2002

Landsat image classification using a neuro-fuzzy system

Jian Zheng
San Jose State University

Landsat Image Classification Using a Neuro-Fuzzy System

A Thesis

Presented to

The Faculty of the Department of Geography

San Jose State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Arts

by

Jian Zheng

August 2002

UMI Number: 1410455

UMI®

UMI Microform 1410455

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

APPROVED FOR THE DEPARTMENT OF GEOGRAPHY

Dr. Richard Taketa

Dr. David Schwarz

Ms. Susan Benjamin

APPROVED FOR THE UNIVERSITY

ABSTRACT

LANDSAT IMAGE CLASSIFICATION USING A NEURO-FUZZY SYSTEM

by Jian Zheng

This study investigates an alternative classification algorithm, NEFCLASS, and its ability to classify remote sensing images. NEFCLASS is a Neuro-fuzzy System that is capable of generating a set of linguistic rules. These rules allow the user to check and interpret the classification results. This study also shows that the neural net rules stabilized after only a few training iterations. The land-use/land-cover classification result produced by NEFCLASS is compared to the result produced by a conventional classification algorithm, Maximum Likelihood Classifier (MLC). NEFCLASS produced better classification accuracy than MLC.

Table of Contents

List of Tables

List of Figures

Introduction

As the human population grows and the economy expands, we put increasing demands on the environment. Our demands for more natural resources, such as fresh water and raw materials, have outpaced the natural speed of recovery. To assess the impact of human development on the ecosystem, the Federal and local governments in the United States have been conducting studies, such as modeling nutrient runoff and developing land use policies. These studies require current, regional to national scale land-use/land-cover maps. To provide the necessary map information, the United States Geological Survey (USGS) Multiresolution Land Characterization Consortium (MRLC) recently has produced a 30-meter resolution land-use/land-cover map of the conterminous U.S.

The project was divided into smaller units due to the vastness of the study area. Each study unit contained multiple Landsat Thematic Mapper (TM) images, covering diverse geological and ecological terrain. Capturing all the TM images on the same day was impossible. Often, the adjacent images were taken in different years and at different points of the growing cycle. Even though the images were matched using the histogram equalization method, the transition between scenes was often visible. Furthermore, the large data set required many people to work on it. The quality of the classification was strongly influenced by the operator's experience and his/her familiarity with the area. The compounded errors produced by these problems and the intrinsic complexity of the data resulted in laborious post-classification editing and inter-study-unit edge matching.

In the last decade, researchers have studied the effectiveness of using Neural Networks (NNs) to analyze remotely sensed data. Researchers have found the NNs to be computationally expensive, but they give superior results compared to the conventional classification algorithms, such as Maximum Likelihood Classification (MLC). A shortcoming of the NNs is that they are a black box. The program cannot communicate the details of the classification decision criteria to the user. This study will examine an algorithm called NEFCLASS, a Neuro-fuzzy system (a hybrid of NNs and Fuzzy Logic). This method is different from conventional NNs due to its ability to produce a set of decision rules, which allow users to examine the validity of the classification. In addition, this method could remedy the inconsistency of human involvement by incorporating more training data. This paper explores this Neuro-fuzzy classification method by classifying complex desert landscape using satellite images.

Land-Cover/Land-Use Classification

Classification is a process of transforming data into information. The information and spectral classes are fundamentally different (Jensen, 1996, p. 200). Spectral classes are defined based on the statistical distribution of the remotely sensed data. From here on, spectral classes will be referred to as "clusters." Information classes, on the other hand, are defined by the user. In this study, the user is interested in the vegetation on the ground and the usage of the land. Therefore, the information classes consist of the vegetation types and various human developments. The objective of this study is to analyze and convert spectral data into information classes.

When designing a classification system, one should carefully define each class and make certain that all the classes in the system are mutually exclusive. Furthermore, one should also be clear on the difference between land-cover and land-use. Land-cover classes describe the geologic condition or the vegetation on the ground. Land-use classes define how the land or the plant life is being utilized. For example, the feature we call "grass" in a strict land cover classification system can include natural grass, hay/pasture, small grain, or other urban grasses. The MRLC project uses a combination of land-cover and land-use classification systems shown in Table 1.

Aerial photographs are systematically selected for the study area to be used as reference data. Ground truth data are collected and marked on the aerial photographs to help the interpretation and labeling of the classes. Back in the lab, the ground truth data are compared to the TM images and the cluster images. Some of the clusters represent only one class of land cover. However some clusters include multiple land cover classes. In that case, "cluster busting", separating the cluster into various classes, will occur. The operator looks for patches of this cluster in the image, using the ground truth data to identify the land cover of the patches, then "spike" a number of pixels to get the values from each data layer. The operator or a computer algorithm then creates a model based on the differences of the observed values. This model is then applied to the image. If the model does not successfully separate the cluster, the "spiking" is repeated, until a good break-up value is found to separate the cluster with an acceptable amount of error.

Table 1. MRLC Regional Land-Cover, Land-Use Classification System

Water
- 11. Open Water
- 12. Perennial Ice/Snow

Developed
- 21. Low Intensity Residential
- 22. High Intensity Residential
- 23. High Intensity Commercial/Industrial/Transportation

Barren
- 31. Bare Rock/Sand/Clay
- 32. Quarries/Strip Mines/Gravel Pits
- 33. Transitional

Forested Upland (non-wet)
- 41. Deciduous Forest
- 42. Evergreen Forest
- 43. Mixed Forest

Shrubland
- 51. Deciduous Shrubland
- 52. Evergreen Shrubland
- 53. Mixed Shrubland

Non-Natural Woody
- 61. Planted/Cultivated (orchards, vineyards, groves)

Natural/Semi-natural Herbaceous Vegetation
- 71. Grasslands

Planted/Cultivated Vegetation
- 81. Pasture/Hay
- 82. Row Crops
- 83. Small Grains
- 84. Bare Soil
- 85. Other Grasses (parks, lawns, golf courses)

Wetlands
- 91. Woody Wetlands
- 92. Emergent Herbaceous Wetlands

In some cases, different land use classes can be separated based on anthropomorphic practices. For instance, watering of cropland and park gives these features a distinctive reflectivity in the early growing season. However in many instances, separating pixels solely based on their spectral values is impossible. In that case, the appropriate ancillary data source will be used to resolve the confused classes. For example, shaded areas are often confused with water bodies. To separate these two classes, a Digital Line Graph of hydrological features might be useful. Another example would be separating golf courses from hay/pasture, where the two classes have very similar spectral signatures. Manual digitizing of golf courses is necessary to solve the confusion.

Image Classification Algorithms

Classification may be performed using a variety of supervised or unsupervised algorithms. Supervised classification involves the use of *a priori* information to group pixels into pre-defined information classes. Unsupervised classifications are performed without any knowledge about the area under study. Instead, pixels with similar spectral values are grouped into clusters. The clusters do not have meaning associated with them. The operator then needs to identify the features represented by each cluster. This study will focus on supervised classification methods.

Supervised classification algorithms can be categorized as either parametric or nonparametric. Parametric classification makes predictions based on a particular statistical distribution of a population, most commonly the Gaussian probability distribution. In contrast, nonparametric methods do not rely on statistical parameters, such as the mean or the standard deviation, to describe the probability distribution (Schowengerdt, 1997). Nonparametric methods are particularly suitable in cases where the sample size is small, such that the distribution of the population is unreliable or the underlying distribution of the population is unknown.

In this study, a nonparametric supervised method, NEFCLASS — the Neuro-fuzzy classifier, will be examined for land cover classification. This will be compared to the Maximum Likelihood Classifier (MLC), which is a standard parametric algorithm that can be trained using the same set of data. The results obtained from both methods will then be compared to the ground truth information.

Conventional Classification Method – Maximum Likelihood Classifier

Each pixel in a remotely sensed image contains a number of measurements. Therefore, each pixel can be represented as a vector, $X_{i,j} = [\ v_1, v_2, v_3, \ldots, v_n]$, where i and j represents row and column, respectively. The same type of vector data is used as input to both the MLC and the NNs.

The MLC algorithm delineates the boundary of each class by obtaining statistics from the training data vectors. The mean measurement vector M_c and the covariance matrix Cov_c of each class are incorporated in the decision rule. The algorithm assigns each pixel or feature vector to the most probable class (Jensen, 1996). The basic MLC equation assumes that a pixel has an equal probability of belonging to any of the classes, and that the input data are normally distributed.

The following equation for the maximum likelihood classifier is obtained from the ERDAS Field Guide, a manual for the Imagine image processing software.

$$D = \ln(a_c) - [0.5\ln(|Cov_c|)] - [0.5(X - M_c)^T(Cov_c^{-1})(X - M_c)]$$

Where:

D	= weighted distance (likelihood)
c	= a particular class
X	= the measurement vector of the candidate pixel
M_c	= the mean vector of the sample of class c
a_c	= percent probability that any candidate pixel is a member of class c (defaults to 1.0, or is entered from *a priori* knowledge)
Cov_c	= the covariance matrix of the pixels in the sample of class c
$\|Cov_c\|$	= determinant of Cov_c
Cov_c^{-1}	= inverse of Cov_c
ln	= natural logarithm function
T	= transposition function

This algorithm's performance will suffer if the data are not equally distributed among the classes or if the training data are multimodal. In case of unequal distribution, the *a priori* information can be incorporated into the algorithm as weights if the operator knows the distribution of the data ahead of the time. When weights of each class are considered in the equation, the equation is called the Bayes' decision rule. In cases where the data are multimodal, better results can be achieved by subdividing a class to obtain unimodal, normal distributions.

Neural-fuzzy System

Neural Networks

The concept of Artificial Neural Networks is inspired by biological neurons. Neurons are the basic units in a nervous system. They form the neural pathways, which receive, process, and transmit electrochemical signals. Each neuron consists of dendrites, a cell body, the axon, and the axonic ending. The dendrites are a branchlike structure, that receive signals from other neurons. The cell body sums the signal, and, when excited above a threshold, sends a signal down the axon. This signal is then received by the dendrites of the adjacent neurons.

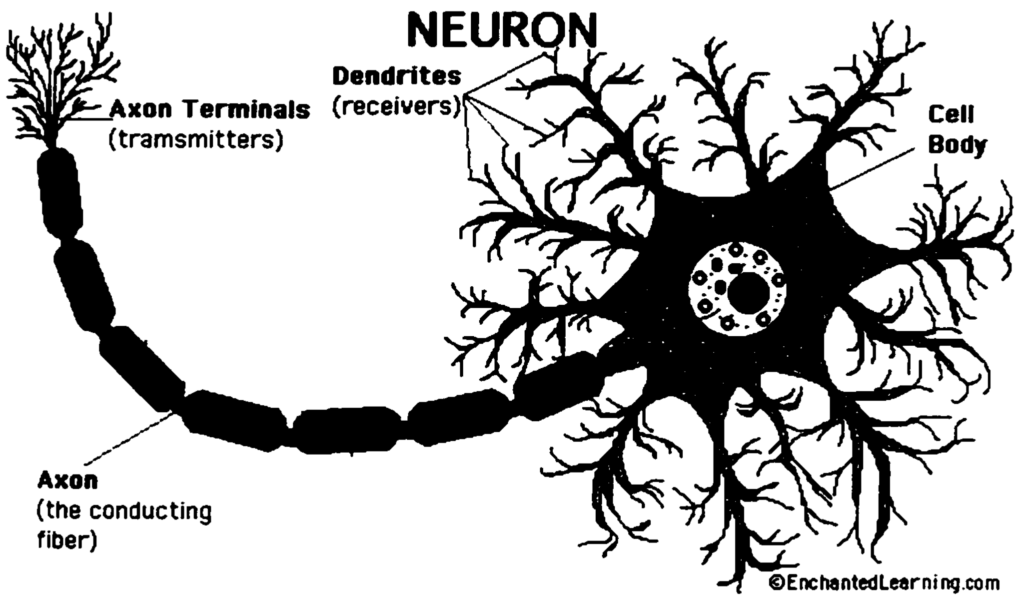


Figure 1. A Neuron

The typical computing speed of a neuron in a human brain is a few milliseconds, where the typical computing speed in computer circuits is on the order of microseconds (Kulkarni, 1994). However, the human brain is capable of processing visual and auditory signals much faster than computers. This fast problem solving can be attributed to the massive parallel nerve system. Neural Networks are designed to emulate this architecture using electronic circuits or computer algorithms.

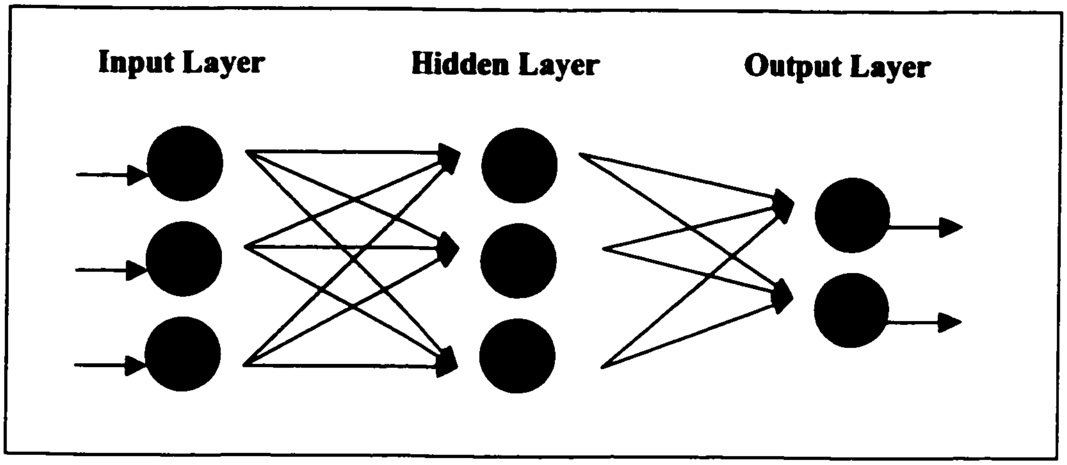


Figure 2. A Generic Model of a Three-layered Neural Networks.

Our understanding of how the nervous system functions is still incomplete. Researchers are trying to formulate the mechanisms of biological neural networks. Currently several models of NNs are in existence. The three-layered perceptron (consisting of an input layer, a hidden layer, and an output layer) is the most common architecture. The number of nodes to incorporate in each hidden layer and even the number of hidden layers to include in the NNs can vary. The more nodes and hidden layers in NNs, the more complex the class boundaries are. For example, two or more hidden layers can create discontinuous class boundaries. The user can decide on the size of the hidden layer based on their knowledge of the input data and their experience with using the NNs. However no known scientific method can be used to determine the optimal number of nodes or layers to incorporate in a network.

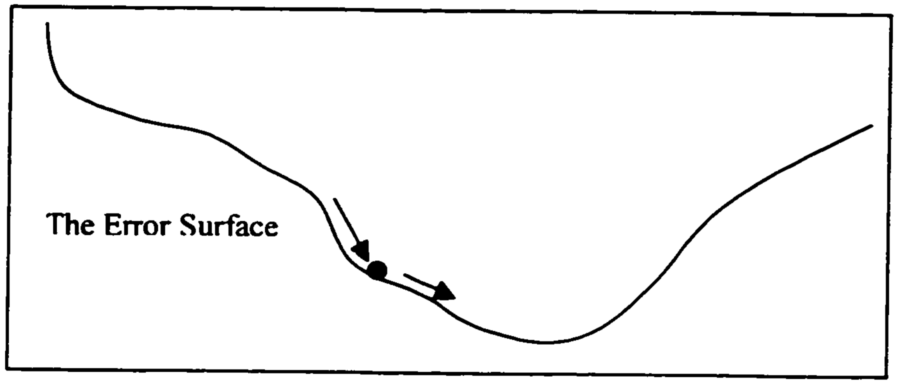


Figure 3. A Graphic Representation of the Error Surface: a depiction of the Gradient Descending Nature of the Back-propagation Algorithm

The back-propagation algorithm is a popular learning algorithm used with the 3-layered perceptron. The typical back-propagation algorithm belongs to the family of iterative, gradient descending algorithms. The term "gradient descending" describes the error minimization process during training (Figure 3). For example, larger errors will have greater effect on the weights, therefore the NNs descend more quickly towards the minimum of the error surface.

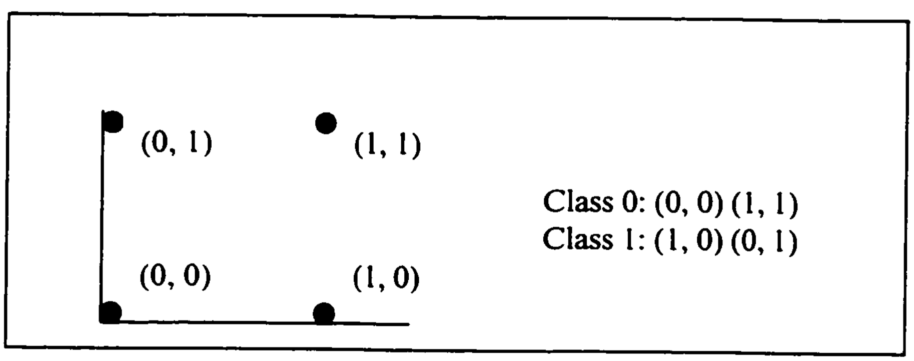


Figure 4. A Graphic Representation of the XOR Problem

To illustrate the concept of back-propagation, a simple "exclusive or – XOR" problem can be useful (Figure 4). In this problem, there are 2 classes, true (class 1) and false (class 0). Class 0 contains (0, 0) and (1, 1), and class 1 contains (1, 0) and (0, 1). The goal is to find a set of weights for the NNs that can correctly solve the XOR problem. To achieve this goal, the network will go through numerous cycles of training using the back-propagation algorithm. In this example (Figure 5), the network has only 2 input nodes, 2 hidden layer nodes, and 1 output node verses 3 input nodes, 3 hidden layer nodes, and 2 output nodes in the generic NNs model shown in Figure 2. The size of the network can vary depending on the complexity of the problem. The following steps describe the training process.

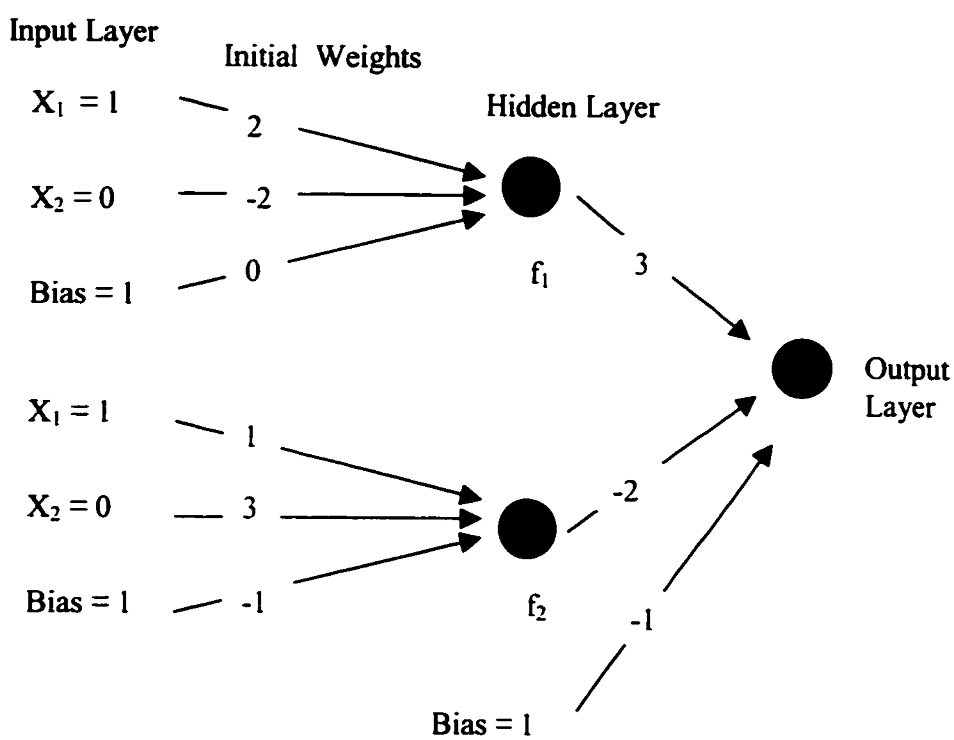


Figure 5. A 3-layered Neural Networks and Initial Weights

1) Select a training pattern and specify the desired output. For example, an input from the XOR problem can be (1, 0) and the desired output is 1.

2) Initialize weights to small random numbers.

3) Propagate the training data forward through the NNs.

The sigmoid function, $1 / (1 + e^{-S})$, is often used in the back-propagation model. S is the sum of the products of the inputs and the weights.

From the input layer to the hidden layer:

$S = X_1 * weight_1 + X_2 * weight_2 + Bias * weight_3$

$S_1 = 1*2 + 0*(-2) + 1*0 = 2,$

$S_2 = 1*1 + 0*3 + 1*(-1) = 0,$

Apply S to the sigmoid function, $f = 1 / (1 + e^{-S})$:

$f_1 = 1 / (1 + e^{-2}) = 0.881$

$f_2 = 1 / (1 + e^{0}) = 0.5$

From the hidden layer to the output layer:

$S = f_1*3 + f_2* (-2) + Bias*(-1)$

$= 0.881*3 + 0.5*(-2) + 1*(-1) = 0.643$

$output = 1 / (1 + e^{-0.643}) = 0.665$

4) Accumulating the total error relative to the desired output and back-propagate the error through the layers.

Output error: $\delta = output * (1 - output) * (t - output)$, where t is the desired output.

$\delta = 0.665 * (1 - 0.665) * (1 - 0.665) = -0.219$

Error at the hidden layer: $\delta_n = f_n * (1 - f_n) * \Sigma w*\delta$, where w is the weight between the hidden layer and the output layer.

$\delta_1 = 0.881 * (1 - 0.881) * (3 * (-0.219)) = -0.069$;

$\delta_2 = 0.5 * (1 - 0.5) * ((-2) * (-0.219)) = 0.110$.

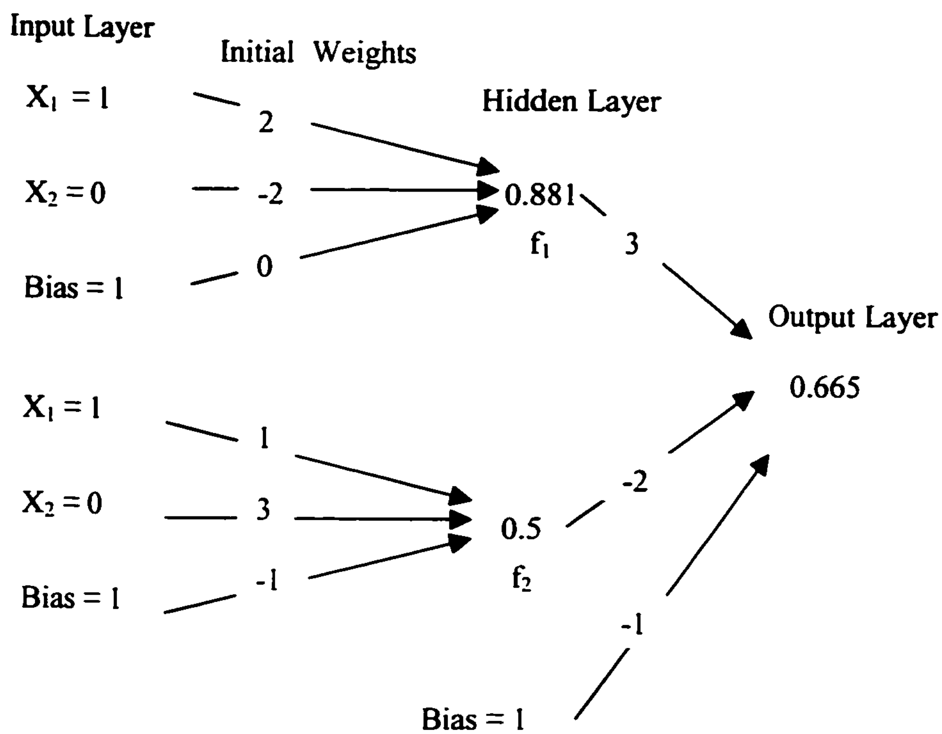


Figure 6. A 3-layered Neural Networks and its First Output: after propagating the input through the network using the initial weights, the output is 0.665. The desired output is 1.

5) Adjusting the weights:

$weight_{new} = weight_{old} + \delta_n * X_n$

The new weights between the first input node and the hidden layer:

2 + (-0.069) * 1 = 1.953;

(-2) + (-0.069) *0 = -2.069;

0 + (-0.069) * 1 = -0.069;

The new weights between the second input node and the hidden layer:

1 + 0.110 * 1 = 1.110;

3 + 0.110 * 0 = 3.110;

(-1) + 0.110 * 1 = 0.890;

The new weights between the hidden layer and the output layer:

3 + (-0.219) * 0.881 = 2.807;

(-2) + (-0.219) * 0.5 = -2.110;

(-1) + (-0.219) * 1 = -1.219.

6) Select another input pattern from the XOR example and repeat steps 3 through 5 using the new weights. Every input pattern will produce errors; therefore the weights will alter every time. After all the input patterns have been through the NNs once, an epoch or iteration is complete.

7) Repeat as many epochs as the operator specifies or until the classification accuracy reaches a certain threshold.

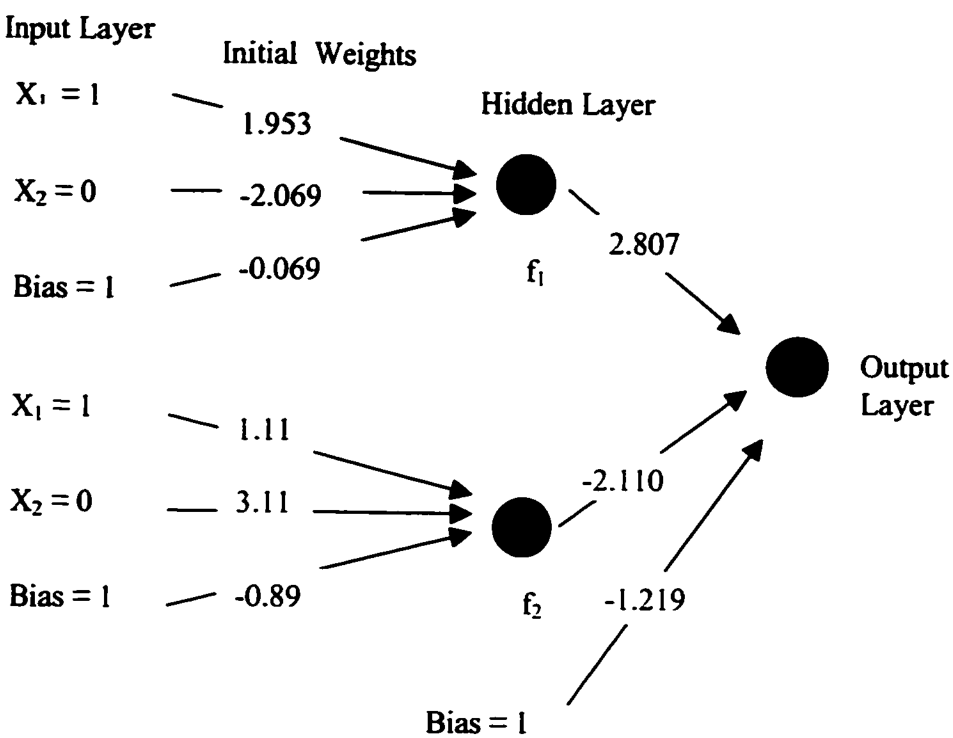


Figure 7. A 3-layered Neural Networks after Error Back-propagation: the weights between the layers have been updated.

Once training is complete, the classification phase goes into a feed-forward mode (Paola and Schowengerdt, 1994). Consequently the NNs operate like a hard-wired circuit to produce the classification. The entire image is fed into the network pixel-by-pixel, and the network output should be between 0 and 1. A simple threshold is applied to the output to make a class selection for each pixel (Paola and Schowengerdt, 1995a, p. 982).

The accuracy of the classification results generated by NNs has varied greatly. Most of the authors found that the NNs perform better than conventional classification algorithms. Paola and Schowengerdt (1995a) and Foody et al. (1995) attributed the better performance of the NNs to the fact that they are nonparametric. Since remotely sensed data rarely have a normal distribution, parametric algorithms are less accurate than NNs at predicting class membership. Paola and Schowengerdt (1994) suggested that given *a priori* distribution for the data, MLC would perform as well as the Neural Networks. Bruzzone et al. (1997) tested this claim by modifying MLC with *a priori* probabilities. In his study, NNs still significantly out performed the modified MLC. The NNs are also better able to differentiate classes with widely different variances, which can cause problems for the MLC (Paola and Schowengerdt, 1994). In practical use, NNs are relatively tolerant of missing data and noise within the data (Hepner et al., 1990).

Fuzzy Logic

In conventional computer logic, an evaluation can result in either true or false, depending on the given criteria. The shortcoming of this method is that it does not accommodate terms that involve degrees of intensity, such as height or speed. Similar input values can be separated into different categories based on rigid evaluation criteria. For example, if we define 180 centimeters or more to be tall, then someone who is 180.5 centimeters would get classified as tall, yet someone who is 179.5 centimeters would not be classified as tall.

Fuzzy logic, on the other hand, allows for imprecise description of conditions. One can specify a membership function, which defines the distribution or degree of truth of a variable to each class. The fuzzy decision rules are often generated based on past experience. All possible input-output relationships in fuzzy terms need to be specified in a rule set. These rules are expressed with if-then statements (Kartalopoulos, 1996). For instance, we have two inputs, A and B, each with three conditions. There would be, at maximum, 9 rules to define all the possible combinations of the conditions in if-then terms:

If A1 and B1, then C1, else
If A1 and B2, then C2, else
If A1 and B3, then C3, else
If A2 and B1, then C4, else
...
If A3 and B3, then C9.

The results, the C terms, can also be fuzzy terms, such as a percentage of membership to each class. In some cases, defuzzification of the fuzzy statement will take place. The most commonly used techniques are maximizer, which takes the output that has the maximum value; weighted average, which uses the average of the weighted possible outputs; and centroid, which finds the output's center of mass.

Neuro-fuzzy

The advantage of using Neural Networks and fuzzy systems is that neither needs a mathematical model to solve a problem. However both methods have shortcomings. Neural Networks are black boxes. The user cannot check or interpret the solution. Fuzzy systems do not possess the ability to learn. The advantage of combining NNs and fuzzy systems is that Neural-fuzzy Systems have the ability to learn from the training data and generate conditional linguistic rules (Nauck et al., 1997). The user can interpret the result and perhaps learn the interrelations between the various input parameters from the rules.

The Neural-fuzzy system used in this study is called NEFCLASS (Nauck et al., 1997). It was developed by researchers at the University of Magdeburg, Germany.

NEFCLASS can learn fuzzy rules from training patterns, perform classification, and generate linguistic rules. NEFCLASS is a three-layer fuzzy perceptron, which learns by using back-propagation. Similar to NNs, it refines the class boundaries by iteratively minimizing the error rate using the training data provided to the program. However, NEFCLASS initiates the network differently than NNs. The initial rule generation consists of the following steps:

1) Input training data and the correct output value.
2) Find an existing membership function such that the input pattern results in the correct target class.
3) If no membership function can give a satisfactory result, then create a membership function that will. Each input pattern usually consists of a vector of values. The membership function is formed by assigning a range to each of the input parameters: small, medium, or large. For example:

   If parameter 1 is small, and parameter 2 is large, and parameter 3 is medium, and …, then class z

   These ranges are created using triangular membership functions. The function consists of three values: a, b, and c. The center of the triangle is identified by b, while a and c determine the spread to the right and left, respectively. In addition, the triangular membership function is allowed to overlap. The membership varies between 0 (no match) and 1 (perfect match). The learning process will later refine the ranges to better fit the training data.

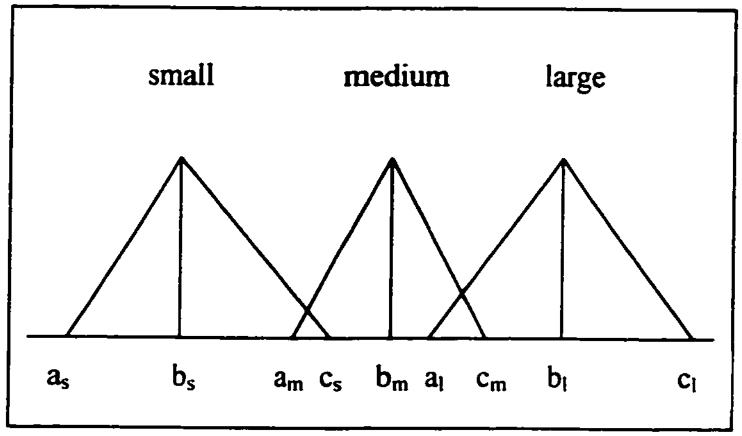


Figure 8. Triangular Membership

4) Repeat steps 1-3 until all the patterns are processed.

Once the initial rule base has been created, one can choose to refine the rule base by using a back-propagation learning algorithm. This process is similar to the previously mentioned algorithm used on NNs. The algorithm described below is applied to every training pattern cyclically until a given criterion, such as an accuracy rate, is met.

1) Select the next input pattern, and calculate the output vector o_c.
2) For each output unit, determine the error value $E_c = t_c - o_c$, where t_c is the correct output.
3) Now propagate the error back through the system

   a) Determine the error for each rule unit in the hidden layer:

   $$E_R = o_R \cdot (1 - o_R) \cdot \Sigma\, W(R, c) \cdot E_c$$

   where o_R is the actual rule value and $\Sigma W(R, c)$ is the sum of all the weights between the hidden layer and the output layer. This formula is almost identical to the one used in NNs.

b) Find the input unit x' such that

$$W(x', R)(o_{x'}) = \min\{W(x, R)(o_x)\}$$

where min stands for "the minimum of," W(x, R) is each weight between the input layer and the hidden layer, and (o_x) is the output value. This step is unique to NEFCLASS.

c) Determine the modification for the parameters a, b, c for the fuzzy set W(x', R).

$$\Delta b = \sigma \cdot E_R \cdot (c\text{-}a) \cdot \mathrm{sgn}(o_x - b),$$

$$\Delta a = -\sigma \cdot E_R \cdot (c\text{-}a) + \Delta b,$$

$$\Delta c = \sigma \cdot E_R \cdot (c\text{-}a) + \Delta b,$$

where σ is the user specified learning rate and $\sigma > 0$, and sgn(x) is the sign (+/–) of x.

4) If an epoch has been completed, and the stopping criterion is met, the calculation ends; otherwise repeat steps 1 through 3.

For any given value x, its membership in a certain group can be established with the following functions;

$(x - a) / (b - a)$	if $x \in [a, b)$,
$(c - x) / (c - b)$	if $x \in [b, c]$,
0	Otherwise

A pattern is interpreted by the system using this fuzzy rule set, and the system assigns membership degree to the various classes. For example, Table 2 is a small section of the network generated by the experiment. NEFCLASS uses a maximizer interpretation that maps the pattern to the class that has the highest association (Nauck et al., 1997).

Table 2. An Example of the NEFCLASS Output

```
% These are the parameters of the (triangular) fuzzy sets:
% <a>  <b>  <c>  <LeftShouldered> <RightShouldered> [name]
FUZZY
3
0.000000     99.000000
0.000000     16.500000     49.500000     1 0  small
16.500000    49.500000     82.500000     0 0  medium
49.500000    82.500000     99.000000     0 1  large
```

The source code for NEFCLASS is available at the University of Magdeburg, School of Computer Science, Department of Knowledge and Language Engineering's website: http://fuzzy.cs.uni-magdeburg.de.

Experimental Results and Discussion

The study region covers mainly Utah and adjacent areas in Colorado, Arizona, and Nevada. The vast Great Basin Desert lies within the boundary of this study site. The harsh desert environment supports unique ecosystems with desert plant and animal life. To survive the harshness of the desert climate, plants have evolved various adaptations including water-swollen stems; furry, gray leaves; thorns; or no leaves at all (Bowers, 1993). These characteristics make it difficult to distinguish desert land cover classes such as shrub, natural grass, and bare rock/sand using satellite imagery. A great amount of time is devoted to post-classification editing to correct the misclassification among the three classes.

Data Description

The spectral data used in this study includes the leaves-on and leaves-off Landsat Thematic Mapper (TM) images collected between 1991 and 1993. Due to the large size of the study area and the effectiveness of the data for analysis, only TM bands 3 (0.63 – 0.69 micrometers), 4 (0.76 – 0.90 micrometers), 5 (1.55 – 1.75 micrometers), and 7 (2.08 – 2.35 micrometers) were used. The USGS EROS Data Center in Sioux Falls, SD performed the pre-classification processing of the source data, which included destriping, terrain correction, geo-registration, and scene mosaicking. To mosaic, a base scene was selected, the histogram of the base scene was extracted, and other scenes were updated to approximate spectral properties of the base scene (Vogelmann et al., 1988). The mosaicked image was clustered into 100 spectrally distinct classes using the CLUSTER algorithm developed at the Los Alamos National Laboratory (Kelly and White, 1993; Benjamin et al., 1996). The EDC also supplied a Normalized Vegetation Index (NDVI) based on the TM image for each season, digital elevation, and slope data. The complete data set is listed in Table 3.

Table 3. Data Layers

	Leaf-on Images		Leaf-off Images		Others
1	100-cluster image	7	100-cluster image	13	Elevation
2	Landsat TM Band 3	8	Landsat TM Band 3	14	Slope
3	Landsat TM Band 4	9	Landsat TM Band 4		
4	Landsat TM Band 5	10	Landsat TM Band 5		
5	Landsat TM Band 7	11	Landsat TM Band 7		
6	NDVI	12	NDVI		

Data Classification

Four sites were selected from the study area in order to evaluate the effectiveness of the classification methods. Figure 9 shows the four sites in Landsat TM image in band combination 5, 4, 3. These sites were selected because they represent the characteristic desert landscape. The land-cover/land-use classes that appeared in these four sites include water, urban, bare soil/rock, evergreen forest, shrub, nature grassland, and hay/pasture. Four aerial photographs for the corresponding sites were used as reference. Training data were selected for each class based on ground truth and aerial photographs. Great care was taken to ensure that only homogeneous pixels were selected as training data.

The MLC and the accuracy assessment of all the classifications were carried out using ERDAS Imagine software. The binary image data were exported to the ASCII text format using ERDAS Imagine software, because NEFCLASS was created to analyze databases in text file format. The text data were then trained and tested in NEFCLASS. The training time varies for both MLC and NEFCLASS depending on the number of epochs pre-specified by the user. In this experiment, MLC training terminated if 95% of the sample data remains in the same class or after 6 epochs, whichever criteria was satisfied first. NEFCLASS was trained at 1, 5, 15, and 25 epochs for assessment of the classification performance. In general, more training cycles will generate rule sets that will yield better classifications for the given training data, and ideally will also classify the rest of the data better. However, over training runs the risk of losing generality of the rule set. Therefore the terminating criteria, percent of samples unchanged, would never be set to 100%.

Accuracy Assessment

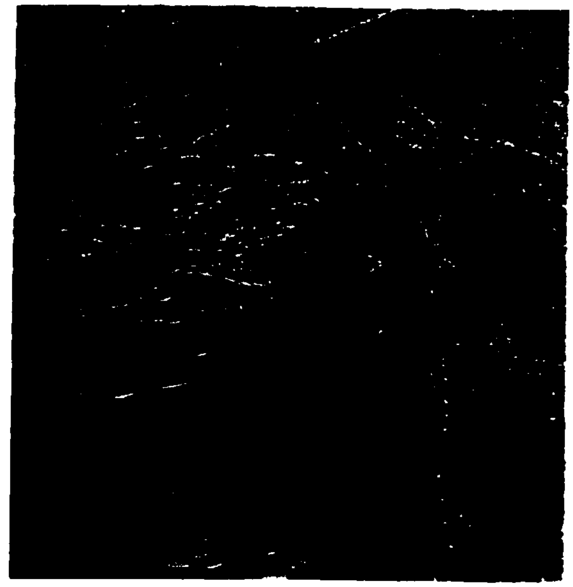
The accuracy assessment used a random stratified method. The number of samples needed for accuracy assessment is debatable. Fitzpatrick-Lins (1981) suggested using the binomial probability theory to determine the sample size.

$N=Z^2(p)(q) / E^2$,

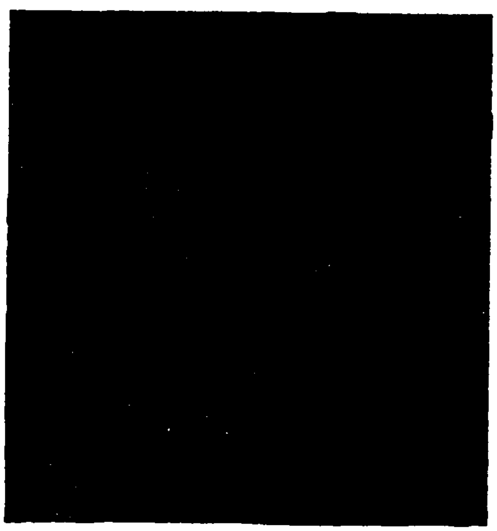
Where Z is the standard score that corresponds to the confidence level, p is expected accuracy, q is $(1 - p)$, and E is the allowable error.

For example, for an accuracy of 85% and an acceptable error of 5%, a minimum of 204 samples should be selected. In addition, Congalton (1991) suggested that, as a practical rule, using a minimum of 50 samples for each class in the error matrix. In this study, 50 samples were selected for the class bare, forest, shrub, and grass. Twenty-five samples were selected for the class water, urban, and hay/pasture, because they are much smaller classes. In total, 275 points are used for the accuracy assessment.

Site 21



Site 27



Site 47

Site 52

Figure 9. The Landsat TM Images of the Four Sites in Band 5 (Red), Band 4 (Green), and Band 3 (Blue).

Figure 10 shows the classification result using the MLC algorithm. The over estimation of the hay/pasture class can be seen in Sites 21, 27, and 52. An over estimation of bare rock/sand class in Site 27 and some misclassification of the urban class in Site 47 can also be observed.

Table 4: Maximum Likelihood Classification Result Error Matrix

	Water	Urban	Bare rock/ sand	Ever-green forest	Shrub-land	Grass-land	Hay/ pasture	Classified Total
Water	25							25
Urban	3	22			2			27
Bare rock/sand			9	1	26	14		50
Evergreen forest				48	6	1	1	56
Shrubland			25	1	21	5		52
Grassland			6		14	9		29
Hay/pasture					15	14	7	36
Reference Total	28	22	40	50	84	43	8	275

The overall accuracy for the MLC is 50.91% and the Kappa accuracy is 0.4169.

Table 5: Maximum Likelihood Classification Result Accuracy Total

	Reference Total	Classified Total	Number Correct	Producer's Accuracy	User's Accuracy
Water	28	25	25	89.29%	100.00%
Urban	22	27	22	100.00%	81.48%
Bare rock/sand	40	50	9	22.50%	18.00%
Evergreen forest	50	56	48	96.00%	85.71%
Shrubland	84	52	21	25.00%	40.38%
Grassland	43	29	9	20.93%	31.03%
Hay/pasture	8	36	6	75.00%	16.67%

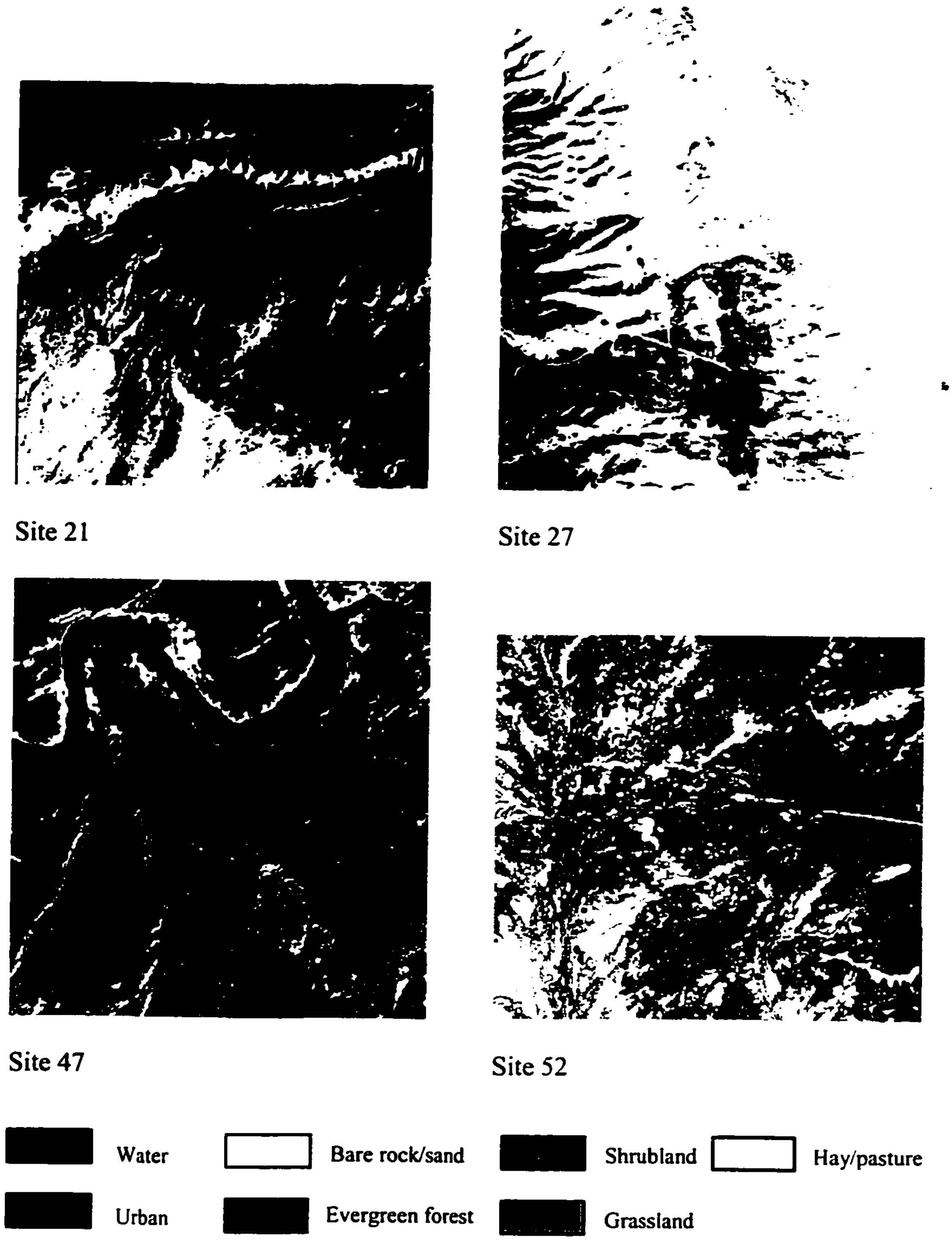


Figure 10. Maximum Likelihood Classification Results

Figure 11 shows the classification result using NEFCLASS. Some of the shrub class pixels are mistaken for urban in all four scenes. For example, a large part of the power plant at the bottom of Site 47 is misclassified. However, the highway is correctly classified in Site 52.

Table 6: NEFCLASS Classification Result Error Matrix

	Water	Urban	Bare rock/ sand	Ever-green forest	Shrub-land	Grass-land	Hay/ pasture	Classified Total
Water	25							25
Urban		21		1	1	2		25
Bare rock/sand			20		14	1		35
Evergreen forest				44		1		45
Shrubland			17	5	28	16		66
Grassland			3		28	17		48
Hay/pasture							25	25
Reference Total	25	21	40	50	71	37	25	275

The overall accuracy for the NEFCLASS is 65.45%, and the Kappa accuracy is 0.5872.

Table 7: NEFCLASS Classification Result Accuracy Total

	Reference Total	Classified Total	Number Correct	Producer's Accuracy	User's Accuracy
Water	25	25	25	100.00%	100.00%
Urban	21	25	21	100.00%	84.00%
Bare rock/sand	40	35	20	50.00%	57.14%
Evergreen forest	50	51	44	88.00%	86.27%
Shrubland	71	66	28	39.44%	42.42%
Grassland	37	48	17	45.95%	35.42%
Hay/pasture	31	25	25	80.65%	100.00%

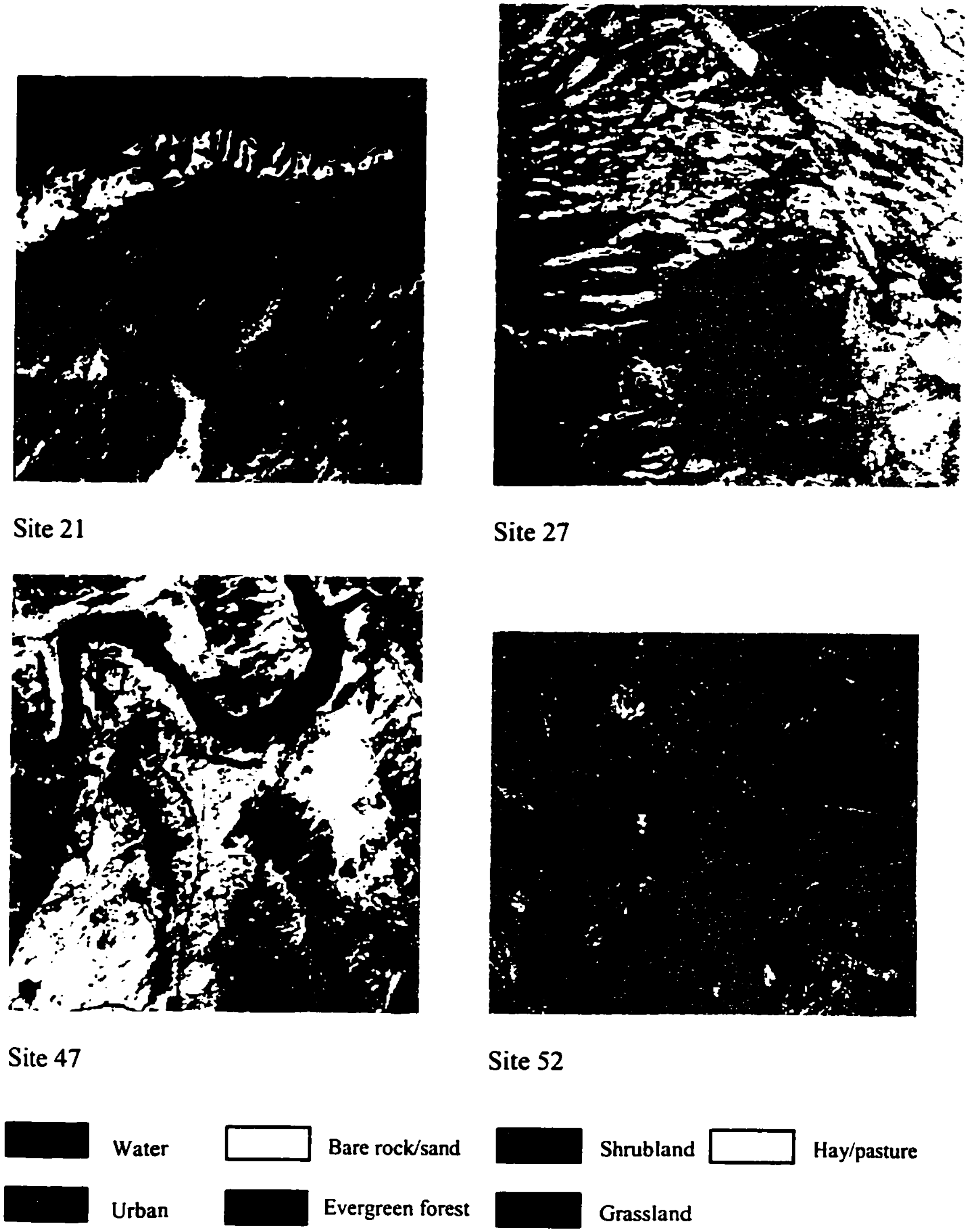


Figure 11. Neuro-fuzzy System Classification Results

The four sets of rules generated by training NEFCLASS with 1, 5, 15, and 25 epochs produced classification results of similar accuracy, varying from 88.65% to 91.17%. However the classification accuracy on the test data declined slightly. The NEFCLASS rule base stabilized very quickly, probably within the first 1 to 5 epochs. Since NEFCLASS is more tolerant, it can be used on multiple images or multi-temporal images (Paola and Schowengerdt, 1995a, p. 995). Therefore, potentially this classification method could save training time when a large data set is involved. A sample of the classification rules generated by NEFCLASS is in the Appendix.

Table 8: Accuracy Summary of the NEFCLASS Classifications

	Training Data Accuracy	Testing Data Accuracy	Kappa Coefficient
1 epoch	88.65%	67.25%	0.6106
5 epoch	90.47%	64.36%	0.5761
15 epoch	91.06%	66.16%	0.5975
25 epoch	91.17%	65.45%	0.5872

In summary, both classification methods gave reasonable accuracy in water, urban, and forest classes. NEFCLASS also classified hay/pasture class relatively well, however this class was grossly overestimated using MLC. This shortcoming could be the result of the equal probability assumption made by the MLC algorithm. Neither method classified bare rock/sand, shrub, and grass well. The reason could be that these three classes often co-exist in the desert landscape. Mixed pixels are very difficult to distinguish and separate. Adding a mixed shrub/grass class might represent the ground condition more accurately.

Discussion

Since the NNs are initialized with different weights and biases each time, the learning process will produce a different set of final weights. In addition, varying learning rates and the occurrences of local minima can also affect the outcome. Many researchers see this inconsistency in training as a shortcoming. Although NEFCLASS also exhibits this inconsistency in training, the difference in the end result is insignificant, as shown in Table 8. In addition, NEFCLASS uses numerous linguistic rules instead of using just one set of weights as in NNs, which avoids problems such as local minima. The learning rates in NEFCLASS could influence the shape of the membership functions, however the maximizer interpretation on class membership could minimize some of that effect unless the change is significant.

The MLC and NEFCLASS algorithms are similar in some respects. Both MLC and NNs, NEFCLASS included, form class borders in an arbitrary non-linear fashion (Ji, 2000). Both algorithms estimate the likelihood of the class membership. MLC compares the input values with the representative values of the possible classes, and assigns the candidate to the most likely class. It is very similar to the maximizer approach of NEFCLASS. NEFCLASS' approach to classification is analogous to the parallelepiped classification algorithm, a non-parametric method, used in conjunction with MLC. NEFCLASS' ability to classify data without considering the data distribution, yet still being able to form a complex class border, is certainly an advantage over MLC. This ability is demonstrated in the better classification of the hay/pasture class in this study.

The MLC classifier, being a parametric method, used mean and covariance statistics, but NNs used only the means (Paola and Schowengerdt, 1995b, p. 3053). The NNs, NEFCLASS included, obtain the mean by iteratively cycling through data during training, rather than using conventional statistical formulas. The NEFCLASS' fuzzy parameters do take into account the distribution of each class and they represent class mixing. MLC considers only the training data for the given class in calculating the parameters for that class. The Neural Networks perform a more mutually exclusive partitioning of the feature space by using all the training data to help delineate each class. Not only does the training data for a given class describe where that class exists in the feature space, but also the training data for all the other classes describe where that class does not exist (Paola and Schowengerdt, 1995b, p. 3035). These advantages of the NEFCLASS could have contributed to the overall better classification performance.

The NEFCLASS algorithm might possibly provide even more information on land classification than have been currently explored. Before the de-fuzzifying step, classification information existed in fuzzy terms. These fuzzy terms represent the degree of membership of a pixel to various classes. A study on this information may potentially yield information on class mixture for each pixel.

Conclusion

The goal of this study is to assess the ability of the NEFCLASS Neuro-fuzzy system to classify land-use/land-cover. The classification results obtained by using NEFCLASS were compared with classification results from MLC. Given the same training data, NEFCLASS performed better than MLC. The nonparametric approach and the learning ability of the algorithm are the main reasons for the superior performance in land-use/land-cover classification problem compared to the standard parametric method.

Bibliography

Benjamin, S., J.M. White, D. Argiro, and K. Lowell, 1996. Land Cover Mapping with Spectrum in Gap Analysis: A Landscape Approach to Biodiversity Planning, *Proceeding of the ASPRS/GAP Symposium*, Moscow, ID.

Bowers, Janice Emily, 1993. *Shrubs and Trees of the Southwest Deserts*, Southwest Parks and Monuments Association, Tucson, AZ.

Bruzzone, L., C. Conese, F. Maselli, and F. Roli, 1997. Multisource Classification of Complex Rural Areas by Statistical and Neural-Network Approaches, *Photogrammetric Engineering & Remote Sensing*, vol. 63, no. 5, May 1997, pp. 523-533.

Congalton, Russell G., 1991. A Review of Assessing the Accuracy of Classification of Remotely Sensed Data, *Remote Sensing Environment*, vol. 37, 1991, pp. 35-46

Fitzpatrick-Lins, K., 1981. Comparison of Sampling Procedures and Data Analysis for a Land-use and Land-cover Map, *Photogrammetric Engineering & Remote Sensing*, vol. 47, no. 3, March 1981, pp. 343-351.

Foody, Giles M., Mary B. McCulloch, and William B. Yates, 1995. Classification of Remotely Sensed Data by an Artificial Neural Network: Issues Related to Training Data Characteristics, *Photogrammetric Engineering & Remote Sensing*, vol. 61, no. 4, April 1995, pp. 391-401.

Hepner, George F., Thomas Logan, Niles Ritter, and Nevin Bryant, 1990. Artificial Neural Network Classification Using a Minimal Training Set: Comparison to Conventional Supervised Classification, *Photogrammetric Engineering & Remote Sensing*, vol. 56, no. 4, April 1990, pp. 469-473.

Jensen, John R., 1996. *Introductory Digital Image Processing*, Prentice Hall, Upper Saddle River, New Jersey.

Ji, C.Y., 2000. Land-Use Classification of Remotely Sensed Data Using Kohonen Self-Organizing Feature Map Neural Networks, *Photogrammetric Engineering & Remote Sensing*, vol.66, No. 12, December 2000, pp. 1451-1460.

Kartakopoulos, Stamatios V., 1996. *Understanding Neural Networks and Fuzzy Logic, Basic Concepts and Applications*, IEEE Press: Piscataway, NJ.

Kelly, P.M., and J.M. White, 1993. Preprocessing remotely sensed data for efficient analysis and classification, Application of Artificial Intelligence 1993: Knowledge-Based Systems in Aerospace and Industry, *Proceedings of SPIE*, 1993, pp 24-30.

Kulkarni, Arun D., 1994. *Artificial Neural Networks for Image Understanding*, Van Nostrand Reinhold, New York.

Nauck, Detlef, Frank Klawonn, and Rudolf Kruse, 1997. *Foundations of Neuro-fuzzy Systems*, John Wiley & Sons Ltd.: West Sussex, England.

Paola, Justin D. and Robert A. Schowengerdt, 1994. Comparison of Neural Networks to Standard Techniques for Image Classification and Correlation, *Proceedings of the International Geoscience and Remote Sensing Symposium* (IGRASS'94), Pasadena, CA, 8-12 August 1994, pp. 1404-1406.

Paola, Justin D. and Robert 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*, vol. 33, no. 4, July 1995, pp. 981-996.

Paola, Justin D. and Robert A. Schowengerdt, 1995b. A Review and Analysis of Backpropagation Neural Networks for Classification of Remotely Sensed Multispectral Imagery, *International Journal of Remote Sensing*, vol. 16, No. 16, pp. 3033-3058.

Paola, Justin D. and Robert A. Schowengerdt, 1997. The Effect of Neural-Network Structure on a Multispectral Land-Use/Land-Cover Classification, *Photogrammetric Engineering & Remote Sensing*, vol.63, No. 5, May 1997, pp. 535-544.

Schowengerdt, Robert A., 1997. *Remote Sensing, Models and Methods for Image Processing*, 2nd Edition, Academic Press: San Diego, CA

Solaiman, B. and M. C. Mouchot, 1994. A Comparative Study of Conventional and Neural Network Classification of Multispectral Data, *Proceedings of the International Geoscience and Remote Sensing Symposium* (IGRASS'94), Pasadena, CA, 8-12 August 1994, pp. 1413-1415.

Vogelman, J.E., T.Sohl, and S.M. Howard, 1988, Regional Characterization of Land Cover Using Multiple Sources of Data, *Photogrammetric Engineering & Remote Sensing*, vol. 54, No. 1, January 1988, pp. 45-57.

Appendix

The following is an example of rules generated by NEFCLASS using the training data selected for this study.

[Rules]
#Rule 1:
IF on3 IS medium AND on4 IS medium AND
on5 IS medium AND on7 IS medium AND
onclust IS medium AND onndvi IS medium AND
off3 IS large AND off4 IS medium AND
off5 IS medium AND off7 IS medium AND
offclust IS medium AND offndvi IS medium AND
dem IS medium AND slop IS small
THEN 71

#Rule 2:
IF on3 IS large AND on4 IS large AND
on5 IS large AND on7 IS medium AND
onclust IS medium AND onndvi IS medium AND
off3 IS large AND off4 IS large AND
off5 IS medium AND off7 IS large AND
offclust IS medium AND offndvi IS medium AND
dem IS medium AND slop IS small
THEN 51

#Rule 3:
IF on3 IS medium AND on4 IS medium AND
on5 IS small AND on7 IS small AND
onclust IS medium AND onndvi IS medium AND
off3 IS medium AND off4 IS medium AND
off5 IS medium AND off7 IS medium AND
offclust IS medium AND offndvi IS medium AND
dem IS medium AND slop IS small
THEN 51

#Rule 4:
IF on3 IS small AND on4 IS medium AND
on5 IS small AND on7 IS small AND
onclust IS small AND onndvi IS medium AND
off3 IS small AND off4 IS medium AND
off5 IS small AND off7 IS medium AND
offclust IS medium AND offndvi IS medium AND

```
    dem IS medium AND slop IS small
THEN 21

#Rule 5:
IF  on3 IS medium AND on4 IS medium AND
    on5 IS medium AND on7 IS medium AND
    onclust IS medium AND onndvi IS medium AND
    off3 IS medium AND off4 IS medium AND
    off5 IS medium AND off7 IS medium AND
    offclust IS medium AND offndvi IS medium AND
    dem IS medium AND slop IS small
THEN 51

#Rule 6:
IF  on3 IS medium AND on4 IS medium AND
    on5 IS medium AND on7 IS medium AND
    onclust IS medium AND onndvi IS medium AND
    off3 IS small AND off4 IS medium AND
    off5 IS medium AND off7 IS medium AND
    offclust IS medium AND offndvi IS medium AND
    dem IS medium AND slop IS small
THEN 51

#Rule 7:
IF  on3 IS small AND on4 IS medium AND
    on5 IS small AND on7 IS small AND
    onclust IS small AND onndvi IS small AND
    off3 IS small AND off4 IS medium AND
    off5 IS small AND off7 IS medium AND
    offclust IS medium AND offndvi IS medium AND
    dem IS medium AND slop IS small
THEN 21

#Rule 8:
IF  on3 IS medium AND on4 IS medium AND
    on5 IS medium AND on7 IS small AND
    onclust IS medium AND onndvi IS medium AND
    off3 IS medium AND off4 IS medium AND
    off5 IS medium AND off7 IS medium AND
    offclust IS medium AND offndvi IS medium AND
    dem IS medium AND slop IS small
THEN 51

...
```