and the extent of these differences needs to be examined as a function of hidden layer size. Particular attention should be paid to the test site accuracy standard deviation (third row of Table 2) and the average overall difference between each pair of maps (fourth row of Table 2) obtained for the multiple runs at each hidden layer size. Figure 6 illustrates the range of accuracy values and the average difference of the classifications obtained for each hidden layer size.

When increasing from two to 15 hidden-layer nodes, there is a significant decrease in the standard deviation of the test site accuracy over the seven runs. For six hidden-layer nodes, the test site accuracy varied from 87.3 to 94.2 percent. This difference is due entirely to the initial random condition of the network weights, as all other parameters were equal. For nine nodes, the difference decreased to about 5 percent. For 12 nodes and above, the differences were about the same (2 percent). Thus, while high accuracies are obtainable with six or nine hidden-layer nodes on any given run, it might be necessary to use more nodes to assure a result closer to the averages given in Table 2. This observation is not intuitive because the larger hidden-layer case would have more weights to randomize at the start of training. It would seem that this would induce more randomness in the procedure and produce less consistent results. A possible explanation is that the larger nets are better able to adjust their decision boundaries consistently because they have more degrees of freedom.

If we look at the average *overall* difference in classifications between pairs of runs at a given hidden-layer size, we see a slightly different result. The test-site areas were chosen because they are fairly homogeneous and highly representative of the ground-cover classes. The average difference over the entire classification, however, is a better indicator of network differences, because there are many more class outlier pixels in the entire image than in the test sites. Thus, small differences in the network weight values would be expected to manifest themselves more strongly in this measure. Figure 6b indicates that the overall classification difference drops sharply from an unacceptable 26.7 percent at three hidden-layer nodes to 9.2 percent at 12 hidden-layer nodes. It then gradually tapers off to a near constant value of 6.2 percent at 30 hidden-layer nodes. The most remarkable result is that

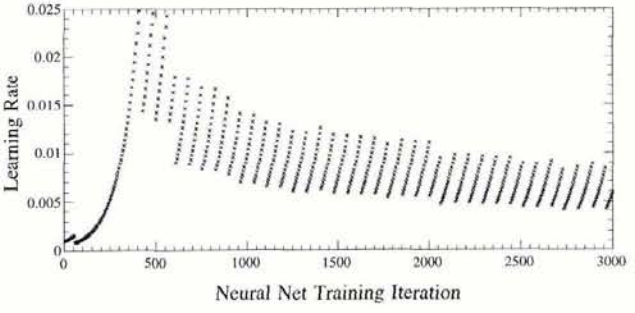


Figure 5. Learning rate vs. training iteration number (five iteration intervals) for the first 3000 iterations of the same 12-node hidden-layer network described in Figures 4a and 4b. This plot illustrates the adaptive learning rate behavior implemented to decrease training time and ensure stability of training.

TABLE 3. THE TOP TWO ROWS CONTAIN THE TRAINING AND TEST-SITE ACCURACIES OF THE MOST ACCURATE (ACCORDING TO TEST SITES) 20,000 ITERATION RUNS FOR EACH HIDDEN-LAYER SIZE. THE REMAINING ROWS SHOW THE PERCENTAGE OF PIXELS WITH DIFFERENT LABELS, BETWEEN ALL COMBINATIONS OF THE CLASSIFICATION MAPS RESULTING FROM THESE RUNS. FOR EXAMPLE, 6.2 PERCENT OF THE PIXELS WERE LABELED DIFFERENTLY IN THE 15-NODE AND THE 12-NODE RUNS. THIS RELATIVE MEASURE IS INDEPENDENT OF THE ABSOLUTE CLASSIFICATION ACCURACY PRESENTED IN THE FIRST TWO ROWS, WHICH IS BASED ON SMALL, MANUALLY INTERPRETED TEST SITES.

Nodes in hidden layer	1	2	3	6	9	12	15	18	24	30	36
(Train Site Accuracy)	32.8	82.0	83.5	95.7	95.8	96.1	96.4	96.4	96.3	96.0	95.6
(Test Site Accuracy)	24.8	79.0	83.5	94.2	95.2	94.0	94.0	93.4	93.9	93.1	93.2

Nodes in hidden layer	1	2	3	6	9	12	15	18	24	30	36
1	—	61.5	72.3	67.1	67.0	65.9	65.9	66.0	65.7	65.7	65.6
2		—	35.4	33.0	32.7	32.0	32.2	33.1	32.2	32.6	31.8
3			—	28.1	27.5	27.0	29.3	30.3	29.0	30.6	31.9
6				—	10.6	10.1	10.7	11.9	10.9	13.4	13.4
9					—	8.4	8.7	10.5	8.5	11.1	13.8
12						—	6.2	8.0	6.4	9.9	12.2
15							—	4.9	4.3	7.2	9.4
18								—	6.0	6.1	8.6
24									—	6.5	9.3
30										—	7.2
36											—

the differences *within* a hidden layer size are comparable to those *among* the different hidden-layer sizes (Table 3). Thus, the dominating factor is the initial randomization, with the size of the hidden layer, within a certain range, being a secondary effect. Figures 7e and 7f show a sample classification for two different runs of 18 hidden-layer nodes. The differences are indeed slight, and on the order of those with the six hidden-layer node example in Figure 7d.

Training and Classification Time

The network structures are compared in Table 4 in terms of the number of iterations required to achieve the same test site accuracy as the maximum-likelihood classifier (89.5 percent). This is an interesting comparison in that it shows a minimum in the middle of the configuration range at 12 hidden-layer nodes (Figure 8). It was expected that, with only a few hidden-layer nodes, it would require more iterations to achieve the same accuracy because the net has fewer degrees of freedom. It should be noted that, of the seven runs with six hidden-layer nodes, one required 42,000 iterations while the other six required an average of only 9,417 iterations. An anomalous run of this type was encountered only on this one occasion. However, it is always a possibility that the random initial weight settings will place the network in a difficult position from which to minimize the error. Presumably, the chance for this happening decreases with more hidden-layer nodes, as indicated by the decreasing standard deviation of test site accuracy in Table 2. It is also possible that optimization of the initial random weight configuration by changing the initial weight range or providing more useful initial weights (e.g., those from a similar classification or those derived from Kohonen's self-organizing algorithm — see Li and Si, 1992) would alleviate this inconsistency by starting the network off in a more favorable position. Even without this single run, however, the six-node case requires more iterations than the nine-node case, which in turn requires more than the minimal 12-node case.

The increase in the number of training iterations required to obtain a given classification accuracy from 12 to 36 hidden-layer nodes was significant, with the 36-node case requiring almost twice as many iterations as the 12-node case. The most probable explanation for this is that the training set, which consisted of 915 training patterns, required a longer time to adjust the larger number of weights in the bigger networks. Because these networks also require a larger training time per iteration, the subsequent total training time (also shown in Table 4) reflects an even larger difference. The training time for the 36-node case is four times longer than for the 12-node case. The lowest training time was achieved by the nine-node case, although the six-node case without the one anomalous run was faster. When overall training time is considered, the time savings per iteration achieved with fewer hidden-layer nodes more than makes up for the requirement of more training iterations.

When all factors are considered, the best neural-network structure for this classification seems to be that of 12 hidden-layer nodes, which provides relatively rapid training and yet delivers consistent, highly accurate test site results. The total time for training and classification in this case is 6,118 seconds (just over ten times the time required for maximum-likelihood). A case could also be made for using the smallest, fastest configuration possible. The six hidden-layer node case might occasionally take longer than the 12-node case, but most of the time it will be faster (an average of an hour for six out of the seven runs). If a threshold is placed on training site accuracy (which can be computed relatively rapidly after some interval of training iterations) during training, the iterative process can be stopped when the desired accuracy is reached. Thus, the consistency of training is not so important for obtaining accurate results. In all cases, the six-hidden-layer node network, despite its inconsistency, surpassed the baseline maximum-likelihood test site accuracy of 89.5 percent. Once training is complete, the six-node case requires only 199 seconds for classification.

Texture-Enhanced Classification

An interesting capability of the neural-network multispectral-classification method is the ease with which multiple data sources, or windows of data, can be incorporated into the classification. Textural information is a potentially useful source of additional information that might enhance the classification. One of the key characteristics of the human visual system is that spatial information is used extensively. While a computer can more easily handle high-order spectral data than does a human, it will not even begin to emulate the full capabilities of a human image interpreter without considering spatial information during classification. To this point, the classifications performed have been of the pixel-by-pixel type, in which each image pixel is considered separately. A simple way of providing some spatial texture to the classifi-

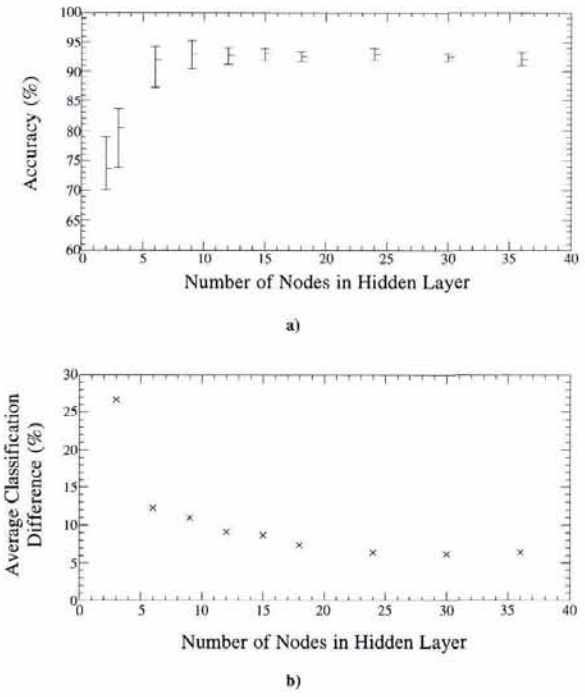


Figure 6. Two measures of classification consistency for a given hidden-layer size. The neural network was trained for 20,000 iterations at least seven times for each hidden-layer size. (a) The minimum, maximum, and average (second row of Table 1) test-site accuracies from the series of runs. (b) The average overall percentage of pixels with different labels between all pairs of classifications produced in the series of runs (fourth row of Table 2).

cation is to provide a 3 by 3 window of pixels as the input to the net.

To provide a basis for comparison, it is interesting to examine what a simple smoothing algorithm would do to the classification accuracy of a pixel-by-pixel type classification. The type of smoothing chosen was a majority filter in which the most common label in a 3 by 3 window is placed at the center. Majority filtering of the maximum-likelihood classification of the Tucson image resulted in an increase in classification accuracy from 89.5 to 92.4 percent. The same filter applied to the most accurate neural-network classification (Figures 4 and 5) resulted in an increase from 96.2 to 98.5 percent. An increase in accuracy was expected because the testing sites were assumed to be homogeneous. Thus, any smoothing will remove single-pixel errors within these regions and increase the apparent accuracy. Of course, this smoothing also reduces fine detail in the classification.

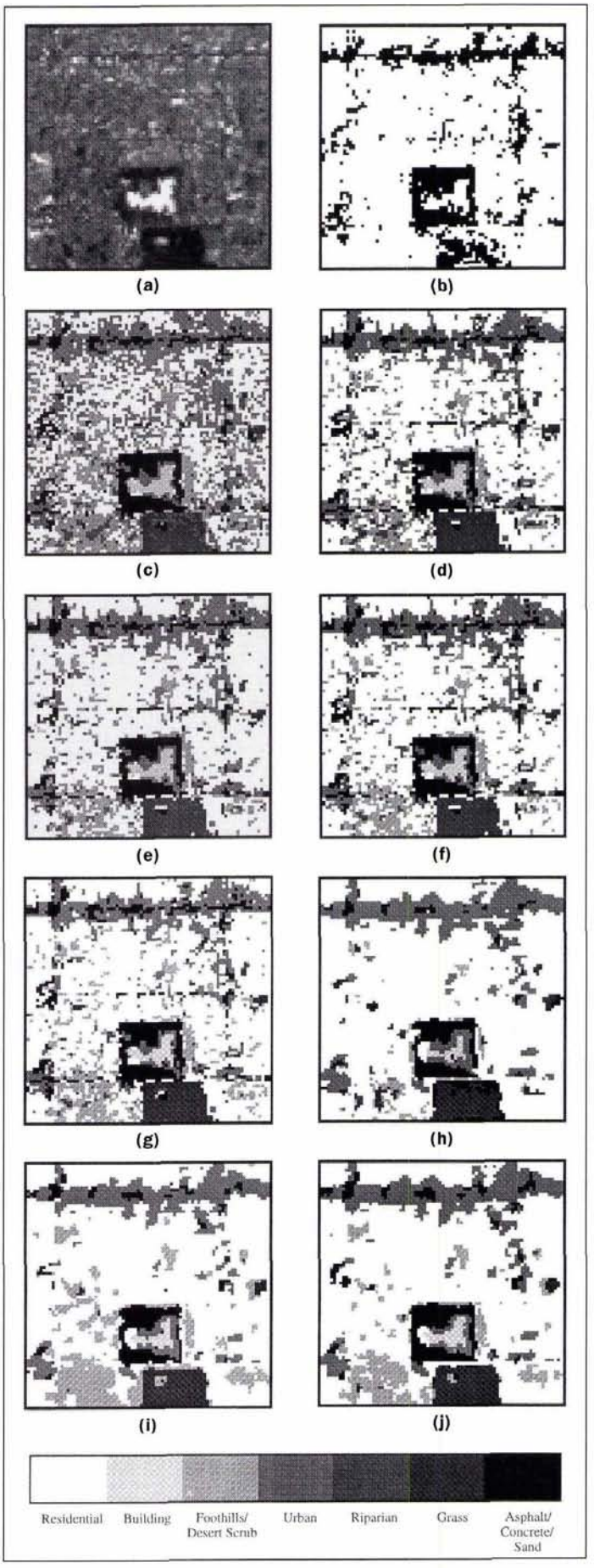


Figure 7. Classification of a portion of the Tucson TM image. The 12 classes are combined into seven gray levels to highlight the differences obtained with different hidden-layer sizes (including two different runs at one size — 18 nodes) and with a 3- by 3-window input. The feature near the bottom of the middle of the image is a large mall (El Con) with surrounding asphalt parking lot. Below the mall is a golf course (Randolph). (a) Original image (TM band 3). Pixel-by-pixel classifications, number of hidden-layer nodes: (b) 1. (c) 3. (d) 6. (e) 18. (f) 18 (different run). (g) 36. 3- by 3-window classifications, number of hidden-layer nodes: (h) 6. (i) 12. (j) 18. ——>

TABLE 4. COMPARISON OF TOTAL CLASSIFICATION TIME FOR NEURAL-NETWORK AND MAXIMUM-LIKELIHOOD CLASSIFIERS TO ACHIEVE SIMILAR ACCURACY (89.5 PERCENT). THE NETWORK VALUES REPRESENT THE AVERAGE OF SEVEN TRAINING RUNS FOR EACH HIDDEN-LAYER SIZE (EXCEPT FOR THE NET WITH HIDDEN-LAYER SIZE 18, WHICH WAS RUN 16 TIMES). THE * IN THE SIX-NODE HIDDEN-LAYER COLUMN INDICATES THE VALUES FOR THE AVERAGE OF THE SIX BEST RESULTS (OUT OF SEVEN). ONE RESULT IN THE SIX-NODE CASE WAS ANOMALOUS AND REQUIRED 42,000 ITERATIONS TO ACHIEVE THE MAXIMUM-LIKELIHOOD ACCURACY, ILLUSTRATING THE POTENTIAL INCONSISTENCY THAT CAN ARISE WHEN VERY FEW NODES ARE USED.

	M-L	6	9	12	15	18	24	30	36
Number of iterations required	N/A	14071/9417*	8643	7786	8714	8600	9214	10429	11357
Training Time (seconds)	negligible	6008/4021*	5087	5824	7941	9221	12859	17926	23211
Classification Time (seconds)	590	199	245	294	339	385	480	574	668
Total Time (seconds)	590	6207/4220*	5332	6118	8280	9606	13339	18500	23879

The neural-network classification was expanded to include a 3 by 3 window of pixels in each band of the image as input into the network. The 3 by 3 window results in a total of 54 inputs for the six-band non-thermal TM classification. Several training runs were carried out with various hidden-layer sizes. The hidden-layer sizes were kept in the same range as those of the original classification. Incorporation of the 3 by 3 window resulted in fewer training patterns than for the original classification. The window was required to fit within each training region, thus excluding the region edge pixels from the training set (except as neighbor pixels in the window). This resulted in a total of 443 training patterns (about half of the original number). Table 5 summarizes the test site accuracy and classification times for all the 3- by 3-window runs.

The surprising result of the runs summarized in Table 5 is that, in order to achieve the baseline 89.5 percent accuracy, many of them required less training time than the pixel-by-pixel networks discussed previously (see Table 4). The increased cost per training iteration brought about by the greater number of inputs apparently is more than offset by faster convergence (fewer iterations). However, the number of training iterations required to achieve the same accuracy varied considerably from one run to the next as shown in Table 5. This large variation is due to the much larger number of interconnecting weights that must be randomly initialized before training relative to the pixel-by-pixel (six-input) networks examined in the previous sections.

Table 6 shows the percent differences in the classifications of the various 3- by 3-window network configurations of Table 5 (only the most accurate six hidden-layer node net is shown). Visually, all of the maps produced with more than six hidden-layer nodes are comparable (Figures 7h through 7j). It appears from these results that there is a slight correlation between the number of inputs and number of hidden-layer nodes required for a given accuracy. The 54 input nets required slightly more hidden-layer nodes to produce classifications comparable to those using six input nodes. Another observation is that, although the maps appeared very similar and had high classification accuracies, the percent difference measurements of Table 6 are significantly greater both in repeated trials at the same hidden-layer size and between different hidden-layer sizes than the pixel-by-pixel networks of Table 3. This indicates that the results are not as consistent, and the initial randomization of the weights has considerable effect in this case.

Figure 9 shows the mean square error of network training as a function of training iteration for the most accurate 18-node hidden-layer network (seventh row of Table 5). Notice how the mean square error decreases more sharply before leveling off than for the pixel-by-pixel network as shown in Figure 4. Perhaps there is a relationship between the rate of the error minimization and the number of input parameters. If the network is provided with more information about the image in a single training iteration, it can make more progress towards the minimum error. Unfortunately, the time savings gained is offset somewhat by the much greater classification time. Because a 3 by 3 window must be examined for every pixel in the image during classification, this time is greatly increased. However, the total classification time is still less than that of the pixel-by-pixel network for many of the runs.

In addition to the training being a little faster, the 3- by 3-window input networks may be more capable of separating the classes than the pixel-by-pixel network. While the incorporation of texture information is no doubt helpful, the exact effect of this information is hard to evaluate without an extensive study involving much more ground truth about the actual class composition of the image. However, the accuracy of the small test sites was consistently higher for the window classifications than for the original classification. The majority filter could be used to enhance the apparent accuracy of the pixel-by-pixel classifier by virtue of its smoothing property. It should be noted that the window classifications, although they already have a smoothed appearance, can also be majority filtered, thereby increasing their apparent accuracy as well. And while the window classifications appear smoothed, there are many places where single pixels have been assigned a class different from neighboring pixels (see Figures 7h through 7j). The application of a smoothing filter will eliminate such pixels, thus destroying any fine detail in the classification. The window input net has the advantage of producing more homogeneous classes while preserving some fine detail.

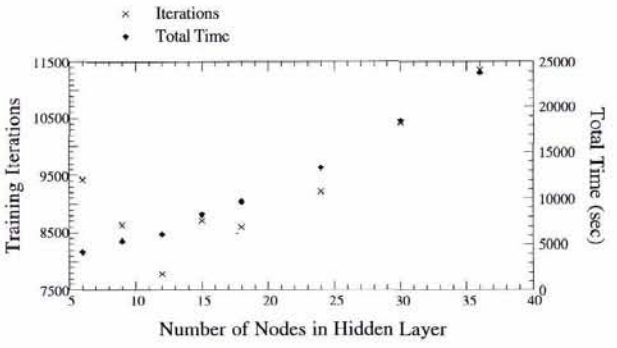


Figure 8. Average number of training iterations and total classification time (training plus classification) required for a neural network to achieve the same accuracy as maximum-likelihood (89.5 percent) versus hidden-layer size. The neural network was trained for 20,000 iterations at least seven times for each hidden-layer size, except for the six hidden-layer node case, where a single anomalous training run has been left out.

TABLE 5. EVALUATION OF NEURAL NETWORK CLASSIFICATION USING A 3 BY 3 WINDOW OF INPUT PIXELS IN EACH BAND FOR THE TUCSON SIX-BAND IMAGE. THE NETWORK HAD 54 INPUTS AND 12 OUTPUTS AND WAS TRAINED WITH 443 TRAINING PATTERNS. THE ACCURACY IS COMPARED TO THAT OF THE MAXIMUM-LIKELIHOOD BASELINE VALUE OF 89.5 PERCENT.

Number of hidden layer nodes	Accuracy at 25,000 iterations (%)	Training time (sec/iteration)	Feed-forward Classification time (sec)	Iterations to achieve ML accuracy (89.5%)	Time to achieve ML accuracy (sec)
3	83.5	0.3380	606	N/A	N/A
6	93.5	0.6080	756	4000	2432
6	96.2	"	"	4500	2736
6	95.2	"	"	4000	2432
12	94.6	1.1336	1042	3500	3968
12	92.9	"	"	6500	7368
18	96.7	1.6777	1335	4500	7550
18	96.5	"	"	3000	5033
18	96.3	"	"	3500	5872

TABLE 6. THE PERCENTAGE OF PIXELS WITH DIFFEENT LABELS, BETWEEN ALL COMBINATIONS OF THE CLASSIFICATION MAPS OF THE MOST ACCURATE 3- BY 3-WINDOW NETWORKS FROM TABLE 5. THESE VALUES WERE CALCULATED IN THE SAME WAY AS THOSE OF TABLE 3.

Hidden Layer size	3	6	12	12	18	18	18
3	—	43.0	39.0	35.9	41.4	44.6	40.1
6		—	24.5	28.6	26.8	23.7	25.1
12			—	15.9	14.6	20.9	14.9
12				—	16.1	20.8	13.7
18					—	11.9	10.8
18						—	15.4
18							—

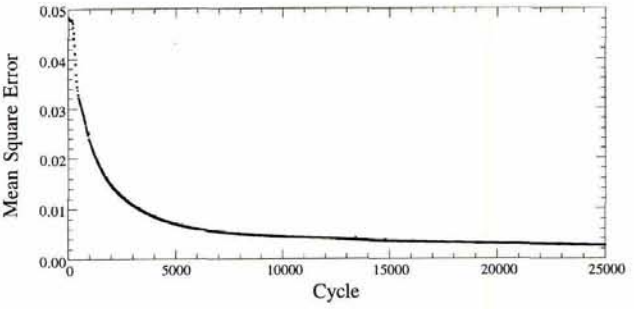


Figure 9. A plot of the mean square error of training vs. training iteration number for a 3- by 3-window input net with 18 hidden-layer nodes. It has a very sharp drop in mean square error in the beginning of training (compare to pixel-by-pixel net mean square error plot of Figure 4).

Summary and Conclusions

Experiments aimed at understanding the behavior of a neural-network classifier, as a function of its key parameters, are described in this paper. The test image was a Landsat TM scene of a complex urban area and Level I and II land-use classes were defined for supervised classification. In the first experiment, a neural-network multispectral-image classifier was trained multiple times with the number of hidden-layer nodes varying from one to 36. This was of interest for two reasons. First, the best structure in terms of accuracy and speed could be determined. The most accurate training run was obtained using 12 hidden-layer nodes, but the fastest classifications that achieved the accuracy of maximum-likelihood could be performed with only six or nine hidden-layer nodes.

The second reason was to determine if the initial randomization of the network weights resulted in significantly different classifications from one run to the next. It was found that substantial differences could occur, especially for low numbers of hidden-layer nodes. In fact, the differences in the classification maps produced at a given hidden-layer size were comparable to the differences produced among different hidden-layer sizes. Thus, the initial randomization is a primary factor, with the hidden-layer size over the range of six to 36 nodes being a secondary effect. However, the initialization effect becomes smaller with more hidden-layer nodes. The variance of both the test site classification accuracy and the overall classification difference decreased with more hidden-layer nodes. Thus, while more nodes take longer to train, the results are more consistent.

Another significant result was that, although the number of hidden-layer nodes could be optimized in terms of classification accuracy or training or classification speed, this was not particularly necessary. Any hidden-layer size greater than three produced adequate classification maps after a similar number of training iterations. Thus, the type of network used here is not very sensitive to the number of hidden-layer nodes for this classification application.

The structure of the neural-network makes it very easy to incorporate ancillary data or spatial information. The second experiment involved the use of 3 by 3 windows of image data as input to the neural-network classifier. The resulting classifications have a smoothed appearance relative to the pixel-by-pixel classifications. There is some single-pixel classification detail evident, however, so the network is not merely smoothing the original classification. A surprising result was that the total network classification time required to achieve the accuracy of maximum-likelihood is less in many cases than for the pixel-by-pixel classification nets. Even though the time per training iteration is much greater, it appears that the information introduced in the 3 by 3 window allows for faster convergence. It was also found that the number of hidden-layer nodes did not need to be increased dramatically with the much larger input of a 3 by 3 window of multispectral data (an increase from six to 54 values) to achieve an accurate classification. However, the consistency of the training is much less than that of the pixel-by-pixel classification, most likely due to the much larger number of weights subject to initial randomization.

This experiment, although limited to a single classification example, illustrates many of the problems one encounters using a neural network for multispectral classification. By restricting the experiments to one image, we eliminate any uncertainties that might originate from scene-dependent factors. Our intent is to demonstrate the characteristics of neural networks in classification; similar experiments are easily performed (and recommended) to insure the quality of neural-network classifications of other images. The hidden-layer size dilemma is well-documented in neural-network lit-

erature. However, the inconsistency of training is an often overlooked potential problem. The anomalous run taking five times as long as most of the others is a good example of this potential for trouble. It shows that the selection of a suitable hidden-layer size must be based on more than a single trial run at each size. This is especially true when a larger network (such as one with 3- by 3-window inputs) is used.

Acknowledgments

This research was partially supported by NASA under Contract NAS 2 13721 to the Universities Space Research Association, Research Institute for Advanced Computer Science (RIACS), NASA Ames Research Center, and under High Performance Computing and Communications grant NAG 5 2198 to the University of Arizona. We also wish to acknowledge the encouragement and assistance provided by our colleague, Dr. Marjory Johnson of RIACS.

References

Anderson, J.R., E.E. Hardy, J.T. Roach, and R.E. Witmer, 1976. *A Land Use and Land Cover Classification System for Use With Remote Sensor Data*, USGS Professional Paper No 964, U.S. Geological Survey, U.S. Government Printing Office, Washington, D.C.

Benediktsson, J.A., P.H. Swain, and O.K. Ersoy, 1990. Neural Network Approaches Versus Statistical Methods in Classification of Multisource Remote Sensing Data, *IEEE Transactions on Geoscience and Remote Sensing*, 28(4):540–551.

Bischof, H., W. Schneider, and A.J. Pinz, 1992. Multispectral Classification of Landsat Images Using Neural Networks, *IEEE Transactions on Geoscience and Remote Sensing*, 30(3):482–490.

Blonda, P., V. la Forgia, G. Pasquariello, and G. Satalino, 1994. Multispectral Classification by a Modular Neural Network Architecture, *1994 International Geoscience and Remote Sensing Symposium*, Pasadena, California, 8–12 August, pp. 1873–1876.

Fierens, F., I. Kanellopoulos, G.G. Wilkinson, and J. Mégier, 1994. Comparison and Visualization of Feature Space Behavior of Statistical and Neural Classifiers of Satellite Imagery, *1994 International Geoscience and Remote Sensing Symposium*, Pasadena, California, 8–12 August, pp. 1880–1882.

Heermann, P.D., and N. Khazenie, 1992. Classification of Multispectral Remote Sensing Data Using a Back-Propagation Neural Network, *IEEE Transactions on Geoscience and Remote Sensing*, 30(1):81–88.

Kanellopoulos, I., G. G. Wilkinson, and J. Mégier, 1993. Integration of Neural Network and Statistical Image Classification for Land Cover Mapping, *Proceedings, 1993 International Geoscience and Remote Sensing Symposium*, Tokyo, Japan, August, pp. 511–513.

Key, J., J.A. Maslanik, and A.J. Schweiger, 1989. Classification of Merged AVHRR and SMMR Arctic Data with Neural Networks, *Photogrammetric Engineering & Remote Sensing*, 55(9):1331–1338.

———, 1990. Neural Network vs. Maximum Likelihood Classifications of Spectral and Textural Features in Visible, Thermal, and Passive Microwave Data, *Proceedings, 10th Annual International Geoscience and Remote Sensing Symposium*, College Park, Maryland, May, pp. 1277–1280.

Li, R., and H. Si, 1992. Multi-spectral Image Classification Using Improved Backpropagation Neural Networks, *Proceedings, 12th Annual International Geoscience and Remote Sensing Symposium*, Houston, Texas, May, pp. 1078–1080.

Lippmann, R.P., 1987. An Introduction to Computing with Neural Networks, *IEEE ASSP Magazine*, April, pp. 4–22.

Liu, Z.K., and J.Y. Xiao, 1991. Classification of Remotely-Sensed Image Data Using Artificial Neural Networks, *International Journal of Remote Sensing*, 12(11):2433–2438.

Paola, J.D., and R.A. Schowengerdt, 1995a. A Detailed Comparison of Backpropagation Neural Network and Maximum Likelihood Classifiers for Urban Land Use Classification, *IEEE Transactions on Geoscience and Remote Sensing*, 33(4):981–996.

———, 1995b. A Review and Analysis of Backpropagation Neural Networks for Classification of Remotely Sensed Multispectral Imagery, *International Journal of Remote Sensing*, 16(16):3033–3058.

Rumelhart, D.E., G.E. Hinton, and R.J. Williams, 1986. Learning Internal Representations by Error Propagation, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations* (D.E. Rumelhart and J.L. McClelland, editors), The MIT Press, Cambridge, Massachusetts, pp. 318–362.

Yoshida, T., and S. Omatu, 1994. Neural Network Approach to Land Cover Mapping, *IEEE Transactions on Geoscience and Remote Sensing*, 32(5):1103–1109.

(Received 11 December 1995; accepted 12 August 1996; revised 1 November 1996)