4-28-2006

CONSENSUAL AND HIERARCHICAL CLASSIFICATION OF REMOTELY SENSED MULTISPECTRAL IMAGES

Jaejoon Lee
Purdue University

Okan Ersoy
Purdue University

CONSENSUAL AND HIERARCHICAL CLASSIFICATION OF REMOTELY SENSED MULTISPECTRAL IMAGES

Jaejoon Lee, Okan Ersoy

TABLE OF CONTENTS

Page

LIST OF TABLES iv

LIST OF FIGURES v

ABSTRACT viii

1. INTRODUCTION 1

2. STATISTICAL METHODS FOR CLASSIFICATION 5

2.1 Maximum Likelihood Classification 6
2.2 Minimum Distance Classification 9
2.3 Spatial-Spectral Classification 9
2.4 Class Separability 12
2.4.1 Divergence 14
2.4.2 Jeffries-Matusita distance 17

3. NEURAL NETWORKS FOR CLASSIFICATION 19

3.1 Neural Networks 19
3.2 Competitive Learning 20
3.3 Self-Organizing Maps 22
3.4 Self-Organizing Global Rankings Algorithm 24
3.5 Hierarchical Classifier with Rejection Schemes 26

4. NONLINEAR IMAGE FILTERING 40

4.1 Median Filter 40
4.2 Morphological Filter 40
4.2.1 Morphological operators 41
4.2.2 Generalized morphological filter 43
4.3 Spatial Filter 44

Page

5. CONSENSUS CLASSIFIER 45

5.1 Neural Network Ensembles 46
5.1.1 Methods for making ensemble members 46
5.1.2 Problems of combining multiple classifiers 48
5.1.3 Combination by majority rule 49
5.1.4 Combination by Bayesian average 50
5.2 Two Cases of Combining Multiple Classifiers 51
5.3 Diversity Measures in Combining Multiple Classifiers 54

6. EXPERIMENTAL RESULTS 57

6.1 Combining Multiple Different Classifiers 57
6.1.1 Experiment with West Lafayette Image 57
6.1.2 Experiment with Tippecanoe County Image 64
6.1.3 Experiment with Colorado Data Set 68
6.2 Combining Multiple Classifiers Generated with the Same Classification Algorithms 69

7. CONCLUSIONS AND FUTURE WORK 73

REFERENCES 76

LIST OF TABLES

Table Page

3.1 Transformed divergence of input data for each stage 39

3.2 J-M distance of input data for each stage 39

6.1 Channel description of West Lafayette image 58

6.2 Number of trained and tested data in each SNN for hierarchical SOM with rejection scheme applied to West Lafayette image 60

6.3 Classification performance of hierarchical approach ..... applied to West Lafayette image..... 60

6.4 Class seperability of West Lafayette image 60

6.5 Experimental results with West Lafayette image 61

6.6 Channel description of Tippecanoe County image..... 64

6.7 Class seperability of Tippecanoe County image 65

6.8 Experimental results with Tippecanoe County image 66

LIST OF FIGURES

Figure Page

1.1 Data in spectral space and feature space 2

2.1 Two dimensional multispectral space showing the degree of separation possible in a single dimensional subspace 13

2.2 Separability showing dependency on means and variances 14

2.3 Definition of the likelihood ratio at a point 15

2.4 Problem of divergence 17

3.1 Training process in the neural network 20

3.2 Procedure of updating the winning neuron 21

3.3 Geometric interpretation of competitive learning 22

3.4 One dimensional self-organizing map 23

3.5 Block diagram of the training procedure of hierarchical classifier with rejection schemes 27

3.6 Training and test data with two classes 33

3.7 Clustered results and constructed rejection boundaries in the SNN 1 33

3.8 *RADPN* and *RAD* constructed on training data in SNN 1 34

3.9 Accepted training and test data in the SNN 1 35

3.10 Rejected training and test data in the SNN 1 36

3.11 Clustered results and constructed rejection boundaries in the SNN 2 36

3.12 Classified training and test data up to SNN 2 37

3.13 Rejected training and test data from the SNN 2 38

3.14 Rejection boundaries and classified training and test data up to SNN2 38

3.15 Rejected training and test data in the SNN 2 39

4.1 Examples of structuring elements 41

4.2 Morphological operations 43

4.3 Block diagram of generalized morphological filter 44

5.1 Block diagram of combining multiple different classifiers 51

5.2 Block diagram of combining multiple same classifiers 53

6.1 Training and testing fields in the West Lafayette image 58

6.2 Number of neighborhoods and learning rate for SOGR 59

6.3 Thematic maps generated by consensual classification with..... West Lafayette image 62

6.4 Thematic maps generated by maximum likelihood classification with..... West Lafayette image 62

6.5 Thematic maps generated by ECHO with ..... West Lafayette image 62

6.6 Thematic maps generated by hierarchical competitive learning with ..... West Lafayette image 63

6.7 Thematic maps generated by hierarchical SOM with..... West Lafayette image 63

6.8 Thematic maps generated by hierarchical SOGR with..... West Lafayette image 63

6.9 Training and testing fields in the Tippecanoe County image 64

6.10 Thematic maps generated by consensual classification with Tippecanoe County image 66

6.11 Thematic maps generated by maximum likelihood classification with Tippecanoe County image 67

6.12 Thematic maps generated by ECHO with Tippecanoe County image 67

6.13 Thematic maps generated by hierarchical competitive learning with Tippecanoe County image 67

6.14 Thematic maps generated by hierarchical SOM with Tippecanoe County image 68

6.15 Thematic maps generated by hierarchical SOGR with Tippecanoe County image 68

6.16 Test results with hierarchical competitive learning 70

6.17 Test results with hierarchical SOM 71

6.18 Test results with hierarchical SOGR 71

ABSTRACT

Classification of remotely sensed multispectral images involves assigning a class to each pixel which has similar characteristics with known land cover. This is the important step in remote sensing to extract information about the Earth's surface. Statistical methods and computational intelligence algorithms such as neural networks are commonly used for classification. However, no single classifier can be good for all kinds of multispectral images. To obtain consistent and improved results, consensual and hierarchical approaches are applied. The proposed method consists of nonlinear image filtering, different multiple classifiers which use statistical methods or hierarchical neural networks with rejection schemes, and a combining scheme for integrating the results of multiple classifiers by consensus rule. Nonlinear image filtering is used to reduce variance of homogeneous region and improve spectral separability.

Most errors in classification occur with the data which are close to boundaries between classes. To handle these data more effectively, hierarchical structure is applied in classification using neural networks. By successive classifiers which are tuned to reduce remaining error, classification performance increases. This structure includes detection schemes to decide whether successive classifiers are utilized for each input. Rules are developed to determine automatically how many successive classifiers are needed. To obtain more reliable classification result for a given input pattern, multiple classification results for the same input pattern are combined by a consensus rule. Optimal weights for combining multiple classification results are computed in the sense of least squares based on the trained results of single classifiers to be combined. These are used to derive a consensus of multiple classification results.

If the classifier is based on neural networks, a classifier with a single algorithm can generate multiple different results by preprocessing input data and varying learning parameters. Since the same learning algorithm can be trained in different ways by preprocessing of the input pattern and varying learning parameters, generated classification results are different from each other with diverse errors as in classification with multiple different types of classifiers. By combining these classification results, classification performance increases. Experimental results with the proposed methods are discussed.

1. INTRODUCTION

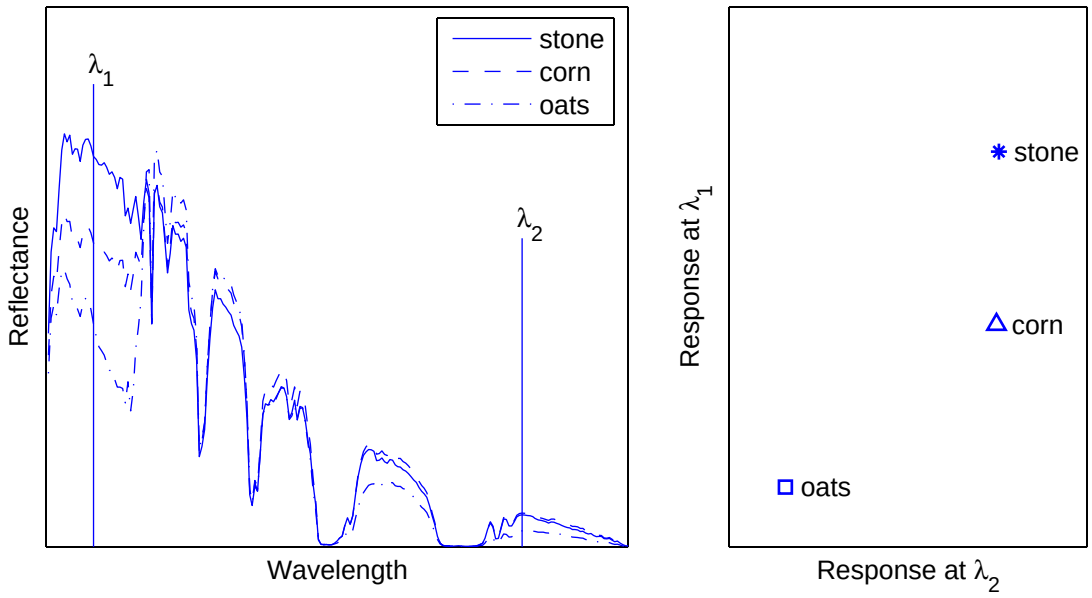
Remote sensing is the technology of acquiring information about the Earth's surface without actually being in contact with it [1]. This includes sensing and recording of reflected or emitted energy and processing and analyzing that information. Electro-magnetic radiation which is reflected or emitted from an object is often the source of remote sensing data. Remote sensors such as cameras or multispectral scanners are used to detect this electro-magnetic radiation. Since each object has unique characteristics of reflection or emission, the characteristics of an object can be determined using reflected or emitted electro-magnetic radiation from the object. Remote sensing is a process to identify the object through the uniqueness of the reflection or emission. Sensors measure the objects with a limited number of wavelength bands with or channels. Advances in sensor systems make it possible to improve multispectral remote sensing systems [2]. More accurate discriminations are possible by increased dimensionality. Each object has its own characteristics of reflection or emission. Using these characteristics, remote sensing data are analyzed and understood automatically with the computer. The results are used in agriculture, land use, forestry, geology, environment etc.

Image-based systems are often used in remote sensing. If remote sensing is done at a very high spatial resolution so that users could see details of interest to them in the scene, identifying objects on the Earth’s surface could be immediately performed. However, spatial resolution is a very expensive parameter in space data collection process. Higher spatial resolution leads not only to larger quantities of data for a given area as resolution is increased, but to larger sensor systems, and increased precision requirements on remote sensing sensors. To identify a vegetation species as corn, for example, one would require a spatial resolution in the centimeters range so that the shape of a corn leaf could be identified. In addition, higher resolution does not automatically increase the classification accuracy of multispectral images. This is what led to the concept of using spectral measurements of a pixel to identify the ground cover which the pixel represents.

Spectrally based method would function best for the same problem with spatial resolutions of order of a few tens of meters. So, it saves cost for data collection and also reduces the volume of data per unit area [17]. A major motivation for using spectral variations as the primary source to derive information is to avoid the need for very high spatial resolution.

Multispectral images are composite images which are taken at different wavelengths and combined together as a set of images of the same scene at different wavelengths. Each pixel of the multispectral image is a vector with N elements if the image consists of N bands. Figure 1.1 (a) represents the spectral response of a specific land covers sampled at 220 wavelengths. They consist of stone, corn, and oats. For any land cover, its spectral response is a complex mixture of finer constituents that reflect electromagnetic energy in different ways. The details of that mixture are its unique characteristic of the land cover. Thus, a way to characterize such mixture spectral responses as fully as possible is needed.

The spectral response of any land cover tends to vary in a characteristic way due to the relationships between the size and mixture of leaves, and so on. This variation is not easily discernable from the graph of reflectance, which is spectral response vs. wavelength for the class as seen in Figure 1.1 (a). The feature space solves this problem.



(a) Spectral space (b) Feature space

Figure 1.1 Data in spectral space and feature space.

In Figure 1.1 (a), spectral responses are sampled at two wavelengths, λ_1 and λ_2. Figure 1.1 (b) shows the two dimensional space representing these spectral responses sampled at two wavelengths. In Figure 1.1 (b), if only the measurement at λ_2 is available, stone and corn can not be distinguished since their spectral responses at λ_2 are the same. However, their spectral responses at λ_1 are different. Thus, corn and stone can be distinguished if the measurements at these wavelengths exist. If more dimensional spaces are used, more information presented in the spectral space can be used and the performance of pattern classification can be improved.

The remotely sensed data including multispectral images and other geographical information represent various features or land cover classes of interest identifying objects on the surface of the Earth. Classification is the process of categorizing similar objects and labeling their data with a single value. Then, one class represents a unique set of similar objects on the Earth's surface. There are many classification methods of remote sensing data [41], [42], [44], [68], [80]. Statistical methods and computational intelligence algorithms such as neural networks are commonly applied for classification. Benediktsson and Ersoy experimentally compared the two approaches, neural networks and statistical methods, in classification [6]. To improve classification performance, several image data from several different sources can also be used [43] [67].

However, no single method can be good for all kinds of multispectral images. To obtain consistent and improved results, consensual and hierarchical classification is used in this thesis. Consensus classification involves combining the results of several classifiers to obtain a better result than that of any single classifier. Compared with a single classifier, combined multiple classifiers have shown to perform better in classification performance since different classifiers could potentially offer complementary information in classification even when the result of each single classifier is not good [25], [37], [70], [71]. Various combining rules have been proposed in the literature [72], [73], [74], [75]. Consensus theory involves procedures for combining single probability distributions to summarize estimates from multiple data sources with the assumption that the experts make decisions based on Bayesian decision theory [3]. Consensus theory is based on the hypothesis that a group decision with multiple data sources is better in terms of mean square error than a decision from a single data source. Final classification result is decided by a consensus rule, such as combining each data source in a weighted sum with weights which are based on the individual performance of each classifier.

Another approach for obtaining higher classification accuracy is adopting hierarchical structure in neural networks. In many pattern recognition applications, classification accuracy is often lowered due to patterns that are likely to be wrongly classified. Data which are close to boundaries between classes usually cause errors in classification. To classify theses data more effectively, hierarchical structure is applied in classification using neural networks. The hierarchical approach means classification is done sequentially by multistage neural networks. When the input vectors which are difficult to classify are detected, their classification is done in the succeeding stage neural networks. In the current stage, input vectors having higher class separability are classified. This is effective to reduce misclassification and improve classification performance since successive classifiers are tuned to reduce remaining errors.

The thesis is organized into seven chapters. In Chapters 2, statistical methods for classification are discussed. In Chapter 3, neural networks and their hierarchical structure for classification are explained. In Chapter 4, nonlinear image filtering applied to input multispectral image is reviewed. In Chapter 5, combining multiple classification results generated by different types of classifiers and how to generate multiple different classification results with a single classifier by preprocessing of input vectors are discussed. Experimental results with remotely sensed multispectral images are given in Chapter 6. Conclusions and future work are discussed in Chapter 7.

2. STATISTICAL METHODS FOR CLASSIFICATION

Classification is a major subject in pattern recognition and involves classifying input data into a number of categories [51]. For N channel multispectral images, patterns are the pixel vectors that contain the sets of intensity values for the channels. A pixel vector can be arranged in column form as $\boldsymbol{x} = [x_1\ x_2\ \ldots\ x_N]^T$, where x_1 to x_N are the intensities in channel 1 to N, respectively.

Classification of multispectral images involves labeling the pixel vectors as belonging to particular classes using the spectral data available. Broadly, there are two kinds of classification procedure. One is unsupervised classification and the other one is supervised classification.

Unsupervised classification is assigning classes to pixels in an image without priori knowledge of the existence or names of those classes. Usually, this is performed by a clustering method which estimate the numbers and locations of the spectral classes. Unsupervised classification is useful for determining the spectral class composition of the data prior to detailed analysis by the methods of supervised classification.

Supervised classification is the most often used procedure for analysis of remotely sensed multispectral images. It assigns labels to the pixels representing particular ground cover types, or classes in an image set. A number of algorithms are available for this. The essential steps of supervised classification are as follows. First, representative pixels from each of the classes are chosen. These pixels are referred as training data. Training sets for each class can be selected using ground reference data. Usually the pixels in the training field for a given class are in a common region enclosed by a border. Secondly, parameters which will be used for the particular classification algorithms are estimated using training data. With this procedure, the classifier is constructed and generalization is performed. Finally, the images which are to be classified are inputted to the trained classifier, and final classification results are generated.

The objective of this thesis is classification of multispectral images by supervised classification. Some unsupervised classification techniques are included in supervised classification using neural networks in Chapter 3.

2.1 Maximum Likelihood Classification

Maximum likelihood classification is a popular method for supervised classification with remote sensing data. In this method, the mean, the variance and covariance of the class spectral response patterns are utilized when classifying an unknown pixel [1]. To do this, it is assumed that the distribution of the pixels forming the category of training data is Gaussian. Under this assumption, the distribution of a category response pattern can be completely described by the mean vector and the covariance matrix. Given these parameters, the statistical probability of a given pixel value being a member of a particular land cover class can be computed.

Let the spectral classes for an image be represented by ω_i, $i = 1, \ldots, C$, where C is the number of classes. To determine the class to which a pixel vector $\boldsymbol{x}$ belongs, the conditional probabilities $p(\omega_i|\boldsymbol{x})$, $i = 1,\ldots,C$, is estimated. The probability $p(\omega_i|\boldsymbol{x})$ gives the likelihood that the correct class is ω_i for a pixel at position $\boldsymbol{x}$. Classification rule is as follows:

$$\boldsymbol{x} \in \omega_i \text{ if } p(\omega_i \mid \boldsymbol{x}) > p(\omega_j \mid \boldsymbol{x}) \text{ for all } j \neq i \tag{2.1}$$

The pixel at $\boldsymbol{x}$ belongs to class ω_i if $p(\omega_i|\boldsymbol{x})$ is the largest. If sufficient training data is available for each ground cover type, it is possible to estimate a probability distribution for a cover type that describes the chance of finding a pixel from class ω_i at the position $\boldsymbol{x}$ [22]. A set of probabilities, which are $p(\boldsymbol{x}|\omega_i)$, can be computed that give the relative likelihoods that the pixel belongs to each available class for a pixel with $\boldsymbol{x}$ in multispectral space. The desired $p(\omega_i|\boldsymbol{x})$ in (2.1) and the $p(\boldsymbol{x}|\omega_i)$, which are estimated from training data, are related by Bayes' theorem [12]:

$$p(\omega_i \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid \omega_i)\, p(\omega_i)}{p(\boldsymbol{x})} \tag{2.2}$$

where $p(\omega_i)$ is the probability that class ω_i occurs in the image. If, for example, 10% of the pixels of an image happen to belong to spectral class ω_i then $p(\omega_i) = 0.10$. $p(\omega_i|\boldsymbol{x})$ are posterior probabilities. $p(\boldsymbol{x})$ in (2.2) is the probability of finding a pixel from any class at location $\boldsymbol{x}$ and can be obtained if $p(\omega_i)$ and $p(\boldsymbol{x}|\omega_i)$ are known as follows:

$$p(\boldsymbol{x}) = \sum_{i=1}^{C} p(\boldsymbol{x} \mid \omega_i) p(\omega_i) \tag{2.3}$$

By (2.2) the classification rule of (2.1) can be written as:

$$\boldsymbol{x} \in \omega_i \text{ if } p(\boldsymbol{x} \mid \omega_i) p(\omega_i) > p(\boldsymbol{x} \mid \omega_j) p(\omega_j) \text{ for all } j \neq i \tag{2.4}$$

where $p(\boldsymbol{x})$ has been removed as a common factor. The rule of (2.4) is more acceptable than that of (2.l) since $p(\boldsymbol{x}|\omega_i)$ and $p(\omega_i)$ can be estimated from training data. For mathematical convenience, logarithm can be applied to (2.4), and the discriminant function is defined as follows:

$$g_i(\boldsymbol{x}) = \ln\{ p(\boldsymbol{x}|\ \omega_i) p(\omega_i)\} = \quad \ln p(\boldsymbol{x}|\ \omega_i) + \ln p(\omega_i) \tag{2.5}$$

where ln is the natural logarithm. Then, (2.4) is restated as

$$\boldsymbol{x} \in \omega_i \text{ if } g_i(\boldsymbol{x}) > g_j(\boldsymbol{x}) \text{ for all } j \neq i \tag{2.6}$$

(2.6) is the decision rule used in maximum likelihood classification. $g_i(\boldsymbol{x})$ are referred to as discriminant functions. The class conditional probabilities $p(\boldsymbol{x}|\omega_i)$ in remote sensing are frequently assumed to belong to a normal probability distribution. In the case of a one dimensional spectral space, this is described by

$$p(x \mid \omega_i) = (2\pi)^{-1/2} \sigma_i^{-1} \exp\left\{ -\frac{(x - m_i)^2}{2\sigma_i^2} \right\} \tag{2.7}$$

where x is the single spectral variable, m_i is the mean value of x, σ_i is its standard deviation, and σ_i^2 is the variance of the distribution.

Usually, the spectral signatures of each class type are modeled to have multivariate normal distribution, and the parameters of such spectral signatures are estimated from training samples. Based on the spectral signatures, the spectral vector of a pixel is used to classify the pixel by using a classifier [13]. Assumption that the probability distributions for the classes are of the form of multivariate normal models leads to mathematical simplifications as follows. (2.7) can be extended for the multivariate distribution by some modification. Vector form $\boldsymbol{x}$ is used instead of x and the

univariate mean m_i of the data in class ω_i is replaced by its multivariate $\boldsymbol{m}_i$. The variance σ_i^2 is modified to take account of multidimensionality and the effect of correlation between spectral bands is included. For this, covariance matrix is defined as

$$\Sigma_i = \mathrm{E}\{(\boldsymbol{x} - \boldsymbol{m}_i)(\boldsymbol{x} - \boldsymbol{m}_i)^t\} \tag{2.8}$$

The covariance matrix is inverted and inserted into the numerator of the exponent. The complete form of the multivariate normal distribution for N spectral dimensions is defined as

$$p(\boldsymbol{x} \mid \omega_i) = (2\pi)^{-\mathrm{N}/2} \mid \Sigma_i \mid^{-1/2} \exp\left\{-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_i)^t \Sigma_i^{-1}(\boldsymbol{x} - \boldsymbol{m}_i)\right\} \tag{2.9}$$

From (2.9) $g_i(\boldsymbol{x})$ in(2.5) can be calculated. The resulting term $-N/2 \ln(2\pi)$ is common to all $g_i(\boldsymbol{x})$ and does not aid discrimination. Consequently it is ignored and the final form of the discriminant function for maximum likelihood classification, based upon the assumption of normal statistics, is given by

$$g_i(\boldsymbol{x}) = \ln p(\omega_i) - \frac{1}{2}\ln \mid \Sigma_i \mid - \frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_i)^t \Sigma_i^{-1}(\boldsymbol{x} - \boldsymbol{m}_i) \tag{2.10}$$

If equal prior probabilities is assumed, $\ln p(\omega_i)$ can be removed from (2.10) since it is then the same for all i. If 1/2 common factor is removed, the discriminant function is as follows:

$$g_i(\boldsymbol{x}) = -\ln \mid \Sigma_i \mid -(\boldsymbol{x} - \boldsymbol{m}_i)^{\mathrm{t}} \Sigma_i^{-1}(\boldsymbol{x} - \boldsymbol{m}_i) \tag{2.11}$$

This is the decision rule for maximum likelihood classification.

The major drawback of maximum likelihood classification is the large number of computations required to classify each pixel. This is particularly true when either a large number of spectral channels are involved or a large number of spectral classes must be differentiated. In such cases, the maximum likelihood classifier will be slower computationally.

2.2 Minimum Distance Classification

The performance of maximum likelihood classification depends on how well the estimation of the mean vector $\boldsymbol{m}$ and the covariance matrix Σ for each spectral class is done. Accurate estimation is possible when there are a lot of training pixels for each class. If there are insufficient training data, estimates of the elements of Σ are inaccurate and this leads to poor classification. When the number of training samples per class is limited, it can be more effective to resort to a classifier that depends only on the mean positions of the spectral classes rather than using covariance information. This is the minimum distance classifier. This classifies the data depending on the distance with each class mean. With this classifier, training data are used only to determine class means. The discriminant function for the minimum distance classifier is developed as follows. Suppose $\boldsymbol{m}_i$, $i = 1,..., C$ are the means of the C classes determined from training data, and $\boldsymbol{x}$ is the position of the pixel to be classified. Distance between the mean of each class and input vectors are calculated as follows:

$$\mathrm{d}(\boldsymbol{x}, \boldsymbol{m}_i) = \left\| \boldsymbol{x} - \boldsymbol{m}_i \right\| \qquad i = 1, \ldots, C \tag{2.12}$$

Euclidean distance is commonly used for distance measure. When the distances between $\boldsymbol{x}$ and all $\boldsymbol{m}_i$ are computed, classification is performed on the basis of

$$\boldsymbol{x} \in \omega_i \text{ if } \mathrm{d}(\boldsymbol{x}, \boldsymbol{m}_i) > \mathrm{d}(\boldsymbol{x}, \boldsymbol{m}_j) \text{ for all } j \neq i \tag{2.13}$$

Input vector $\boldsymbol{x}$ is classified to the class which has the minimum distance to the mean of the class.

2.3 Spatial-Spectral Classification

Remotely sensed multispectral images provide information based on the spectral, spatial, and temporal variations of the electromagnetic fields emanating from the scene. However, conventional statistical pixel-based classification techniques such as maximum likelihood classification perform class assignments based only on the spectral signatures of the pixel. Hence, the spatial information, which is potentially useful, is not taken into for classification. This leads to ignoring useful contextual information in neighboring pixels. While traditional per-pixel classifiers ignore this, spatial-spectral classifiers

specifically try to incorporate this additional information into the classification algorithm. All per-pixel classifiers try to assign every pixel to its spectrally closest class. However, spatial-spectral classifiers assign a whole set of pixels to its spectrally closest class. The higher the degree of spatial autocorrelation is, the better the results of spatial-spectral classifiers.

To use spatial information as well as spectral information, spatial-spectral classifications are used. These methods make use of the spatial context of a pixel in its classification. These methods are based on the assumption that the response and class of two spatially neighboring pixels are highly related. For example, if $x(i,j)$ and $x(i+1, j)$ are two neighboring pixels and if $x(i,j)$ belongs to class ω_1, then there is a higher probability that $x(i+1,j)$ belongs to the same class. Thus, the decision for a pixel $x(i,j)$ is to be taken based not only on $x(i,j)$ but also on all pixels which are neighbor of $x(i,j)$. Use of spatial information usually results in a reduction of misclassification. Several classification methods which exploit both the spatial information and the spectral information were developed [15], [23], [24], [49] [60]. Spatial image filtering and segmentation are common strategies for using spatial information [85], [86].

One type of spatial information in a remotely sensed image useful for classification is the information on homogeneous regions. Region based classification uses homogenous regions of images in the classification process. This kind of classification consists of comparing the distribution of sample, which is an object's pixels, to the training class distributions instead of comparing just a single pixel vector to the training class distributions. Basically, region based classification involves two parts, scene segmentation into objects and sample classification algorithm in which each object would be classified based on the statistical properties of each object's pixels. Here the pixels in homogeneous regions are assumed to belong to a common class.

Homogeneous regions are found by image segmentation techniques. Image segmentation refers to the decomposition of a scene into non-overlapping and meaningful components according to the characteristics such as texture [18]. A criterion in this process is that the union of adjacent regions is not homogeneous. By segmentation, an image is partitioned into disjoined regions corresponding to objects. This enables further classification to be performed based on the information provided by objects rather than individual pixels.

There are many image segmentation techniques such as edge-based methods, region growing, thresholding histogram, and split and merge schemes [26], [48], [61], [81], [82], [83]. Various segmentation methods for remote sensing images have been

proposed in the literature [62], [63], [64], [65], [66] [84]. Edge based segmentation is based on detection of edges. This method has drawback that closed boundaries are not guaranteed. To form closed boundaries, additional image processing algorithms are needed. Instead of edge based segmentation, region based segmentation exploits the internal homogeneity of the objects. These methods always make closed boundaries. There are conjunctive and disjunctive approaches in this method. Conjunctive algorithms cluster pixels starting with seed points which grow into regions until a certain threshold is reached. A region grows until no more pixels can be attributed to any of the segments, new seeds are placed and the process is repeated. This continues until the whole image is segmented. A disjunctive approach begins with a simple partition and subdivides it until each element satisfies a criterion of homogeneity.

The method of extraction and classification of homogeneous objects (ECHO) is a region based classification method which uses both spectral and spatial information [14], [50]. It uses the dependence between adjacent pixels. All pixels of a segment or region are assumed to belong to the same class. ECHO is used as one of the classifiers for consensus between multiple classifiers in this thesis. First, the image is segmented by conjunctive method, which begins with very fine partition and merges adjacent segments together to form homogeneous regions. For the scene segmentation, cells which have n×n pixels are defined. Then the homogeneity of the cells is tested by the following equations [17]:

$$\frac{\sqrt{sample\ \ variance}}{sample\ \ \ mean} > threshold \qquad \text{Reject} \qquad (2.14)$$

$$\text{Otherwise} \qquad\qquad\qquad\qquad \text{Accept}$$

In this way, homogeneities of all cells in an image are tested. In the second stage, statistically similar homogeneous cells that are adjacent are merged together. In this way, a region, referred to as an object, having common statistical characteristics continues to grow across and down until the merged region meets the heterogeneous cell with the different statistics, indicating a boundary in the scene.

The cell that fails the initial homogeneity test would be called singular and would be dealt with on pixel basis. When the scene segmentation is done, each segmented region is classified by maximum likelihood classifier, and pixels in the singular cells are classified on a pixel basis. Whereas the pixel is classified in per pixel basis depending on

its likelihood, the sample, which is the collection of pixels with similar characteristics, is classified by the maximum likelihood method in sample classification. By integration of spatial and spectral information, classification accuracy can be improved.

2.4 Class Separability

The ideal goal of classification is to assign correct labels to the input data. Classification performance depends on the class separability of input data as well as how classification system is designed well. Class separability implies the ease with which patterns can be correctly associated with their classes using statistical pattern classification. If the separability is higher, the classification becomes easier and more correct. In this thesis, nonlinear filtering and hierarchical approach are used. How these methods improve classification performance is investigated in terms of classification separability later.

Separability is usually used for feature selection. Since classification cost increases with the number of spectral bands associated with a pixel, if the image has a number of channels such as in hyperspectral images, removal of least effective features is needed to reduce the cost of classification. Features which are not helpful for discrimination, by contributing little to the separability of spectral classes, are to be discarded for saving processing time and cost. This is referred to as feature selection. By feature selection, the dimensionality of a data set is reduced. If the separability is not lowered even though some features are excluded, then the set of excluded features is considered not to contribute to discrimination in classification.

A two dimensional multispectral space with two spectral classes ω_1 and ω_2 is shown in Figure 2.1. Suppose that it is not known which feature offers the best prospects a priori, and we want to see whether the classes could be separated using only one feature, either x_1 or x_2. This is determined by a measure of separability. Consider the case that only x_1 subspace is used for classification. The spectral classes in the x_1 subspace are shown in Figure 2.1. There is an overlapped region between two classes as shown in this figure. If the distributions are well separated in the x_1 dimension, then the overlapped region will be small and it would be unlikely that a classifier would make little error in discriminating between two classes on the basis of that feature alone. On the other hand, for a large degree of overlapped region, substantial classification error would be expected.

Therefore, the usefulness of the x_1 feature subset can be measured in terms of the overlap of the distributions in that domain.

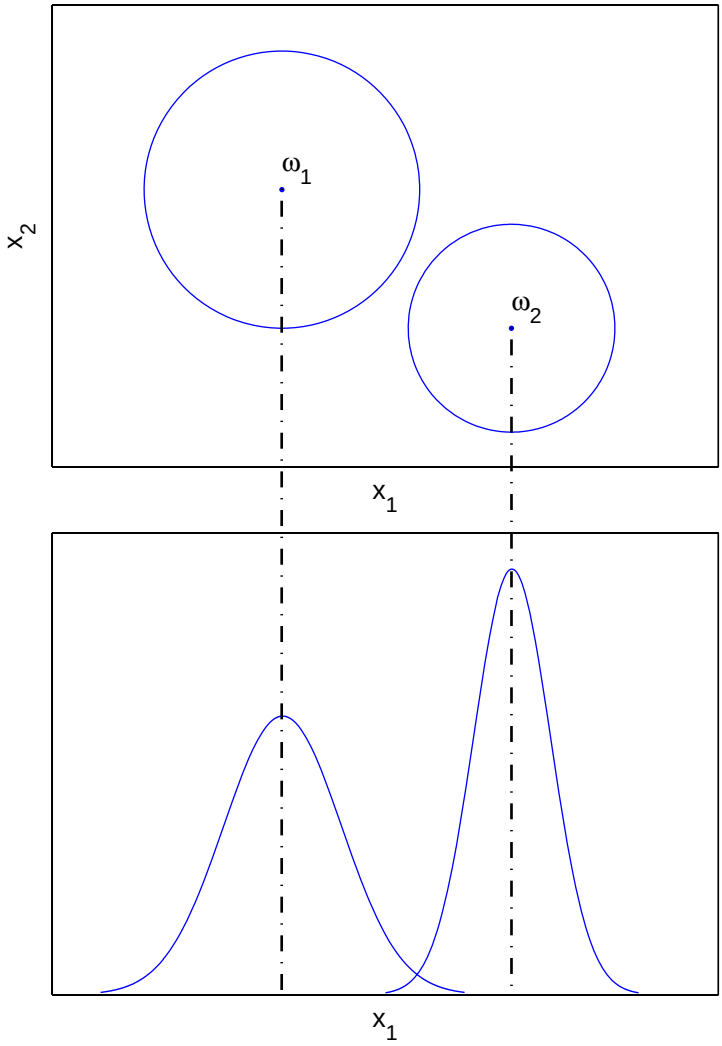


Figure 2.1 Two dimensional multispectral space showing the degree of separation possible in a single dimensional subspace.

Class separability is used for quantification of the separation between a pair of probability distributions as an indication of the degree of overlap. As shown in Figure 2.2, if the degree of separation is directly related to the distances between means, and inversely to the standard deviations of the distributions, a simple measure of separability can be written as

$$d_{norm} = \frac{|\mu_1 - \mu_2|}{\sigma_1 + \sigma_2} \tag{2.15}$$

However, if the means of two classes are equal even when the variances are not equal, degree of separation is zero. So this measure is not enough as a separability measure. Instead, several other measures are available to measure class separability more accyrately.

2.4.1 Divergence

Class separability depends on the concept of a measure of statistical distance between the probability densities characterizing the pattern classes [45]. Divergence is a measure of the separability of a pair of probability distributions that has its basis in their degree of overlap [22], [46]. In Figure 2.3, A is defined as $p(\boldsymbol{x}_0|\omega_i)$ - $p(\boldsymbol{x}_0|\omega_j)$. The larger the interval A is, the more likely it is that the classifier will classify $\boldsymbol{x}_0$ to the class ω_i. The size of a for any point in $\boldsymbol{x}$ is characterized by the value of the likelihood ratio at that point as in (2.16).

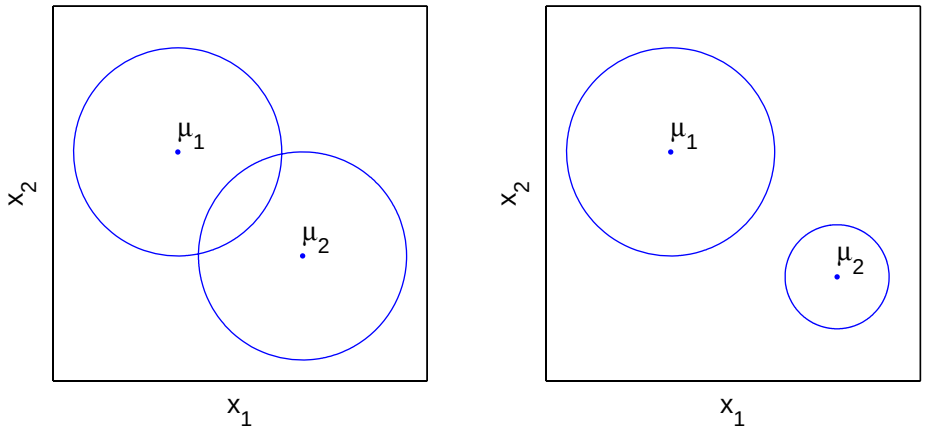


Figure 2.2 Separability showing dependency on means and variances.

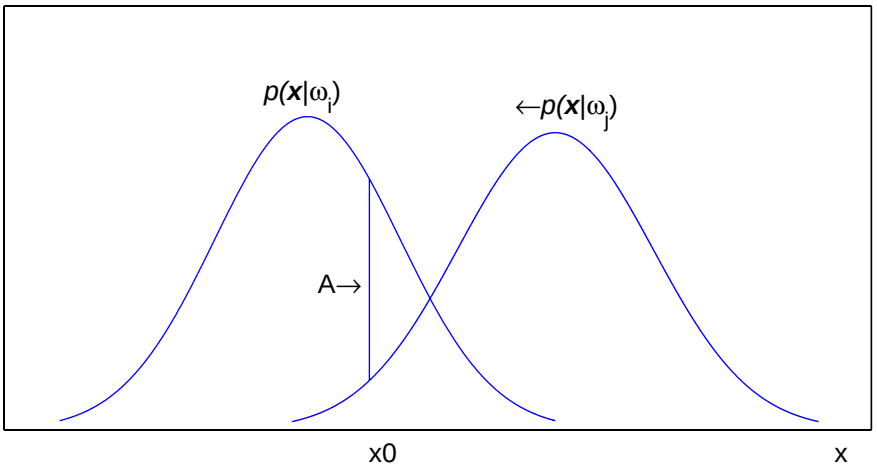


Figure 2.3 Definition of the likelihood ratio at a point.

$$L_{ij}(\boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid \omega_i)}{p(\boldsymbol{x} \mid \omega_j)} \tag{2.16}$$

where $p(\boldsymbol{x}|\omega_i)$ and $p(\boldsymbol{x}|\omega_j)$ are the values of the ith and jth spectral class probability distributions at the position $\boldsymbol{x}$. The larger A is, the larger the value of the likelihood ratio. These are shown in an overlap region in Figure 2.3. $L_{ij}(\boldsymbol{x})$ is a measure of instantaneous overlap. If two classes are very separable, then spectral classes $L_{ij}(\boldsymbol{x}) = 0$ or ∞ for all $\boldsymbol{x}$. The logarithm of the likelihood ratio is mathematically more convenient:

$$L'_{ij}(\boldsymbol{x}) = \ln L_{ij}(\boldsymbol{x}) = \ln p(\boldsymbol{x} \mid \omega_i) - \ln p(\boldsymbol{x} \mid \omega_j) \tag{2.17}$$

The divergence D_{ij}, of classes i and j is defined in terms of this logarithm of the likelihood ratio as:

$$D_{ij} = E[L'_{ij}(\boldsymbol{x}) \mid \omega_i] + E[L'_{ji}(\boldsymbol{x}) \mid \omega_j] \tag{2.18}$$

where $E[]$ denotes the expectation as follows:

$$E[L'_{ij}(\boldsymbol{x}) \mid \omega_i] = \int_{\boldsymbol{x}} L'_{ij}(\boldsymbol{x}) p(\boldsymbol{x} \mid \omega_i) \mathrm{d}\boldsymbol{x} \tag{2.19}$$

This is the average or expected value of the likelihood ratio with respect to all patterns in the ith spectral class. From (2.18) it can be seen that

$$D_{ij} = \int_{\boldsymbol{x}} [p(\boldsymbol{x} \mid \omega_i) - p(\boldsymbol{x} \mid \omega_j)] \ln \frac{p(\boldsymbol{x} \mid \omega_i)}{p(\boldsymbol{x} \mid \omega_j)} \mathrm{d}\boldsymbol{x} \tag{2.20}$$

from which a number of properties of divergence can be established. For example, it is always positive and also $D_{ij} = D_{ji}$, i.e., it is symmetric. Moreover, if $p(\boldsymbol{x} \mid \omega_i) = p(\boldsymbol{x} \mid \omega_j)$ for all $\boldsymbol{x}$, then $D_{ij} = D_{ji} = 0$. For statistically independent features (i.e., spectral components) $x_1, x_2, \ldots, x_N$, then

$$p(\boldsymbol{x} \mid \omega_i) = \prod_{n=1}^{N} p(x_{\boldsymbol{n}} \mid \omega_i) \tag{2.21}$$

which leads to

$$D_{ij}(\boldsymbol{x}) = \sum_{n=1}^{N} D_{ij}(x_{\boldsymbol{n}}) \tag{2.22}$$

Since divergence is never negative it follows that

$$D_{ij}(x_1, \ldots, x_{\mathrm{n}}, x_{\mathrm{n+1}}) > D_{ij}(x_1, \ldots, x_{\mathrm{n}}) \tag{2.23}$$

In other words, divergence never decreases as the number of features is increased. Since spectral classes in remote sensing image data are often modeled by multidimensional normal distributions, (2.18) is represented as in (2.24) when $p(\boldsymbol{x}|\omega_i)$ and $p(\boldsymbol{x}|\omega_j)$ are normal distributions with means and covariances of $\boldsymbol{m_i}, \Sigma_i$ and $\boldsymbol{m_j}, \Sigma_j$ respectively:

$$\begin{aligned} D_{ij} &= \frac{1}{2}\mathrm{tr}\{[\Sigma_i - \Sigma_j][\Sigma_j^{-1} - \Sigma_i^{-1}]\} + \frac{1}{2}\mathrm{tr}\{[\Sigma_i^{-1} + \Sigma_j^{-1}][\boldsymbol{m}_i - \boldsymbol{m}_j][\boldsymbol{m}_i - \boldsymbol{m}_j]^T\} \\ &= \mathrm{A+B} \end{aligned} \tag{2.24}$$

where tr{} is the trace of the subject matrix. Note that A involves only covariances whereas B is the square of a normalized distance between the means of the distributions. Although divergence only provides a measure of the distance between two class densities, its use is extended to the multiclass case by taking the average over all class pairs. If there are more than two spectral classes, all pairwise divergences need to be checked to compute average divergence. Average divergence in case of multiclass problem is defined as in (2.25):

$$D_{avg} = \frac{2}{M(M-1)} \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} D_{ij} \tag{2.25}$$

where M is the number of spectral classes.

2.4.2 Jeffries-Matusita distance

Even for the two class case, the relationship between divergence and classification accuracy is highly nonlinear. Divergence increases without bound as class separability increases, whereas probability of correct classification must saturate at 1 as shown Figure 2. 4. This behavior is quite misleading if divergence is to be used as an indication of how successfully patterns in the corresponding spectral classes could be mutually discriminated or classified. The Jefries-Matusita distance (J-M distance) does not suffer from this drawback. J-M distance provides a much more reliable criterion, presumably because as a function of class separability it behaves much more like probability of correct classification.

The J-M distance between a pair of probability distributions (spectral classes) is defined as

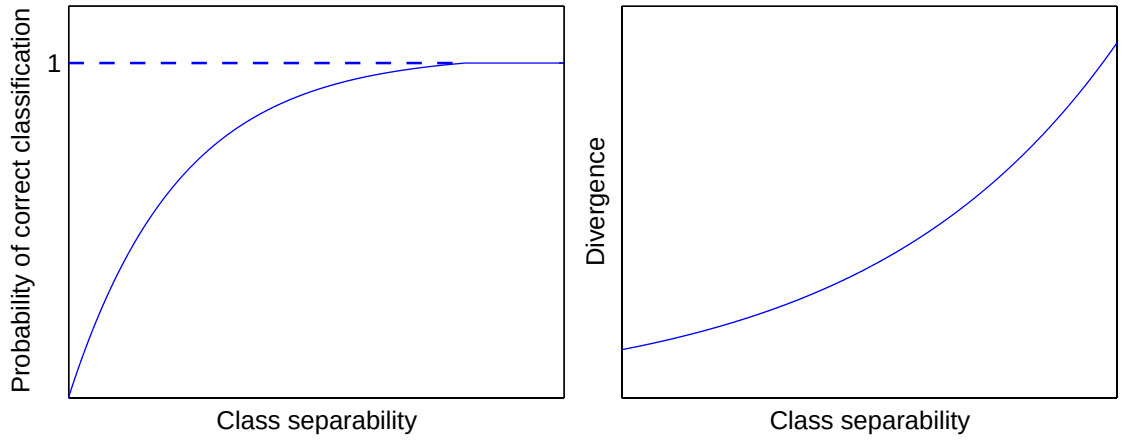


Figure 2.4 Problem of divergence.

$$J_{ij} = \int_{\boldsymbol{x}} \left\{ \sqrt{p(\boldsymbol{x} \mid \omega_i)} - \sqrt{p(\boldsymbol{x} \mid \omega_j)} \right\}^2 \mathrm{d}\boldsymbol{x} \tag{2.26}$$

which is seen to be a measure of the average distance between the two class density functions. For normally distributed classes this becomes

$$J_{ij} = 2(1 - e^{-B}) \tag{2.27}$$

where

$$B = \frac{1}{8}(\boldsymbol{m}_i - \boldsymbol{m}_j)^T \left\{ \frac{\Sigma_i + \Sigma_j}{2} \right\}^{-1} (\boldsymbol{m}_i - \boldsymbol{m}_j) + \frac{1}{2} \ln \left\{ \frac{\frac{1}{2}\left(\Sigma_i + \Sigma_j\right)}{\sqrt{\left|\Sigma_i\right|\left|\Sigma_j\right|}} \right\} \tag{2.28}$$

which is referred to as the Bhattacharyya distance [47]. In (2.28), since $0< e^{-B}<1$, J_{ij} ranges from 0 to 2. J-M distance of 2.0 between spectral classes would imply classification with 100% accuracy. The presence of the exponential factor in (2.27) gives an exponentially decreasing weight to increasing separations between spectral classes. If the J-M distance is plotted as a function of separability between classes, it shows a saturating behavior unlike the divergence. Owing to this saturating behavior, J-M distance does not suffer from the difficulty experienced with divergence. If J_{ij} is the J-M distance between classes i and j, then the average J-M distance is defined as:

$$J_{avg} = \frac{2}{M(M-1)} \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} J_{ij} \tag{2.29}$$

where M is the number of spectral classes.

Transformed divergence was proposed as in (2.30) to make divergence as effective as J-M distance and computationally effective and to compensate the problem as shown in Figure 2.4 [46].

$$D_{ij}^T = 2000(1 - \mathrm{e}^{-D_{ij}/8}) \tag{2.30}$$

Transformed divergence has a saturating behavior with increasing class separation, as in J-M distance because of its exponential character.

3. NEURAL NETWORKS FOR CLASSIFICATION

3.1 Neural Networks

Neural network is a mathematical or computational model for information processing. This is inspired by biological nervous systems, especially the human brain. It is composed of a lot of interconnected processing elements (neurons) which work to solve specific problems. Learning occurs in the neural networks by adjustment of the synaptic connections that exist between the neurons. Learning enables a neural network to know how to do tasks based on the given data for training. Most neural networks have some training rule whereby the weights of connections are adjusted on the basis of data. In other words, neural networks learn from examples and exhibit some capability for generalization beyond the training data. The objective of training may be to extract relevant information from the database in order to classify future input patterns [34]. A neural network is a massively parallel distributed processor that has a natural propensity for storing experimental knowledge and making it available for use [5].

A neural network acquires knowledge through a learning process. This knowledge is stored in the synaptic weights which are the interneuron connection strengths. Neural networks are usually computational nonlinear algorithms for numerical data processing. Neural networks are especially useful for classification and function mapping problems which are tolerant of some imprecision, and which have lots of training data available. Almost any mapping between vector spaces can be approximated to arbitrary precision by feedforward neural networks if there are enough data and enough computing resources. Figure 3.1 visualizes the training process in a neural network. The network function is determined largely by the connections between elements. Neural network can be trained to perform a particular function, which leads to a specific target output, by adjusting the values of the weights between the neurons. In Figure 3.1, the network is adjusted, based on a comparison of the output of the neural network and the target output, until the network output matches the target. This is supervised learning. A supervised learning scheme is implemented using a database which consists of samples from the set of

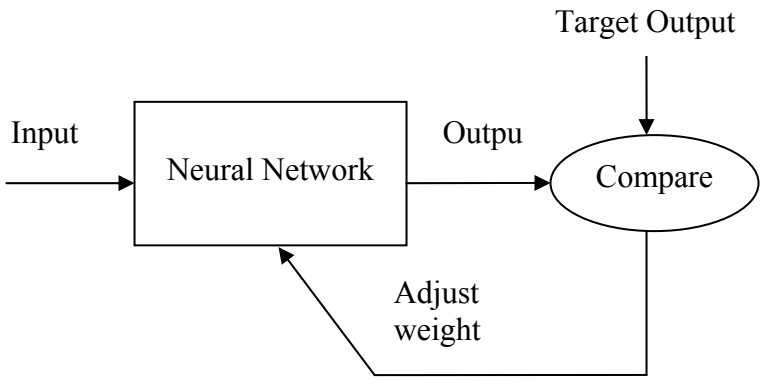


Figure 3.1 Training process in the neural network.

possible inputs together with the corresponding targets (classes), which are labeled samples. That is, neural networks learn by external teacher such that each output unit is told what its desired response to input signals ought to be.

Neural networks are used in various fields of applications including pattern recognition, identification, classification, and control systems. In this thesis, neural networks are used for classification of remote sensing data. This involves labeling the remote sensing data to the class with similar characteristics. Neural networks have an advantage over the statistical methods since they are distribution-free and no prior knowledge is needed about the statistical distributions of the classes in the data sources in order to apply these methods for classification [6].

3.2 Competitive Learning

Competitive learning is an unsupervised learning method. As the name implies, the neurons are in competition for input patterns [5]. During training, the neuron that provides the highest activation to a given input pattern is declared the winner and the winner moves closer to the input pattern, whereas other neurons are left unchanged. To find the winning neuron, the Euclidean distance between the input vectors and the weight vectors are usually used as activation function. The neuron c whose weight vector is the closest to $\boldsymbol{x}$ is declared the winning neuron. This is the best matching unit (BMU) of $\boldsymbol{x}$.

$$\|\boldsymbol{x} - \boldsymbol{w}_c\| = \min_i \{\|\boldsymbol{x} - \boldsymbol{w}_i\|\} \tag{3.1}$$

Moving of the winning neuron towards the input patterns is achieved by updating of the winning neuron by (3.2). After updating, the winning neuron becomes a little more like the current input vector. It is this feature that makes competitive learning highly suited to discover statistically salient features that may be used to classify a set of input patterns. The equations for competitive learning are as follows:

$$\begin{aligned} \boldsymbol{w}_i(k+1) &= \boldsymbol{w}_i(k) + C(k)\,[\boldsymbol{x} - \boldsymbol{w}_i(k)] && \text{if } i \text{ wins} \\ \boldsymbol{w}_i(k+1) &= \boldsymbol{w}_i(k) && \text{otherwise} \end{aligned} \tag{3.2}$$

where $\boldsymbol{w}_i(k)$, which is the ith weight vector, becomes $\boldsymbol{w}_i(k+1)$ by learning at the kth iteration. $\boldsymbol{x}$ is the training vector during the iteration. The mechanism of competitive learning is an iterative process. The weight vector of active neuron is adjusted slightly following each pattern presentation, so that the set of weight vectors adapts slowly to match the characteristics in the distribution of input vectors [7]. The procedure of updating the winning neuron is visualized in Figure 3.2. By updating, the winning neuron becomes closer to the training vector $\boldsymbol{x}$. Geometric interpretation of competitive learning is shown in Figure 3.3. There are three patterns with shapes of circle, diamond, and triangle. The initial three neurons shown with shape of asterisk and randomly distributed are far from each pattern in the initial state. Weight vectors are adjusted to resemble the input patterns during iterations of competitive learning. Weight vectors represent the input patterns in the final state after the iterations of competitive learning.

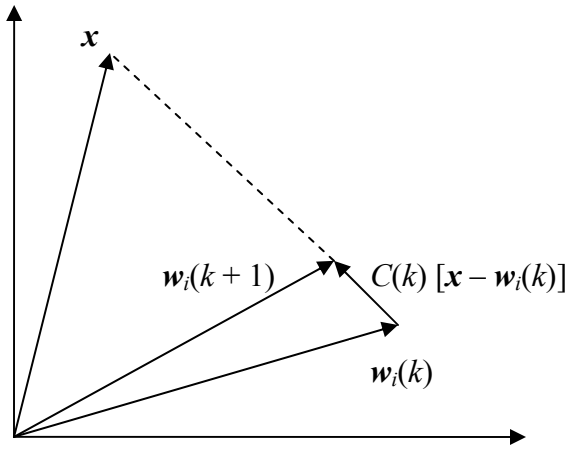
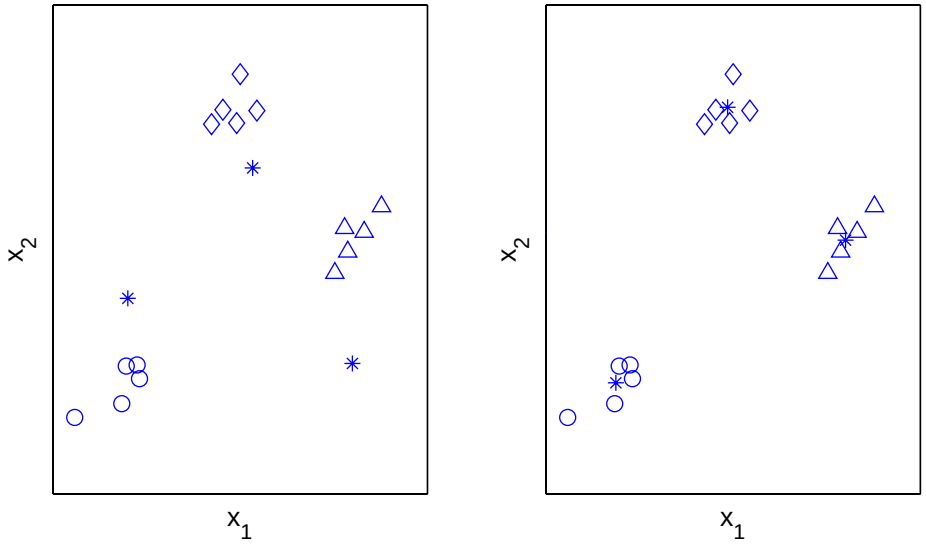


Figure 3.2 Procedure of updating the winning neuron.



(a) Initial state (b) Final state

Figure 3.3 Geometric interpretation of competitive learning.

3.3 Self-Organizing Maps

Self-organizing map (SOM) is an unsupervised learning algorithm similar to competitive learning. SOM produces a mapping from a multidimensional input space onto a one or two dimensional topology-preserving map of neurons. Hence, the underlying structure of the input space is kept while the dimensionality of the space is reduced. This is often used for visualizing and interpreting large high-dimensional data sets by mapping them to low dimensional space based on a competitive learning scheme [10]. SOM arranges feature vectors according to their internal similarity, creating a continuous topological representation of the input space. In a topological map, the vectors that are similar in space are grouped together, or clustered, while those vectors that are different are kept far apart. Due to its clustering property, SOM have been, for example, applied in image segmentation [87], [88], [89]. [90].

The main feature of SOM is to preserve a topological order in the map so that neighboring neurons respond to similar input patterns. Figure 3.4 shows one dimensional SOM structure. SOM is essentially a two layer neural network with full connections as shown in Fig 3.4. Output layer in Figure 3.4 consists of linearly arranged nodes fully connected to each input node with some weights. The biological basis of SOM is that

sensory inputs, such as visual and auditory inputs are mapped onto corresponding areas of

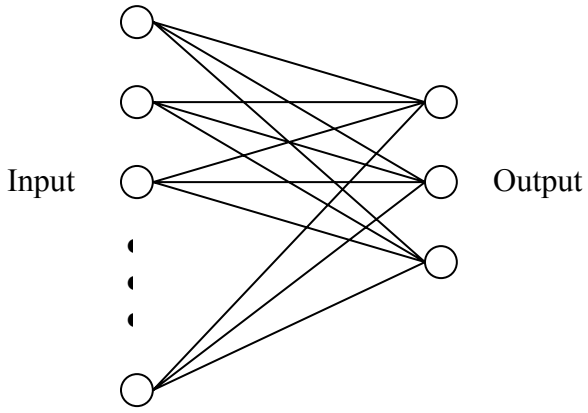


Figure 3.4 One dimensional self-organizing map.

the cerebral cortex in an orderly fashion. Learning in SOM is performed by a combination of three processes, which are competition, cooperation, and adaptation. Each neuron in a SOM is assigned a weight vector with the same dimensionality as the input data. For a given input pattern, the neurons in the network compete to see which one is closest to the input.

Suppose the weight vector of each neuron is represented as $\boldsymbol{w}_i=[w_{i1}\ldots w_{id}]^{\mathrm{T}}$, where d is the dimension of input vector. For each training step, distances between input vector $\boldsymbol{x}$ and all weight vectors are computed, and the winning neuron is found similarly as in competitive learning using (3.1). The winning neuron determines a neighborhood to spread the activation of the winning neuron over a neighborhood so that topologically close neurons will become sensitive to similar patterns. This is cooperation. During learning, the winning neuron and its topological neighbors are adapted to make their weight vectors more similar to the input pattern that caused the activation. Neurons that are closer to the winner will adapt more heavily than neurons that are further away. Through adaptation, neurons move closer to the input pattern. The magnitude of the adaptation is controlled with a learning rate, which decreases over time to ensure convergence of the SOM. The learning rule of SOM is as follows:

$$\begin{aligned} \boldsymbol{w}_i(k+1) &= \boldsymbol{w}_i(k)+\alpha(k)[\boldsymbol{x}-\boldsymbol{w}_i(k)] \quad \text{if } i=c \text{ or } i\in N_c \\ \boldsymbol{w}_i(k+1) &= \boldsymbol{w}_i(k) \qquad\qquad\qquad\qquad\quad \text{otherwise} \end{aligned} \tag{3.3}$$

$\boldsymbol{w}_i(k)$, which is the ith weight vector, becomes $\boldsymbol{w}_i(k+1)$ by learning at the kth iteration. c is the winning neuron and N_c is the neighborhood of c. $\boldsymbol{x}$ is the input training vector and $\alpha(k)$ is the learning rate. $\alpha(k)$ decreases linearly or exponentially with iterations. The neighborhoods of the winning neurons also shrink as the iteration number increases. In this thesis, the weight vectors are updated separately for each class by SOM as follows:

$$\begin{aligned} \boldsymbol{w}_i^j(k+1) &= \boldsymbol{w}_i^j(k)+\alpha(k)[\boldsymbol{x}^j - \boldsymbol{w}_i^j(k)] \qquad \text{if } i = c \text{ or } i \in N_c \\ \boldsymbol{w}_i^j(k+1) &= \quad \boldsymbol{w}_i^j(k) \qquad\qquad\qquad\qquad\quad \text{otherwise} \end{aligned} \tag{3.4}$$

where $\boldsymbol{w}_i^j(k)$, which is the ith weight vector of class j, becomes $\boldsymbol{w}_i^j(k+1)$ by learning. $\boldsymbol{x}^j$ is the input vector of class j.

The basic SOM algorithm is unsupervised, and it basically describes the degree of clustering of the input data. It projects input space on prototypes of a low-dimensional regular grid that can be effectively utilized to visualize and explore properties of the data [69]. However, when the SOM was originally suggested for classification tasks, it turned out that the class-separation of weight vectors, and thus also the classification could be improved by a significant amount if information about the class identity was taken into account during learning [10]. By this idea, SOM can be used for supervised learning. In order to make SOM supervised, the final weight vectors are labeled with class identity. This can be obtained by selection of training vectors belonging to each class. After updating of weight vectors, which is done separately for each class, by SOM learning rule, we can classify the input vectors based on the distance with labeled weight vectors. The unsupervised SOM constructs a topology-preserving representation of the statistical distribution of all input data. The supervised SOM tunes this representation to discriminate between classes. The weight vectors differently labeled define decision borders between classes [10]. This supervised SOM with rejection schemes are used for classification in this research.

3.4 Self-Organizing Global Ranking Algorithm

Recently, the self-organizing global ranking algorithm (SOGR) was proposed for the purposes of simplification of neighborhood concepts and reduction of computational load of self organizing maps [11]. SOGR is a simple iterative algorithm like competitive learning and SOM. However, it uses a different neighborhood concept from SOM. In

SOGR, neighbor neurons are chosen globally based on similarity ranking, and hence they can be chosen to be any neuron depending on similarity with the winning neuron regardless of topological neighborhood. This results in simple and faster computation. Initial weight vectors are randomly chosen from input training vectors with a predefined size. During training, an input vector is randomly chosen and the distance with the weight vectors are calculated. Euclidean distance can be used for distance measure. The closest neuron c is the winning neuron and the next closest rank-ordered neurons constitute the neighborhood of the winning neuron. Learning rule of SOGR is as follows:

$$\begin{aligned} \boldsymbol{w}_i(k+1) &= \boldsymbol{w}_i(k) + \beta_i(k)[\boldsymbol{x} - \boldsymbol{w}_i(k)] \text{ if } \quad i = c \text{ or if } i \in N_c \\ \boldsymbol{w}_i(k+1) &= \boldsymbol{w}_i(k) \qquad\qquad\qquad\qquad\quad \text{otherwise} \end{aligned} \tag{3.5}$$

where β_i is the learning rate and N_c is the neighborhood of the winning neuron c. $\beta_i(k)$ is the learning rate. As the number of iterations increases, N_c and $\beta_i(k)$ decrease. This helps reach to the global minimum or a very deep minimum.

In order to use the SOGR in supervised learning, it is necessary to label the weight vectors with appropriate classes. After updating of the weight vectors is finished, the distance between each training vector and all weight vectors are calculated. Then, the nearest weight vector is the winner, and a counter for the corresponding class is increased by one. This process is repeated for all the training data. After this process, the label of the counter which is the maximum for a weight vector is chosen as the label of the weight vector.

Equation (3.6) is an example of counter for labeling. The matrix to the right of weight matrix $\boldsymbol{w}$ is the counter. Each column of the counter corresponds to a class. Training data has 5 classes and the number of training data is 40, which is the sum of the elements of the counter. For all training data, we find the closest weight vector and add 1 to the column of the corresponding class at the corresponding row of the winning weight vector. For example, if the closest weight vector of $\boldsymbol{x}$ is $\boldsymbol{w}_1$, and $\boldsymbol{x}$ belongs to class 3, we add 1 to the position of the first row and the third column of the counter. After the counting procedure for all training data is finished, the column with the maximum number of the counter at each row corresponding to the closest weight vector is labeled with the class of the training vector.

$$\boldsymbol{w} = \begin{bmatrix} w_{11} & w_{12} & w_{13}.....w_{1D} \\ w_{21} & w_{22} & w_{23}.....w_{2D} \\ w_{31} & w_{32} & w_{33}.....w_{3D} \\ w_{41} & w_{42} & w_{43}.....w_{4D} \end{bmatrix} \quad \overbrace{\begin{bmatrix} 8 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 9 \\ 0 & 7 & 0 & 1 & 2 \\ 0 & 2 & 0 & 7 & 1 \end{bmatrix}}^{1\ 2\ 3\ 4\ 5} \tag{3.6}$$

According to (3.6), the class labels of the weight vectors are 1, 5, 2 and 4, respectively. In this particular example, class 3 is neglected. The testing process is simpler than the training process. For each test input vector, the closest weight vectors are found. Then, each test vector is labeled with the label of the winning weight vector.

3.5 Hierarchical Classifier with Rejection Schemes

Most errors in classification occur with the data which are close to boundaries between classes [8]. So if the hard vectors, which are difficult to classify, are detected before classification, the misclassification rate can be reduced by proper processing with the detected hard vectors. To improve classification reliability, identifying the input patterns which cause unreliable classifications is needed, and for this, rejection schemes have been applied [76], [77], [78], [79]. Rejection schemes aim at rejecting the possible input patterns which would otherwise be misclassified and make them processed in the next stages using newly constructed neural networks using rejected input patterns.

A hierarchical classifier consists of several stage neural networks (SNN). For each SNN, rejection schemes are constructed to detect whether the input data are hard to classify or not. Rejection schemes are constructed using training vectors and trained weight vector. Training vectors inside rejection boundaries are classified in each SNN, and the rejected training vectors are fed into the next SNN and can be processed with the same procedure as done in the previous SNN. Once the rejection schemes are constructed during training, testing is performed similar to training, using the constructed rejection schemes. By adopting rejection schemes, misclassification due to hard vectors is reduced. Rejection schemes make it possible to find the hard vectors before classification. Whereas the input vectors inside or on the rejection boundaries are classified in the current SNN, the input vectors outside rejection boundaries, which are rejected from the current SNN, are fed into the next SNN. In the next SNN, new weight vectors are selected from the rejected input training vectors from the previous SNN, and competitive learning is performed to update the weight vectors. Then, another rejection scheme is constructed as in the previous SNN. In this way, the rejection scheme is executed for

each SNN until there is no rejected input vector or the determined maximum number of SNNs, which is determined during training based on training accuracy, is exceeded. Figure 3.5 represents the block diagram of the training procedure with rejection schemes when the number of SNNs is 3. Since the input vectors classified in the current SNN are determined by rejection boundaries, how well the rejection boundaries are constructed is

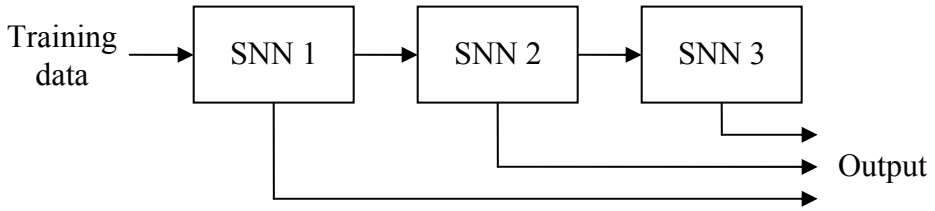


Figure 3.5 Block diagram of the training procedure of hierarchical classifier with rejection schemes.

important, and affects the classification performance. This is because the decision surface of classification is to a large degree determined by the rejection boundaries [9]. Rejection schemes are constructed depending on the training data and trained weight vectors.

In competitive learning with rejection schemes, learning of weight vectors is done for each class separately. Equation for learning in (3.2) is modified for this as follows:

$$\boldsymbol{w}_i^j(k+1) = \boldsymbol{w}_i^j(k)+C(k)[\,\boldsymbol{x}^j - \boldsymbol{w}_i^j(k)] \quad \text{if } i \text{ wins} \tag{3.7}$$
$$\boldsymbol{w}_i^j(k+1) = \boldsymbol{w}_i^j(k) \quad \text{otherwise}$$

where $\boldsymbol{w}_i^j(k)$, which is the ith weight vector of class j, becomes $\boldsymbol{w}_i^j(k+1)$ by learning. $\boldsymbol{x}^j$ is the training input vector of class j. Since competitive learning is performed for each class separately, rejection schemes are constructed separately for the training vectors in each class. Suppose we have input training data $\boldsymbol{x}$, which consists of C classes. $\boldsymbol{x}$ is D-dimensional data. Weight vectors for each class, $\boldsymbol{w}^j$, where j=1,2,…,C, are chosen from $\boldsymbol{x}^j$ belonging to the jth class. If the number of weight vectors per class is defined to be P, the total number of weight vectors will be $P\times C$. Weight vector for each class j is represented as in (3.8) and the total weight vector is represented as in (3.9):

$$\boldsymbol{w}^j = \begin{bmatrix} w_{11}^j & w_{12}^j & w_{13}^j \ldots\ldots w_{1D}^j \\ w_{21}^j & w_{22}^j & w_{23}^j \ldots\ldots w_{2D}^j \\ \bullet & \bullet & \bullet \ \ldots\ldots \bullet \\ w_{P1}^j & w_{P2}^j & w_{P3}^j \ldots\ldots w_{PD}^j \end{bmatrix} \tag{3.8}$$

$$\boldsymbol{w} = \begin{bmatrix} \boldsymbol{w}^1 \\ \boldsymbol{w}^2 \\ \bullet \\ \boldsymbol{w}^C \end{bmatrix} \tag{3.9}$$

For a training vector $\boldsymbol{x}_a^j$, which is the ath training vector belonging to the j-th class, where a=1,2,…, number of vectors in $\boldsymbol{x}^j$, and j=1,2,…,C, we find the winning weight vector $\boldsymbol{w}_i^j$, which has the minimum distance with $\boldsymbol{x}_a^j$. For all training vectors in each class, finding the winning neuron is thus performed. Then, the training vectors belonging to the jth class are separated into groups decided by the winning weight vectors. For example, a group of training vectors which belong to the jth class and their winning weight vector are $\boldsymbol{w}_i^j$ can be represented as in (3.10):

$$\boldsymbol{X}^j = \begin{bmatrix} x_{11}^j & x_{12}^j \ldots\ldots x_{1D}^j \\ x_{21}^j & x_{22}^j \ldots\ldots x_{2D}^j \\ \bullet & \bullet \ldots\ldots \bullet \\ x_{L1}^j & x_{L2}^j \ldots\ldots x_{LD}^j \end{bmatrix} \tag{3.10}$$

where x^j_{mn} is an element of $\boldsymbol{X}^j$ where m = 1,2,…,L, n = 1,2,…,D, i=1,2,…,P, and j=1,2,…C, L is the number of training vectors belonging to the group $\boldsymbol{X}^j$. Rejection boundaries are constructed for each column of $\boldsymbol{X}^j$ defined as follows:

$$RADP_{in}^j = \max_m \{x_{mn}^j\} \tag{3.11}$$

$$RADN_{in}^j = \min_m \{x_{mn}^j\} \tag{3.12}$$

$RADP^j_{in}$ and $RADN^j_{in}$ are the outer and inner rejection boundaries, respectively. The size of ***RADP*** and ***RADN*** is the same as that of weight vector $\boldsymbol{w}$. $\boldsymbol{RADP}^j$ for the training vectors belonging to the jth class and total ***RADP*** are shown in Eqs. (3.13) and (3.14). ***RADN*** has the same dimension as that of ***RADP***, and can be expressed similarly to Eqs. (3.13) and (3.14).

$$\boldsymbol{RADP}^{j} = \begin{bmatrix} RADP_{11}^{j} & RADP_{12}^{j}.....RADP_{1D}^{j} \\ RADP_{21}^{j} & RADP_{22}^{j}.....RADP_{2D}^{j} \\ \bullet & \bullet \quad ..... \quad \bullet \\ RADP_{P1}^{j} & RADP_{P2}^{j}.....RADP_{PD}^{j} \end{bmatrix} \tag{3.13}$$

$$\boldsymbol{RADP} = \begin{bmatrix} \boldsymbol{RADP}^{1} \\ \boldsymbol{RADP}^{2} \\ \bullet \\ \boldsymbol{RADP}^{C} \end{bmatrix} \tag{3.14}$$

For $\boldsymbol{RADP}^{j}$ and $\boldsymbol{RADN}^{j}$, a training vector $\boldsymbol{x}^{j}_{a}$, which is the ath training vector in the jth class, is determined to be inside or on the rejection boundary if it satisfies the following condition for every $n = 1,2,\ldots,D$:

$$RADN^{j}_{mn} \le x^{j}_{an} \le RADP^{j}_{mn} \tag{3.15}$$

$RADP^{j}_{mn}$ and $RADN^{j}_{mn}$ represent the elements located at the mth row and the nth column of $\boldsymbol{RADN}^{j}$ and $\boldsymbol{RADP}^{j}$ which are the rejection boundaries of the jth class training vectors. x^{j}_{an} is the nth element of $\boldsymbol{x}^{j}_{a}$. If at least one elements of $\boldsymbol{x}_{a}^{j}$ does not satisfy (3.15), $\boldsymbol{x}^{j}_{a}$ is outside the rejection boundary and is rejected from the current SNN.

Another rejection boundaries based on the distance between a training vector and weight vectors can be constructed [9]. With a group of training vectors which belong to the jth class and their winning weight vector being $\boldsymbol{w}_{i}^{j}$, obtained from (3.10), other rejection boundary can be constructed by

$$RAD_{i}^{j} = \max_{m} \left\| \boldsymbol{x}_{m}^{j} - \boldsymbol{w}_{i}^{j} \right\| \tag{3.16}$$

This is the maximum distance from the training vectors in the group $\boldsymbol{X}^{j}$ to its winning vector $\boldsymbol{w}_{i}^{j}$. If the training vector $\boldsymbol{x}^{j}_{a}$ satisfies the following equation, it is inside or on the rejection boundary:

$$\left\| \boldsymbol{x}_{a}^{j} - \boldsymbol{w}_{i}^{j} \right\| \quad \le \quad RAD_{i}^{j} \tag{3.17}$$

where a= 1,2,…,number of the training vectors belonging to the jth class.

Rejection boundary ***RAD*** can be used in place of ***RADPN*** (***RADP*** and ***RADN***). However, in the experiments, the result with ***RAD*** were less accurate than the results with ***RADPN***. Using ***RAD*** also takes more time since it needs to calculate distances between all training vectors belong to the jth class and their winning vector to decide ***RAD***.

Once learned weight vectors are obtained, rejection schemes are constructed as explained above. For the training vectors in current SNN, we determine which training vectors are rejected using the rejection scheme ***RADPN***. The training vectors rejected by ***RADPN***, which are outside the rejection boundaries, are fed into the next SNN and the training vectors inside or on the rejection boundaries are classified. If a training vector is inside any rejection scheme, it is classified in the current SNN. If there is no rejected training vector or the training accuracy of whole network reaches 100 %, or the training accuracy of the whole network including the new SNN is less than that of the previous whole network without the new SNN, the training procedures is stopped. This is the stopping criterion to determine the number of SNNs.

Each weight vector represents how the input nodes are interconnected with a particular output node identified with the weight vector. The output of an output node is set to 1 when a training vector is inside or on its rejection boundary and 0 when a training vector is outside its rejection boundary. If one or more weight vectors belonging to the same class have output 1, the class output becomes 1. If no weight vector belonging to a class has output 1, the class output becomes 0. If more than one class has output 1, the training vector is rejected. This means that more than one class is assigned to a training vector, and hence it is difficult to classify this training vector. These training vectors are also determined to be rejected and are fed into the next SNN. If the training vector is accepted by the two rejection schemes, but is classified to different classes, it is thus rejected from the current SNN.

An example of how rejection schemes are used is shown in (3.18). Suppose we have training vectors with three classes. The number of weight vectors per class is assumed to be two. The total weight matrix $\boldsymbol{w}$ can be represented as the matrix in (3.18). The three column vectors in (3.18) are the example of output of each node in $\boldsymbol{w}$ in terms of rejection boundaries. These are the trained result with three different training vectors $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, and $\boldsymbol{x}_3$, respectively. Here a node is represented by a row vector of $\boldsymbol{w}$. The first column is the node output for a $\boldsymbol{x}_1$. This column vector has output 1 only in $\boldsymbol{w}^l$, which is the first two rows in weight matrix $\boldsymbol{w}$. So, $\boldsymbol{x}_1$ is classified to class 1. The second column is

the node output for a $\boldsymbol{x}_2$. This column vector has output 1 only in $\boldsymbol{w}^2$, which is the fourth row in weight matrix $\boldsymbol{w}$.. So, $\boldsymbol{x}_2$ is classified to class 2. The third column is the node output for a $\boldsymbol{x}_3$. Class output of $\boldsymbol{w}^1$, $\boldsymbol{w}^2$, and $\boldsymbol{w}^3$ are all 1 since one or more weight vectors belonging to

$$\boldsymbol{w} = \begin{bmatrix} w_{11}^1 & w_{12}^1 & w_{13}^1 \ldots\ldots w_{1D}^1 \\ w_{21}^1 & w_{22}^1 & w_{23}^1 \ldots\ldots w_{2D}^1 \\ w_{11}^2 & w_{12}^2 & w_{13}^2 \ldots\ldots w_{1D}^2 \\ w_{21}^2 & w_{22}^2 & w_{23}^2 \ldots\ldots w_{2D}^2 \\ w_{11}^3 & w_{12}^3 & w_{13}^3 \ldots\ldots w_{1D}^3 \\ w_{21}^3 & w_{22}^3 & w_{23}^3 \ldots\ldots w_{2D}^3 \end{bmatrix} \quad \overbrace{\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}}^{\text{Example of node output}} \tag{3.18}$$

each class have output 1. This means that the training vector can be classified to any class since it is on the boundary of class. This is a hard vector. It is rejected and fed into the next SNN.

When training is stopped by stopping criterion, if there are still rejected training vectors, these remaining training vectors are classified to the class of the closest weight vector. Once training is stopped by stopping criterion, no more SNNs are generated. This is because additional SNN may lower training accuracy of the whole network. This may occur, for example, when there are imbalanced data, that is, at least one of the classes constitutes only a very small minority of the data. In this case, SNN aims to minimize the overall error rate, rather than taking special attention to a class with few data. If there are few data points, left from a number of classes, generating a complex SNN with many parameters does not make sense since the parameters can not be accurately estimated. Thus, rather than generating more SNNs, the remaining training vectors are classified in the nearest sense.

Test is performed using the classifier constructed by training. Testing procedure is similar to the training procedure. By rejection boundaries constructed with the training vectors, all test vectors are tested for acceptance or rejection. Test vectors inside or on the rejection boundaries are classified in the current SNN, and the test vectors outside rejection boundaries are rejected and fed into the next SNN. Testing is continued until there is no test vector or the determined number of SNNs, which is determined during training, is exceeded. If there are some test vectors when the determined number of SNNs

is exceeded, those test vectors are classified to the class of the closest weight vector as in training.

Hierarchical classifier can be constructed using SOM or SOGR. The structure of SOM with rejection schemes is the same as competitive learning with rejection schemes except for the learning method. They consist of a number of SNNs. Each SNN uses SOM as the learning method and rejection scheme is constructed for each SNN. Rejection schemes are constructed with the same method as competitive learning with rejection schemes. Initially, weight vectors are selected from the training vectors. Random initialization is commonly used in SOM since the randomly chosen unordered weight vectors will become ordered by a number of iterations of learning, eventually. When the computation of weight vectors is finished by SOM, rejection boundaries are constructed to reduce the number of misclassified training vectors. Since the learning method of SOM and SOGR is the same as that of competitive learning except the concept of neighborhood, rejection schemes can be constructed in the same way as in competitive learning.

To show how the hierarchical classifier with rejection schemes is working, classification of randomly generated two channel data with two classes is performed. If the number of channel is over than 3, it can not be visualized. Since multispectral images have several decades of channels, it is impossible to visualize all channels. To make it simple and show it graphically, two channel data with two classes are used. Fig 3.6 shows randomly generated two class data. 460 data are generated and 60 data are taken as training data and the remaining 400 data are taken as test data. For each class, there exist 30 training data and 200 test data are used. Each data point in the Figure 3.6 has the form of $\boldsymbol{x}=[x_1\ x_2]^T$. This is the two channel data. Mean of class 1 data is $[20\ 20]^T$ and mean of class 2 is $[22\ 20]^T$. Competitive learning with 30 iterations is used as learning method in this example.

To construct rejection boundaries, learning is performed using training data. Initial weight vector is chosen randomly before learning is performed. In this example, the number of weight vector for each class is set to one. So, there will be only one cluster for each class. Learning is performed for the training data of each class separately. Clustered result by competitive learning is shown in Figure 3.7. Initially randomly chosen weight vector is updated by iterations of competitive learning. After learning, the weight vector becomes the center of cluster. Asterisks in the Fig 3.7 represent the center of each cluster. Since one weight vector is used for each class in this example, one cluster is generated for each class.

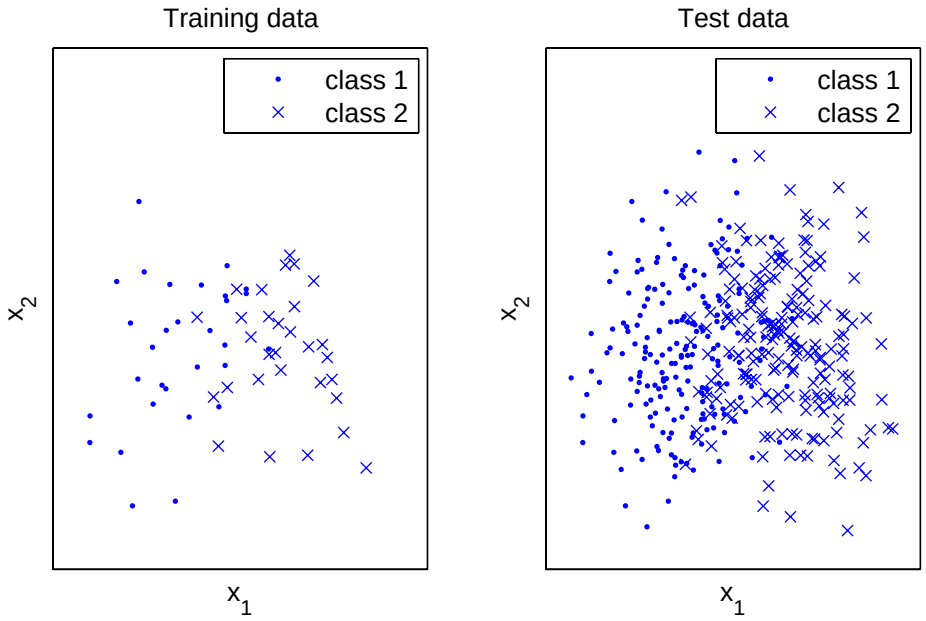


Figure 3.6 Training and test data with two classes.

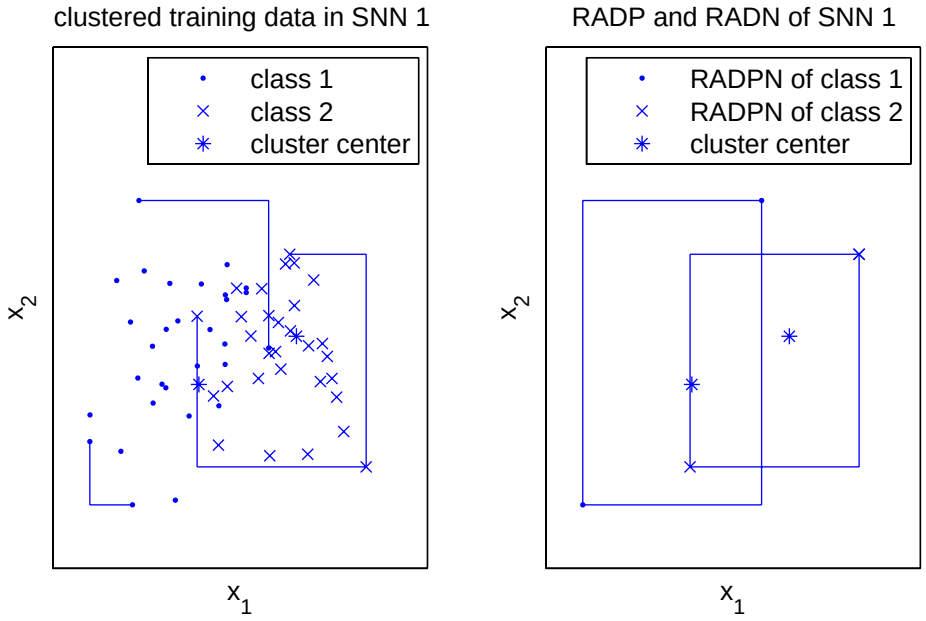


Figure 3.7 Clustered results and constructed rejection boundaries in the SNN 1.

Using the clustered results, rejection boundaries are constructed. The maximum and minimum values for x_1 and x_2 can be recognized in the clustered results. Maximum values for each channel are x_{1max} and x_{2max}. Minimum values for each channel are x_{1min} and x_{2min}. Then, intersection point of two lines $x_1 = x_{1max}$ and $x_2 = x_{2max}$ will be RADP,

which is the outer rejection boundary. Intersection point of two lines $x_1 = x_{1\min}$ and $x_2 = x_{2\min}$ will be ***RADN***, which is the inner rejection boundary. Rejection boundaries are constructed with ***RADP*** and ***RADN***. These are in the shape of rectangle as shown in Figure 3.7. These rectangular shaped rejection boundary is constructed for each class and for each cluster. If the number of cluster for each class is set to two, two rejection rectangles will be constructed for each class, and totally four rejection rectangles are constructed since there are two classes in this example.

Another rejection boundary can be constructed using the clusterd result. This is circular shaped. The radius of this circular rejection boundary is the maximum diatance between the cluster center and each element in that cluster as in (3.16). Figure 3.8 shows both ***RADPN*** and ***RAD*** on training data constructed in SNN 1. However, the intersection area of two circles, which is circular rejection boundary for each class, is bigger than that of ***RADPN***. In addition, calculation of distances between cluster and the all elements of cluster to find the maximum distance, it takes long time. Actually, ***RAD*** is not good as ***RADPN*** in classification of multispectral images. Thus, only ***RADPN*** is used for rejection scheme both in this example and in the experiments on multispectral images of this thesis.

Once the rejection boundaries are constructed, training data are to be decided whether they are accepted or rejected. From the constructed rejection boundaries, we can

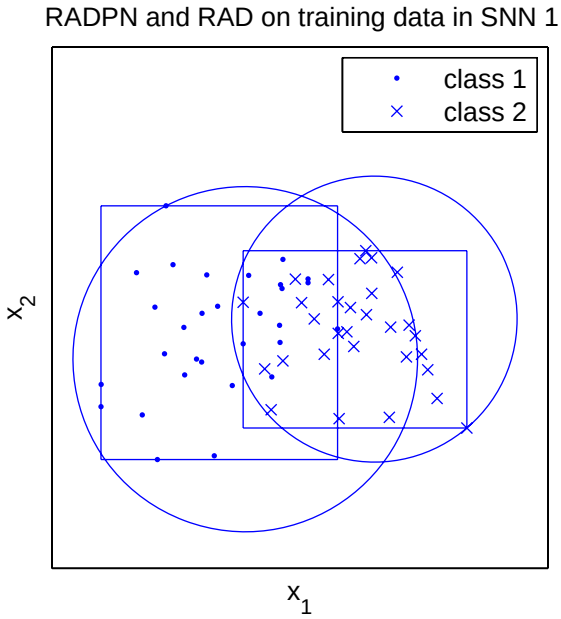


Figure3.8 ***RADPN*** and ***RAD*** constructed on training data in SNN 1.

recognize the intersection area of two rectangles. If the training data are inside the intersection area of two rectangles, it is difficult to classify since they are located on near the boundary of two classes. These are hard vectors. The classification of these data is not performed in the current SNN since these data can be misclassified if there are classified in the current SNN. These data are rejected by current SNN and fed into the next SNN. In the next SNN, new rejection boundaries are constructed by competitive learning using the rejected data, and the classification which will be done in the next SNN is more reliable. If the training data are inside the rejection rectangle for each class but outside or on the intersection boundary of two rectangles, they are accepted and classified in the current SNN. It is easy to classify these data, since they are far away from the class boundary.

Similarly, the same procedure done in training data is applied for the test data. For the given test data for current SNN, whether accepted or rejected is decided based on rejection rectangles which are constructed using training data. Test data which are inside of rejection rectangle but outside of the intersection area of two rejection rectangles are accepted and classified. The test data which are outside the rejection rectangles and inside the intersection of two rejection rectangles are rejected from current SNN and their processing is postponed in the next SNN. Figure 3.9 represents the training and test data which are accepted and classified in the SNN 1. Figure 3.10 represents the training and testing data which are rejected by SNN 1. These data are to be fed into the next the SNN.

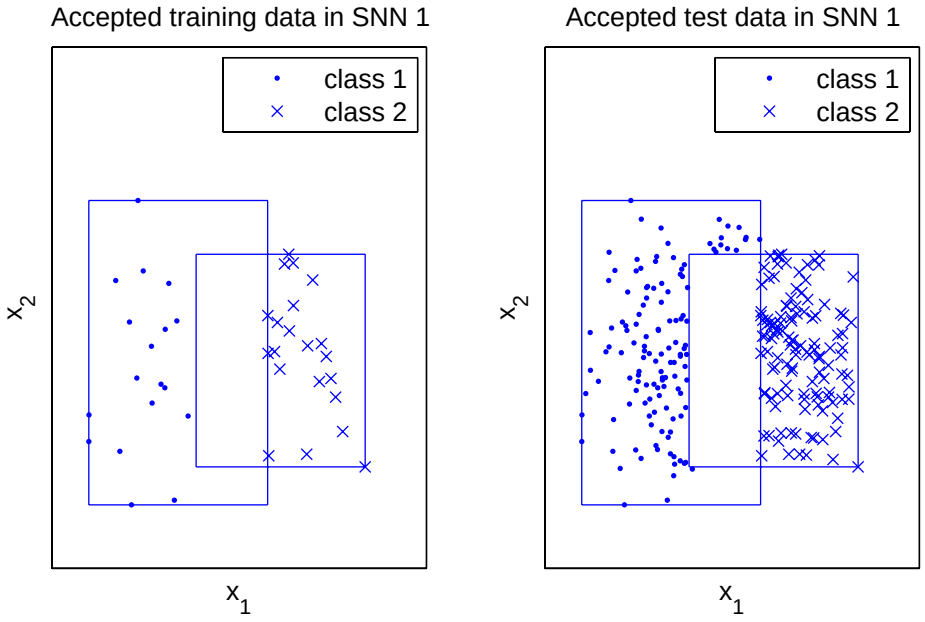


Figure 3.9 Accepted training and test data in the SNN 1.

Figure 3.10 Rejected training and test data in the SNN 1.

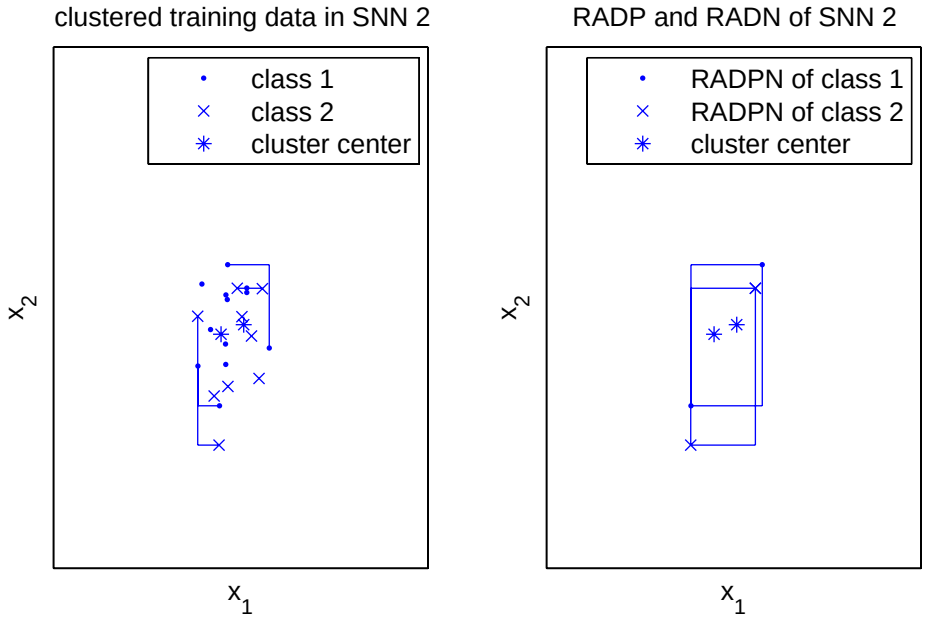


Figure 3.11 Clustered results and constructed rejection boundaries in the SNN 2.

In SNN 2, the same procedure done in the SNN 1 is performed. Competitive learning is performed using the training data of SNN 2 which are rejected in the SNN 1 and cluster center is decided. Figure 3.11 represents the clustered result and constructed rejection

boundaries in the SNN 2. Only the training data which are rejected in the SNN 1 are used for clustering. Figure 3.12 represents the accepted and classified training and test data up to SNN 2. The data which are outside the intersection area of two rejection rectangles are training and test data which are classified up to in the SNN 2. The data inside the two rectangles but outside the intersection of two rectangles are classified data in the SNN 2. Figure 3.13 represents the training and test data which are rejected in the SNN2. In this example, the number of the SNN is set to two and the rejected data in the SNN are classified to the class of the nearest cluster center. Figure 3.14 represents rejection boundaries and classified data which are performed up to SNN 2. Rectangles with solid line are constructed in the SNN 1 and rectangles with dashed line are constructed in the SNN 2. The data in the intersection area of two rectangles with dashed line are rejected in SNN 2, which are shown in Figure 3.15, and will classified to class of the nearest cluster center.

Class separability of each SNN is calculated. Transformed divergence and J-M distance of training and test data for each stage are given in Table 3.1 and Table 3.2, respectively. Since the easy data which are inside the rejection boundary constructed in SNN 1 have higher separability, both transformed divergence and J-M distance for the data classified in SNN1 are greater than those of input to SNN1. Transformed divergence

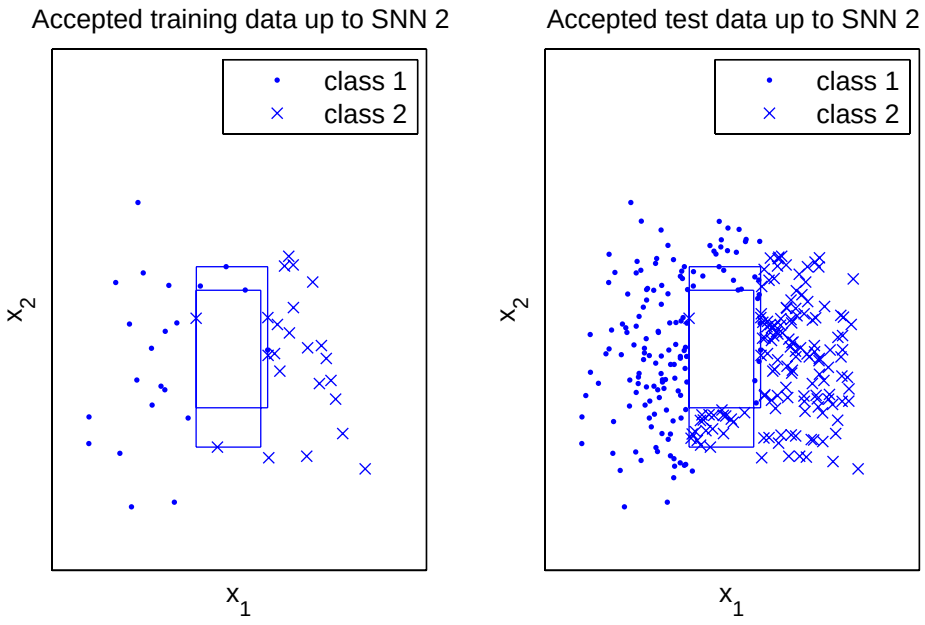


Figure 3.12 Classified training and test data up to SNN 2.

Figure 3.13 Rejected training and test data from the SNN 2.

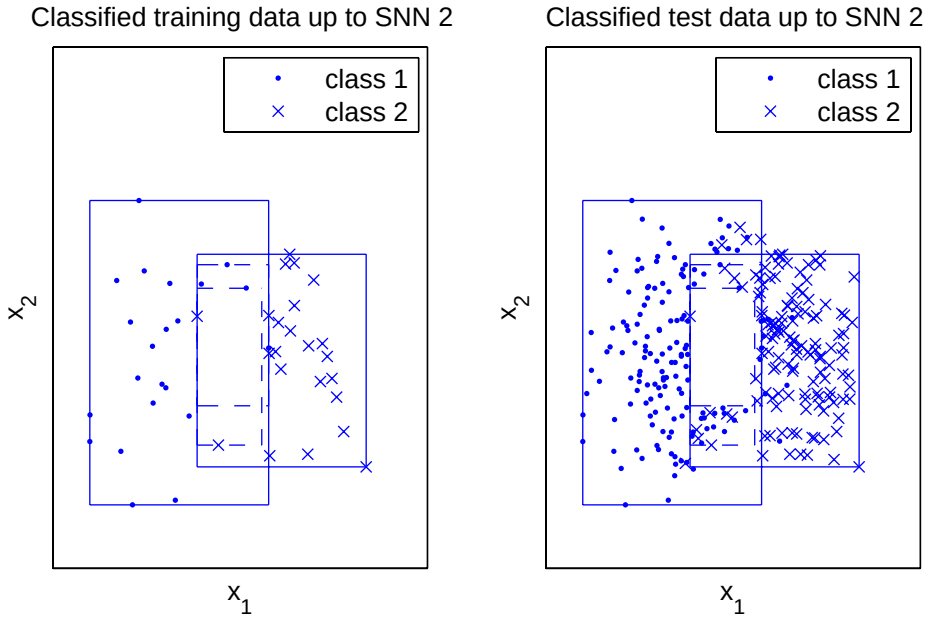


Figure 3.14 Rejection boundaries and classified training and test data up to SNN2.

and J-M distance for the rejected data by SNN 1 are much lower than those of input to SNN 1. This means that the rejected data by SNN 1 are very difficult to classify. This is the reason why the transformed divergence and J-M distance classified up to SNN 2 are lower than those of the data classified in SNN 1.

Figure 3.15 Rejected training and test data in the SNN 2.

Table 3.1 Transformed divergence of input data for each stage

Data type	Training data	Test data
Input to SNN1	1106.08	1000.85
Classified in SNN1	1981.58	1755.79
Rejected by SNN1	171.05	396.62

Table 3.2 J-M distance of input data for each stage

Data type	Training data	Test data
Input to SNN1	1.052	0.950
Classified in SNN1	1.970	1.678
Rejected by SNN1	0.168	0.434

Hierarchical classification improves classification performance by detecting the hard input data and classifying easy data first and hard data later rather than classifying at the same time.

4. NONLINEAR IMAGE FILTERING

Classification performance depends on several factors such as class separability, training sample size, dimensionality, classification algorithm, and so on. Image filtering can be used as a means for increasing class separability. Image filtering typically refers to the process of noise reduction and smoothing in the image by neighborhood operations that calculate a new value for the center pixel based on the values of its neighbors within a window. For preserving image details such as edges, nonlinear filtering is appropriate. Nonlinear filtering makes it possible to reduce noise but preserves edges without blurring. By nonlinear filtering, the variances of pixels within each class are reduced. Therefore, the gap between classes in the feature space is widened and class separability is improved. This results in higher classification accuracy. This also improves visual interpretation. In practical, image filtering have been applied in remote sensing for the improvement of classification performance [16], [52], [53].

4.1 Median Filter

Median filter is a simple and effective noise removal filter. Filtered pixel is determined by taking the median of the pixels contained in a window around the pixels [18]. Median filtering requires arranging the pixel values in the window in increasing or decreasing order and picking the middle value. This smoothes the image while preserving the small and sharp details. Median filter is good for removing pepper and salt noise, which has isolated pixel value from its neighborhood.

4.2 Morphological Filter

Mathematical morphology or simply morphology is a theory for analysis of spatial structures and aims at analyzing the shape and form of objects [19], [54]. Morphological filter is a nonlinear signal transformation that locally modifies geometric features of signals [55]. Morphological filters are suited to suppression or extraction of image objects or structures. Noise reduction and edge detection in image analysis are two

possible applications for them. Using mathematical morphology, various morphological filters can be constructed [56], [57], [58], and also applied to remote sensing image analysis [59].

Morphological operators are used for extracting relevant structures of the image. This is achieved by using a structuring element. The structuring element is to mathematical morphology what the convolution kernel is to linear filter theory. The structuring element can be chosen as a 3×3 window and has its origin at the center pixel. Some examples are shown in Figure 4.1. The blanks in the structuring elements are zeros. The structuring element moves over the image and its elements are compared with the set of the underlying pixels at each pixel of the image. Structuring elements consist of a pattern specified at the coordinates of a number of discrete points relative to some origin. Morphological operators take two input data. One is the input image, which may be either binary or grayscale. The other one is the structuring element. The shape of the structuring element is usually chosen according to some a priori knowledge about geometry of the relevant and irrelevant image structures. Irrelevant structures mean noise or other objects to be suppressed.

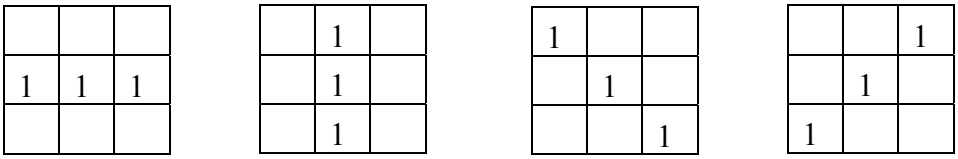


Figure 4.1 Examples of structuring elements.

4.2.1 Morphological operators

The two most basic operations in mathematical morphology are erosion and dilation. Morphological operators take an image to be processed and a structuring element as input data. The two pieces of input data are treated as representing sets of coordinates in a way that is slightly different for binary and grayscale images. For a binary image, white pixels are normally taken to represent foreground regions, while black pixels denote background. Then, the set of coordinates corresponding to that image is simply the set of two dimensional Euclidean coordinates of all the foreground pixels in the image, with an origin normally taken in one of the corners so that all coordinates have positive elements. For a grayscale image, the intensity value represents height above a base plane, so that the grayscale image represents a surface in the three-dimensional

Euclidean space. Then, the set of coordinates associated with this image surface is simply the set of three-dimensional Euclidean coordinates of all the points within this surface and also all points below the surface, down to the base plane. Note that even when we are only considering points with integer coordinates, this is a lot of points, so usually algorithms are employed that do not need to consider all the points.

Let f and h be two discrete-valued functions defined on a two dimensional discrete space. f is the input image and h is a structuring element. Four basic morphological operators are defined as follows:

Erosion : The effect of the erosion operator on an image is to erode away the boundaries of regions of foreground pixels (i.e. white pixels, typically). Thus areas of foreground pixels shrink in size, and holes within those areas become larger. This operation shrinks image regions. This means that the output image tends to be darker than the input image. The image f is eroded by

$$(f \ominus h)(x, y) = \min\{f(x+i, y+j) \; ; (i, j) \in D_h\} \tag{4.1}$$

where h is a structuring element. h is a two dimensional grayscale image with a finite domain (D_h), similar to a filter. Erosion of an image is defined as the minimum of the pixels of a local region of an image decided by a grayscale structuring element. Grayscale erosion is used to smooth small light regions. Light elements within the image are reduced or eliminated, depending on how their shapes are related to the used structuring element. The shape of the input structuring element is generally chosen to emphasize or de-emphasize elements in the image. The degree of these effects depends greatly on the shape and values within the structuring element and by the details within the image itself.

Dilation: The effect of the dilation operator on an image is to gradually enlarge the boundaries of regions of foreground pixels. Thus, areas of foreground pixels grow in size while holes within those regions become smaller. This operation grows image regions. This means that the output image tends to be brighter than the input image. This operation is the dual operation of erosion, and is defined by

$$(f \oplus h)(x, y) = \max\{f(x+i, y+j) \quad ; (i, j) \in D_h\} \tag{4.2}$$

Dilation of an image is defined as the maximum of the pixels of a local region of an image decided by a grayscale structuring element. Grayscale dilation is used to smooth small dark regions. Dark elements within the image are reduced or eliminated, depending on how their shapes relate to the structuring element used.

Opening: Opening of an image is an erosion followed by a dilation. The result is the reduction of small positive regions within the image. Equation for opening is defined by

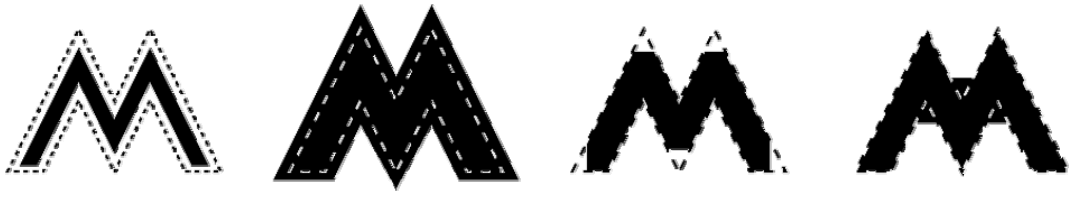
$$(f \circ h) = (f \ominus h) \oplus h \tag{4.3}$$

The basic effect of an opening is somewhat like erosion in that it tends to remove some of the bright pixels from the edges of regions of foreground pixels. However, it is less destructive than erosion, in general. Opening is used to remove small objects from an image while preserving the shape and size of larger objects in the image. The result is the reduction of small brighter regions within the image.

Closing: Closing is the reverse of opening. This is a dilation followed by an erosion. The effect of the operator is to preserve darker regions that have a similar shape to the structuring element. Equation for closing is defined by

$$(f \bullet h) = (f \oplus h) \ominus h \tag{4.4}$$

Four morphological operations are illustrated in Figure 4.2



(a) Erosion (b) Dilation (c) Opening (d) Closing

Figure 4.2 Morphological operations.

4.2.2 Generalized morphological filter

Generalized morphological filter (GMF) uses multiple structuring elements and combines linear and morphological operations [20]. The GMF suppresses various types of noise, but preserves geometrical structure in an image. The block diagram of GMF is shown in Figure 4.3. GMF consists of a cascade of two stages of morphological operations. The input image is closed and opened independently, then fed into opening and closing. Then the outputs are summed with equal weight of 0.5. All operations in the filter are with structuring elements shown in Figure 4.1. Those structuring elements define four directions of images. In each stage, image is processed with 4 structuring elements separately and the mean of the results with 4 structuring elements becomes the output of each stage. The GMF reduces noise while preserving geometrical structure.

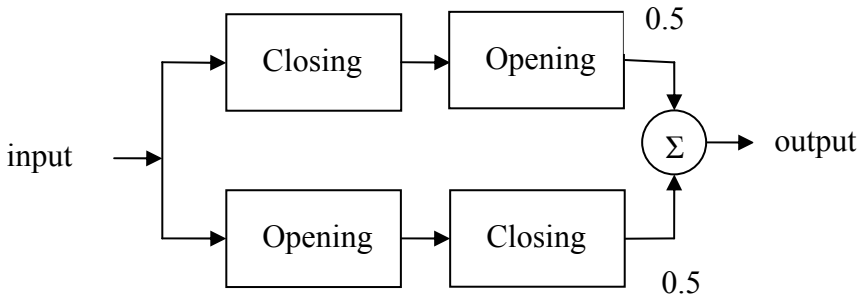


Figure 4.3 Block diagram of generalized morphological filter.

4.3 Spatial Filter

The spatial filter is a simple mean filter, which uses variance as a homogeneity criterion [16]. First, a 5×5 window is used to determine homogeneity. If the variance of the pixels in this window is less than a chosen threshold, the mean value of the pixels in the window is assigned to the center pixel. If not, the window size is reduced to 3×3, and the variance of the pixels in the window is examined. If the variance is less than a chosen threshold, the mean value of the pixels in the window is assigned to the centered pixel. If not, 3 minimum and 3 maximum values are excluded from the window and the mean value of the remaining pixels is assigned to the center pixel. For each channel image, this filter is applied to reduce noise. Spatial filtering results in more homogeneous regions. The variance in homogeneous regions is reduced since the pixel values in a homogeneous region approach the mean value of the region. This also results in improved spectral separability.

5. CONSENSUS CLASSIFIER

In supervised classification of remotely sensed images, there are a number of classification algorithms available including statistical and neural networks. Usually, for a specific problem, each of these classifiers could attain a different degree of success, but maybe none of them is totally perfect [27]. Experiments reported by the authors and other researchers in the literature have shown that the superiority of one algorithm over another cannot be claimed for remote sensing image classification [36]. Even though an algorithm performs well on a certain image, it may yields poor performance on other images. There is no classification algorithm which shows the best performance in all images and all situations. This is the main reason why the methodology of integrating the results of a number of different classification algorithms so that a better result could be obtained is needed.

Consensual classification is one of the strategies to improve classification performance and obtain more reliable and accurate result. This is combining the results of several different classifiers to obtain the improved classification results for the given patterns. The results of multiple classifiers would be the basis for choosing one of the classifiers as a final decision to the classification problem. The main idea of consensual classification is that group decision by combining individual opinions to derive a consensus is better than a single decision. This suggests that a different classifier could potentially offer complementary information in classification even when the result of classification by it self is not good.

In the field of pattern recognition, there have been many methods using multiple classifiers to combine outputs of different single classifiers to develop high performance classification systems [27], [28]. Typically, the classifier outputs are combined by majority voting rules, statistical techniques, and other combining methods [27]. For a given input pattern, each single classifier performs classification and then the different results generated by single classifiers are combined using a consensus scheme to decide the collective classification. To obtain improved performance by consensual classification, the classification errors generated by single classifiers should be different

from each other. For example, when neural networks are used for classification, if the different neural networks generalize in the same way, there is no advantage to combining a set of neural networks [30]. Even marginal classifiers can contribute to improve classification performance if they are combined with other classification results which have independent classification errors from each other. Since the classification performance by combining multiple classification results is improved only if the classifiers used for combining make independent errors, the design of error-independent classifiers is important as well as how to combine multiple classification results [29]. Most combination methods described in the literature assume that multiple classifier systems to obtain combined result are made up of classifiers making independent classification errors.

5.1 Neural Network Ensembles

When combining multiple classifiers, if each single classifier is composed of neural networks, each of which is a general function approximator, the set of neural networks is sometimes called neural network ensembles [34]. The members of ensemble are combined in order to obtain better generalization performance than that of any other single neural network. The basic idea underlying neural network ensembles is to find ways of exploiting the information of ensemble members. In a neural network ensemble, there are two main issues. One is how to make or select candidate ensemble members to be combined in an ensemble and the other one is how to combine the outputs of the ensemble members [30].

5.1.1 Methods for generating ensemble members

To generate members of a neural network ensemble, one important consideration is the extent by which the neural networks forming the ensembles show error diversity in the sense that they make different errors. When errors generated by ensemble members are different, the improvement of classification is possible by combining their outputs. Therefore, the design of neural network ensembles involve creating a set of networks which show the highest possible degree of error diversity [35].

Several methods which create members of neural network ensemble making different errors have been developed. Such methods basically involve varying the parameters related to the design and training of neural networks. These methods include

varying the initial random weights, varying the network architecture, varying the network type, and varying the training data [30]. By selecting initial weight vectors randomly, neural networks can be trained differently and thereby reach different local minima. Varying the training data which is most frequently used for creation of ensemble member include sampling training data, disjoint training data, and preprocessing. Sampling data is a common method to the creation of ensemble members. This makes the network trained on a different subsample of the training data. Disjoint training sample is a similar method to the sampling data. This is the use of mutually exclusive training set. There is no overlap between data used to train different neural networks. Preprocessing is altering input data to extract different features from the input data. Nonlinear transformation is one possible method for this purpose.

Partridge experimentally compared the capabilities of the above methods to make error independent networks and concluded that varying the network type and the training data are the two best ways for creating ensembles of networks making different errors [31]. To make ensemble members, two basic strategies are possible. One is aimed at generating an ensemble of error independent networks directly [32]. The other one is to overproduce and to choose a method based on the creation of an initial large set of networks and the subsequent choice of the subset of the most error independent networks. For the second method, some error diversity measures which can be used to choose a subset of independent classifiers are used [33]. One simple method to overproduce and to choose a method is to investigate all possible combinations of results of classifiers to be combined. If one combination yields the best training accuracy by some combining rule, then members of this combination is also used in testing. However this exhaustive search to find the best combination takes long time.

When a single training algorithm is used, it is possible to create members of an ensemble by multiplying input pattern by a random matrix or varying initial weight vectors randomly. Neural network ensembles are desirable due to the basic fact that selection of the weight vectors $\boldsymbol{w}$ is an optimization problem with many local minima. All global optimization methods in the face of many local minima yield optimal parameters ($\boldsymbol{w}$) which differ greatly from one run of the algorithm to the next, i.e., which show a great deal of randomness stemming from different initial weight vector and sequencing of the training examples. This randomness tends to differentiate the errors of the networks, so that the networks will be making errors on different subsets of the input space [34], [93].

The networks will differ in the values of the weight vectors $\boldsymbol{w}$. These different weight vectors correspond to different ways of forming generalizations about the patterns inherent in the training set. As each network makes generalization errors on different subsets of the input space, the collective decision produced by the ensemble is less likely to be in error than the decision made by any of the individual networks.

5.1.2 Problems of combining multiple classifiers

There are three categories in the problem of combining the results of multiple classifiers according to the levels of information produced by various classifiers. Suppose that we have pattern P with M classes $(\omega_1,\omega_2,\ldots,\omega_M)$ and K different classifiers cl_k, $k=1,2,\ldots,K$, which can be used for combining results of multiple classification of pattern P. The job of classifier cl is to assign a sample $\boldsymbol{x}$ from pattern P one index j, $j=1,2,\ldots,M$ as a label to represent that $\boldsymbol{x}$ is regarded as being from class ω_j. Regardless what internal structure a classifier has, we may simply regard a classifier as a function that receives an input sample $\boldsymbol{x}$ and output a label j like $cl(\boldsymbol{x})=j$.

Although j is the only output information needed at the final stage of classification, practically many of the existing classification algorithms usually provide or are able to provide some other related information. For example, a Bayes classifier may also provide M values of post probabilities $P(\omega_i|\boldsymbol{x})$, $i=1,2,\ldots,M$ for each possible label. In fact, the final label j is the result of maximum selection from the M values, and this selection certainly discards some information that is considered useless for the final output when there is only a single classifier. However such discarded information may be useful for combination of multiple classifiers. Depending on whether some output information other than one label j is used and the other kind of information is used, there are different types of combining multiple classifiers [27].

In general, the output information from various classification algorithms can be categorized into three levels: the abstract, the rank, and the measurement levels [27], [38]. In the abstract level, a classifier only outputs a unique label. For the rank level, a classifier ranks all labels or a subset of the labels in a queue with the label at the top being the first choice. For the measurement level, a classifier attributes to each class a measurement value that reflects the degree of confidence that a specific input belongs to a given class. Among the three levels, the measurement level contains the highest amount of information while the abstract level contains the lowest. The methods of combining multiple classifiers are varied based on which output information levels are used. When

combining is based on the abstract level, the individual classifier could be very different from each other in their theories or methodologies. In fact, any kind of classifiers will at least supply the output information at the abstract level, so the combining multiple classifiers based on the information of the abstract level covers all kinds of pattern recognition areas. For this reason, the abstract level is used for combining multiple classifiers in this thesis. In contrast, if the measurement level is used for combining, this requires that all the individual classifiers should be able to supply the output information at the measurement level. Furthermore, if there are any measurement vectors of different kinds, the measurements should be able to be transformed into the same kind of measurement, since a reasonable combination operation on these measurements could be made only when they have the same measurement scale.

5.1.3 Combination by majority rule

When only the abstract level is available, which is the most general, majority rule or voting rule can be used. We have K different classifiers. Each classifier provides the results in terms of the class labels assigned to the patterns such as $cl_k(\boldsymbol{x})=j$, $k=1,2,\ldots,K$. A given input pattern receives K classification labels from the multiple classifiers, each label corresponding to one of the M data classes. Equation (5.1) does not usually hold:

$$cl_1(\boldsymbol{x}) = cl_2(\boldsymbol{x}) = \ldots = cl_K(\boldsymbol{x}) \tag{5.1}$$

Each single classifier may make different decision. A common rule to combine the results of the K classifiers is majority rule by voting [39], [40]. This is that the data class that receives a larger number of votes is assumed to be the class of the input pattern.

To combine the multiple classification results, counter is needed. $\boldsymbol{C}_{\text{training}}$ and $\boldsymbol{C}_{\text{test}}$ are counter for training and test results.

$$\boldsymbol{C}_{training} = \begin{bmatrix} 0 & 0.....0 & 0 & 0 & 0 \\ 0 & 0.....0 & 0 & 0 & 0 \\ \bullet & \bullet....\bullet & \bullet & \bullet & \\ 0 & 0.....0 & 0 & 0 & 0 \end{bmatrix} \tag{5.9}$$

In the counter for trained result $\boldsymbol{C}_{\text{training}}$, the number of rows is the number of training vectors and the number of columns is the number of classes of input image. We have multiple classification results and corresponding weights λ_i. Weights λ_i for the output of jth classifier can be calculated by least squares based on the goodness of classification results. To make it simple, weights for all single classification results can be 1 regardless of the goodness of each classification result. For each training vector, the column corresponding to the trained result of each single classifier is increased by corresponding weights λ_i. When counting process for all training vectors is finished, the column number with maximum becomes the class of the training vector. Similarly, $\boldsymbol{C}_{\text{test}}$ is obtained and final consensual test result is obtained.

5.1.4 Combination by Bayesian average

Bayesian average is used when the measurement level is available as output information of classifiers. In some classification algorithms such as the K-nearest neighbor (*KNN*), it is possible to calculate the estimates of the posterior probabilities that an input pattern $\boldsymbol{x}$ comes from the class ω_i. When K classifiers are available, each classifier can provide the following estimate:

$$p_k(\omega_i|\boldsymbol{x}), \qquad i = 1,\ldots,M,\ k = 1,\ldots,K \tag{5.2}$$

For each single classifier cl_k, a decision is made as

$$cl_k(\boldsymbol{x}) = j \quad \text{with} \quad p_k(\omega_j \mid \boldsymbol{x}) = \max_i p_k(\omega_i \mid \boldsymbol{x}) \tag{5.3}$$

If each single classifier can provide the estimate as in (5.3), the average value of theses estimates can be calculated as follows:

$$p_{avg}(\omega_i \mid \boldsymbol{x}) = \frac{1}{K}\sum_{k=1}^{K} p_k(\omega_i \mid \boldsymbol{x}) \quad i = 1,\ldots,C \tag{5.4}$$

Then, the final collective decision is made as follows:

$$CL\,(\boldsymbol{x}) = j \text{ with } \quad p_{avg}(\omega_j \mid \boldsymbol{x}) = \max_i p_{avg}(\omega_i \mid \boldsymbol{x}) \tag{5.5}$$

where *CL* is the classifier which combines multiple classifiers. *CL* classifies input pattern based on the Bayesian criterion,

5.2 Two Types of Generating and Combining Multiple Classifiers

In this thesis, two types of combining multiple classifiers are experimented. One is combining multiple classifiers which use different classification algorithms from each other, and the other one is combining multiple classifiers which use the same classification algorithm. The output of a combined classifier for some input is usually defined as the linear combination of outputs of multiple classifiers to be combined [4].

The overall block diagram of generating and combining procedure when each single classifier uses a different classification algorithm is shown in Figure 5.1. First, a nonlinear filter is applied to the input image to reduce noise and to have more homogeneous regions. From the filtered image, training and testing samples are selected referring to the ground reference data. Ground data is more professionally generated by ground observation in order to properly interpret remotely sensed images. Using ground reference data, we can know the classes of a limited number of pixels of the remotely sensed image. Since the target data is already known, optimal weights can be calculated using least squares estimation during training. Each classifier in Fig 5.1 uses a different algorithm for classification. Statistical methods and neural network approaches explained

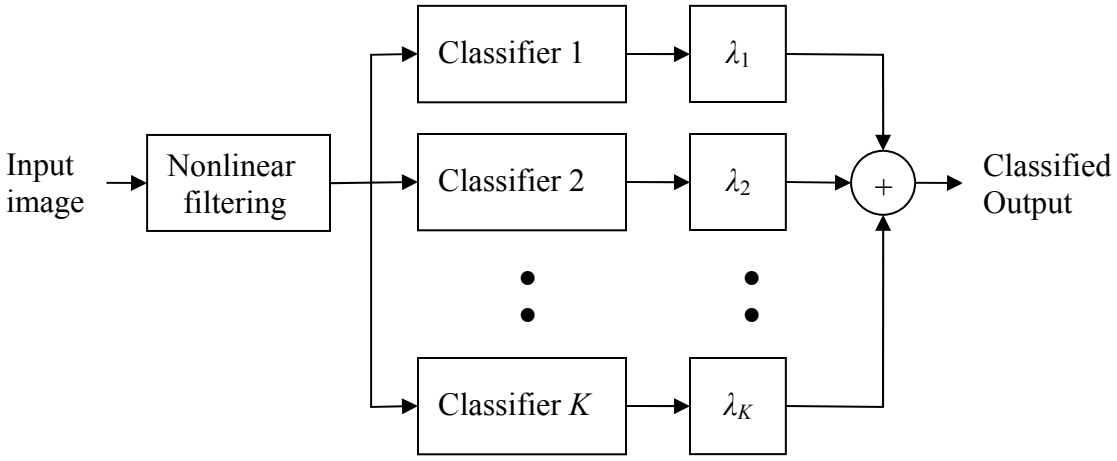


Figure 5.1 Block diagram of combining multiple different classifiers.

in Chapters 2 and 3 are used for classification. The classification results of the classifiers are combined with the weight vector $\boldsymbol{\lambda}=[\lambda_1\ \lambda_2 \ldots \lambda_K\ ]^T$ to produce the consensual results. To combine the results of the multiple classifiers, weight selection is needed. Weight reflects the goodness of each single classification result [3]. For example, a classification result with higher accuracy, can be given relatively higher weight. There are two methods for weight selection. One is giving equal weights for all single classification results, and the other method is giving optimal weights depending on the reliability of each single classification result.

Obtaining the optimal weight can be done by the delta rule based on least squares. From the results of multiple classifiers, we know the training and testing outputs. We also have target outputs which are the classes labeled to the training data. Suppose the trained results of single classifiers are represented as $\boldsymbol{X}=[\boldsymbol{X}_1\ \boldsymbol{X}_2 \ldots\ \boldsymbol{X}_K]$ and the desired output is $\boldsymbol{D}$. $\boldsymbol{X_i}$ is column vector containing the output of a single classifier. $\boldsymbol{X}$ is $l \times K$ matrix, where l is the number of training vectors and K is the number of single classifiers to be combined. Then, we can find the optimal weight by solving the following equation.

$$\boldsymbol{X\lambda}=\boldsymbol{D} \tag{5.6}$$

Optimal weights $\boldsymbol{\lambda}=[\ \lambda_1 \lambda_2 \ \ldots \lambda_K]^T$ are obtained when the square error is minimized as follows:

$$\boldsymbol{\lambda}_{\text{opt}} = \min_{\lambda} \left\| \boldsymbol{X\lambda} - \boldsymbol{D} \right\|^2 \tag{5.7}$$

Using pseudo inverse of $\boldsymbol{X}$, $\boldsymbol{\lambda}_{\text{opt}}$ is calculated by

$$\boldsymbol{\lambda}_{\text{opt}}=(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{D} \tag{5.8}$$

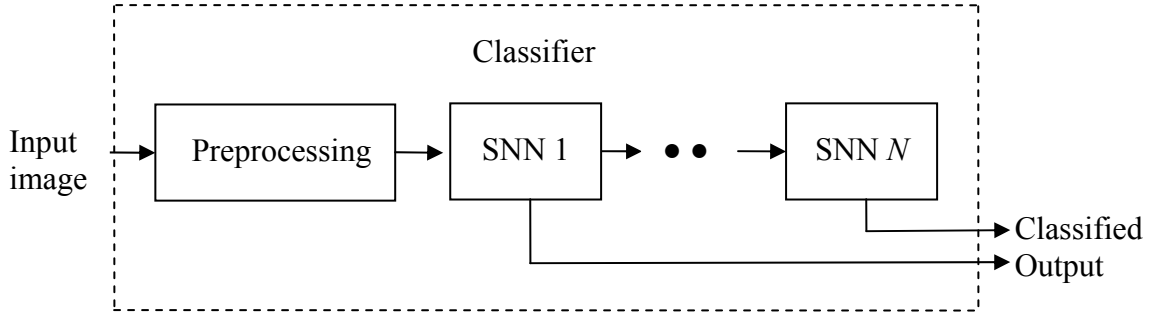
where $\boldsymbol{X}^T$ is the transpose of $\boldsymbol{X}$ and $(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$ is the pseudo inverse of $\boldsymbol{X}$. $\boldsymbol{\lambda}_{\text{opt}}$ is the optimal weight vector. Once the optimal weighs are obtained, consensual classification results can be obtained by applying a maximum rule. For example, majority rule can be used for final classification.

Multiple classifiers which use the same classification algorithm can be designed. By varying training data or other parameters used in training, multiple classification results can be produced using single classification algorithms. For example, hierarchical

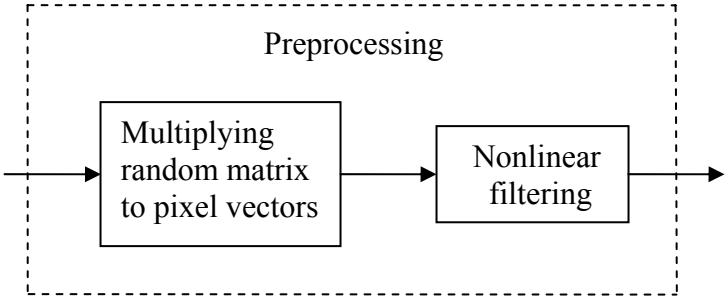
competitive learning, hierarchical SOM, and, hierarchical SOGR which can be used in classification generate various classification results by varying tuning parameters.

In this thesis, the following technique is used. First, random matrix per classifier with elements uniformly distributed between -1 and 1 is used to transform pixel vectors of input images. Then nonlinear filtering is applied to each channel of image. This preprocessing of input is done for each single classifier with rejection schemes.

To add more randomness, whenever each single classifier is trained, weight vectors are selected randomly out of training data. Both procedures are aimed at making each single classifier produce independent errors. Even though a single classification algorithm is used, each single classifier makes different classification errors since its parameters generated by training are different due to the two procedures used. The block diagram for combining multiple classifiers which use the same classification algorithm is shown in Figure 5.2.

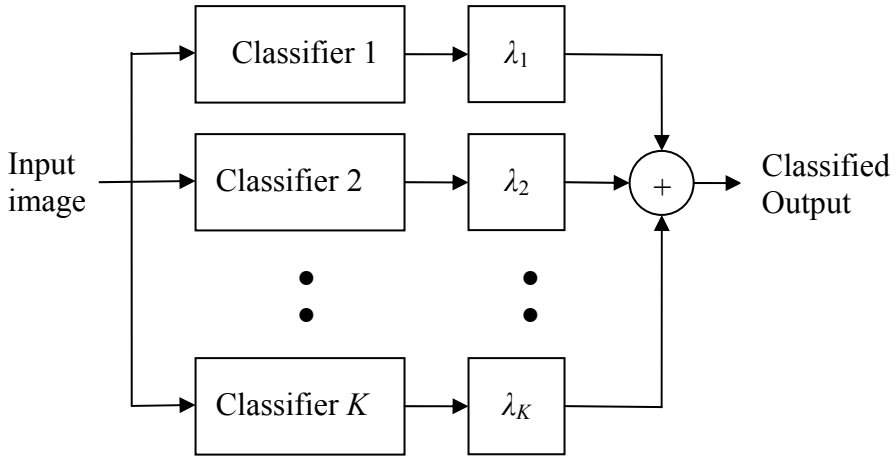


(a) Structure of a single classifier to produce a member of the ensemble.



(b) Preprocessing block in (a)

Figure 5.2 Block diagram of combining multiple same classifiers.



(c) Block diagram of combining multiple same classifiers

Figure 5.2 Block diagram of combining multiple same classifiers (continued).

5.3 Diversity Measures in Combining Multiple Classifiers

Classifiers in the ensemble should be as accurate as possible and should not make coincident errors to obtain improved classification result with consensual classification. Thus, single classifiers to be combined should make independent errors. If a classifier makes errors, it can be complemented with another classifier which makes errors differently. Disagreement of errors is highly beneficial for improvement of performance in consensual classification. Diversity measures can be helpful in designing the individual classifiers and for combining them [37], [94].

Diversity measure is meaningful if it is applied to a group of classifiers. In the simplest case, a measure can be applied for examining diversity between two classifiers. Such measures are referred to as pairwise diversity measures. For more than two classifiers, pairwise diversity measure is typically obtained by averaging the pairwise diversity measures calculated for all pairs of classifiers from the considered pool of classifiers.

Let $\mathbf{Z}=\{\mathbf{z}_1, \mathbf{z}_2,\ldots, \mathbf{z}_N\}$ be a labeled data set and $D=\{D_1, D_2,\ldots, D_K\}$ be a pool of classifiers. Each classifier labels every data point $\mathbf{z}_j$ with a class label from $\Omega=\{\omega_1, \omega_2,\ldots, \omega_M\}$. The output of classifier D_i for all the input vectors can be represented as a binary vector $\boldsymbol{y}_i=[y_{1,i}, y_{2,i}, \ldots, y_{N,i}]^T$ such that $y_{j,i}=1$ if D_i recognizes correctly $\mathbf{z}_j$, and 0,

otherwise, where i=1, …, K. Various statistics are used to asses the similarity of two classifier outputs.

The Q statistics for two classifiers, D_i and D_k, is defined as

$$Q_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}} \tag{5.10}$$

where N^{ab} is the number of elements $\mathbf{z}_j$ of $\mathbf{Z}$ for which $y_{j,i}=a$ and $y_{j,k}= b$. These relationships between the two classifiers are shown in Table 5.1

Table 5.1 The 2×2 table of the relationship between two classifiers.

	D_k correct (1)	D_k wrong (0)
D_i correct (1)	N^{11}	N^{10}
D_i wrong (0)	N^{01}	N^{00}

When the classifiers make the same correct and incorrect decisions, it can be seen that the value of the Q statistic becomes one. Negative values indicate classifiers that make errors on different inputs. Q varies between - 1 and 1. Classifiers that tend to recognize the same objects correctly will have positive values of Q, and those that commit errors on different objects will render Q negative [94]. For sets of more than two classifiers the mean value of the pairwise Q statistics is considered to be the Q value for that set. The best subset of member classifiers is thus selected by minimizing the value of the Q statistic. When there are more than two classifiers in a combination pool, average Q statistics is defined as

$$Q_{avg} = \frac{2}{K(K-1)} \sum_{i=1}^{K-1} \sum_{k=i+1}^{K} Q_{i,k} \tag{5.11}$$

The correlation between two binary classifier outputs (correct/incorrect), $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ is defined as

$$\rho_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{\sqrt{(N^{11} + N^{10})(N^{01} + N^{00})(N^{11} + N^{01})(N^{10} + N^{00})}} \tag{5.12}$$

For any two classifiers, Q and ρ have the same sign, and it can be proved that $|\rho| \leq |Q|$.

The disagreement measure is used to characterize the diversity between a base classifier and a complementary classifier. It is the ratio between the number of observations on which one classifier is correct and the other one is incorrect to the total number of observations. This measure is defined as in (5.13).

$$S_{i,k} = \frac{N^{01} + N^{10}}{N^{11}N^{00} + N^{01}N^{10}} \tag{5.13}$$

6. EXPERIMENTAL RESULTS

Experiments were conducted with the consensual and hierarchical classification approaches on multispectral images. First, combining multiple classifiers which use different classification algorithms was investigated. Second, combining multiple classifiers which use the same classification algorithm was investigated. In the second approach, input data and learning parameters are varied by preprocessing to make independent errors. The results of the experiments are discussed below.

6.1 Combining Multiple Different Classifiers

Fifteen results of multiple different classifiers are used for combining. Maximum likelihood, ECHO, hierarchical competitive learning, hierarchical SOM, and hierarchical SOGR are used as single classifiers. ECHO was performed using Multispec [21]. For each single classifier, three input images are used. Three different input images were generated by filtering with median filter, generalized morpholofical filter (GMF), and spatial filter.

6.1.1 Experiments with West Lafayette image

This data set is a multispectral image from Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) built by Jet Propulsion Laboratory (JPL) and flown by NASA/Ames on June 12, 1992 [21]. The scene is over an area 6 miles west of West Lafayette. The scene is a subset of a significantly larger image file. Originally, this image has 220 channels. Image of 9 channels, which is reconstructed by selecting 9 channels that represent different regions of spectrum out of 220 channels, was used in the experiments. This image has 17 classes (background, alfalfa, corn-notill, corn-min, corn, grass/ pasture, grass/trees, grass/pasture-mowed, hay-windrowed, oats, soybean-notill, soybean-min, soybean-cleas, wheat, woods, bldg-grass-tree-drives, stone-steel towers). It is a 16 bit pixel image with the size of 145×145. Each pixel in one channel represents one of 65536 gray levels. 3403 and 8495 pixels out of the filtered image were selected referring to ground reference data for training and testing fields, respectively. These

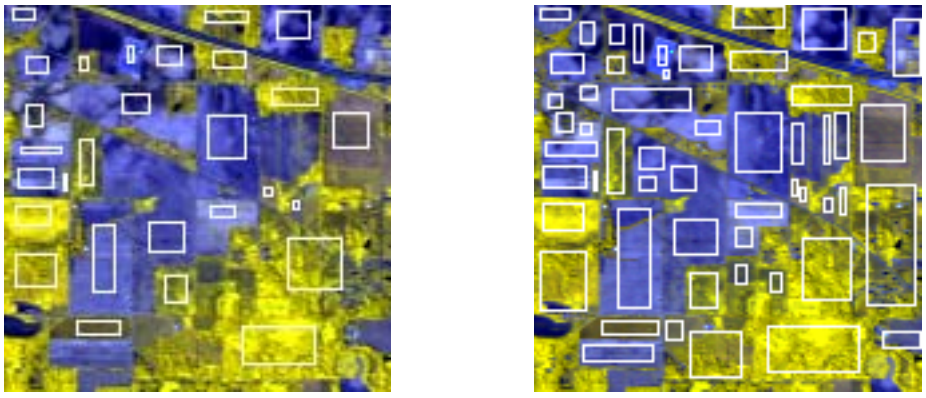
training and testing data were selected from pixels belonging to all classes. These are 16 % and 40 % of the number of total pixels in the image. Selected training and testing fields of West Lafayette image are shown in Figure 6.1. Channel description of this image is given in Table 6.1. Channel 3, 4, and 5 are selected to display the input image in Fig 6.1.

For the classifiers using neural networks, which are competitive learning, SOM, and, SOGR, hierarchical approach with rejection scheme ***RADPN*** was used. (6.1) is used for learning rate of competitive learning and SOM:

$$0.1 \times \left\{ 1 - \frac{k}{(\text{number of iterations}) \times N_i^j} \right\} \tag{6.1}$$

Table 6.1 Channel description of West Lafayette image.

Channel	Spectral Band (μm)
1	0.475-0.485
2	0.554-0.564
3	0.663-0.673
4	0.751-0.761
5	0.828-0.838
6	1.050-1.060
7	1.204-1.214
8	1.655-1.755
9	2.213-2.223



(a) Training fields (b) Testing fields

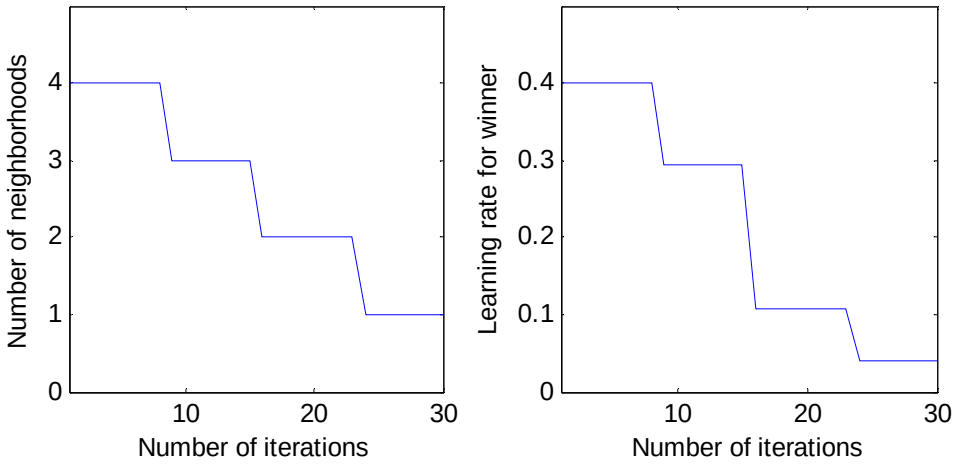
Figure 6.1 Training and testing fields in the West Lafayette image.

In (6.1), k is the current iteration number and N_i^j is the number of training vectors belonging to the jth class in the ith SNN. Learning is executed for 30 iterations for all methods using neural network algorithms.

One dimensional SOM is used for hierarchical SOM with rejection scheme. The size of neighborhood for the winning weight vector was set to 5 initially and neighborhood shrinks to 3 and 1 as the number of iterations increases.

Size of neighborhoods and learning rate for SOGR are shown in Figure 6.2. Like SOM, neighborhood of winning vector shrinks as the number of iterations increases. Figure 6.2 (b) represents the learning rate for winner. Learning rate for neighborhood of winner decreases to 3/4, 1/2, and 1/4 of it, respectively depending on the distance with winner.

When SOM is used for learning algorithm and the image filtered by GMF is classified, 0 training vector and 4113 test vectors were rejected from SNN 3 and classified to the class of the closest weight vector. Training and testing accuracy was 100% and 77.63%, respectively. Number of trained and tested data in each SNN is given in Table 6.2. 91.89 % of training data and 48.25% of testing data were classified in SNN 1. As the number of SNN increases, the number of training and testing data decreases. To evaluate hierarchical approach applied to competitive learning, SOM, and SOGR, experiments were also done without hierarchical approach with generalized morphological filtered image. Comparison of two results with and without hierarchical approach using rejection schemes is given in Table 6.3. When hierarchical approach with



(a) Size of neighborhood (b) Learning rate for winner

Figure 6.2 Number of neighborhoods and learning rate for SOGR.

Table 6.2 Number of trained and tested data in each SNN for hierarchical SOM with rejection scheme applied to West Lafayette image.

Stage	Number of trained data	Number of tested data
SNN 1	3127	4099
SNN 2	271	278
SNN 3	5	5
Rejected data	0	4113
Total	3403	8495

Table 6.3 Classification performance of hierarchical approach applied to West Lafayette image.

Classification methods	Training accuracy (%)	Testing accuracy (%)
Competitive learning	88.33	62.14
Hierarchical competitive learning	100.00	76.99
SOM	90.21	64.81
Hierarchical SOM	100.00	77.63
SOGR	81.93	60.42
Hierarchical SOGR	100.00	80.06

Table 6.4 Class separability of West Lafayette image.

Image type	Transformed divergence		J-M distance	
	Training data	Testing data	Training data	Testing data
Original image	1986.68	1979.02	1.963	1.924
Median filtered	1992.66	1987.27	1.981	1.963
GMFed	1996.58	1994.25	1.989	1.963
Spatial filtered	1996.65	1993.83	1.988	1.960

rejection schemes is used, classification accuracy is improved as shown in Table 6.3. As shown in Table 6.4, class separability for training and test data increases by nonlinear filtering.

Since three different input images, which are filtered by median filter, GMF, and spatial filter, are classified using five different classification algorithms, which are maximum likelihood, ECHO, hierarchical competitive learning, hierarchical SOM, and hierarchical SOGR, there are fifteen classification results to be combined. Using delta rule discussed in Chapter 5, optimal weights are calculated for combination. Optimal

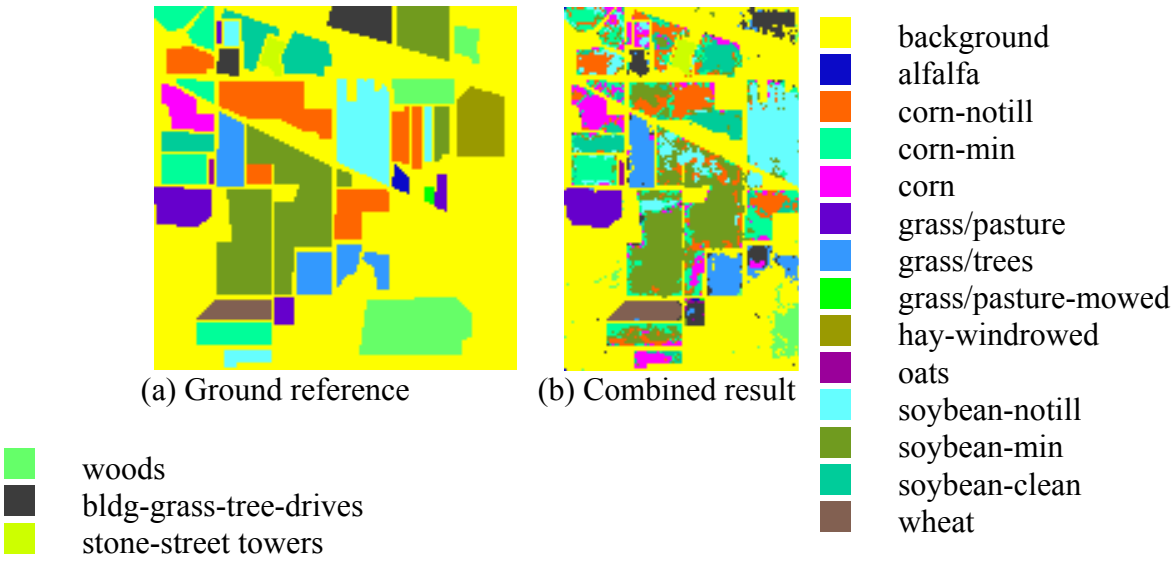
weights represent the goodness of each classification results. These are used in combination of test results.

Complete experimental results are given in Table 6.5. Test accuracy of hierarchical SOGR with spatial filtered image is 80.67 %. This is the best accuracy among those of single classifiers. By combining with consensus rule, the testing accuracy is improved to 85.82 %. In hierarchical competitive learning, hierarchical SOM, and hierarchical SOGR, three SNNs are used and the first 20 training vectors belonging to each class are selected as weight vectors.

The ground reference thematic map and thematic maps generated by all single classifiers and consensual classifiers are shown in Figure 6.3 to Fig 6.8. It is observed that the thematic map of consensual classification is more similar to the ground reference thematic map than the thematic maps obtained by any other single classifier.

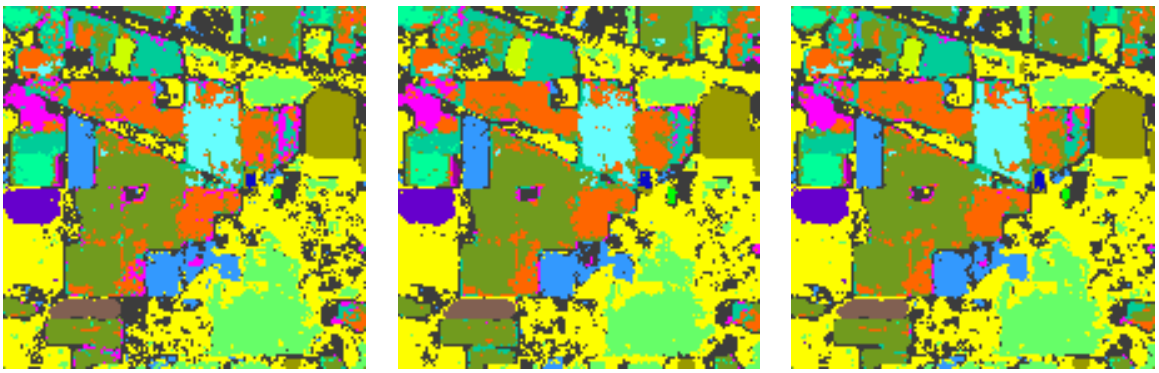
Table 6.5 Experimental results with the West Lafayette image.

Classification Methods	Filter	Training accuracy (%)	Testing accuracy (%)
Maximum likelihood	Median	90.33	72.95
	GMF	93.33	75.59
	Spatial	91.98	75.59
ECHO	Median	94.68	77.27
	GMF	96.03	79.34
	Spatial	95.39	79.21
Hierarchical Competitive learning	Median	100.00	75.15
	GMF	100.00	76.99
	Spatial	100.00	73.29
Hierarchical SOM	Median	100.00	77.83
	GMF	100.00	77.63
	Spatial	100.00	72.25
Hierarchical SOGR	Median	100.00	78.14
	GMF	100.00	80.06
	Spatial	100.00	80.67
Combined by consensus rule		100.00	85.82



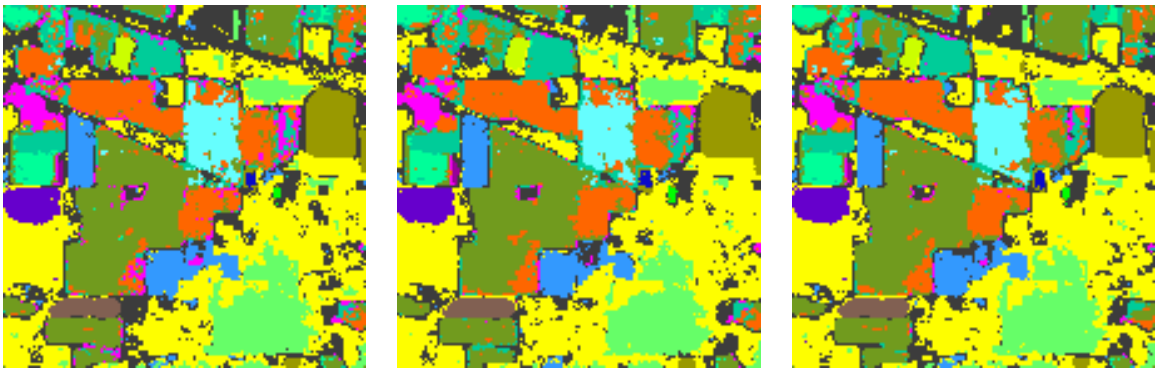
(a) Ground reference (b) Combined result

Figure 6.3. Thematic maps generated by consensual classification with West Lafayette image.



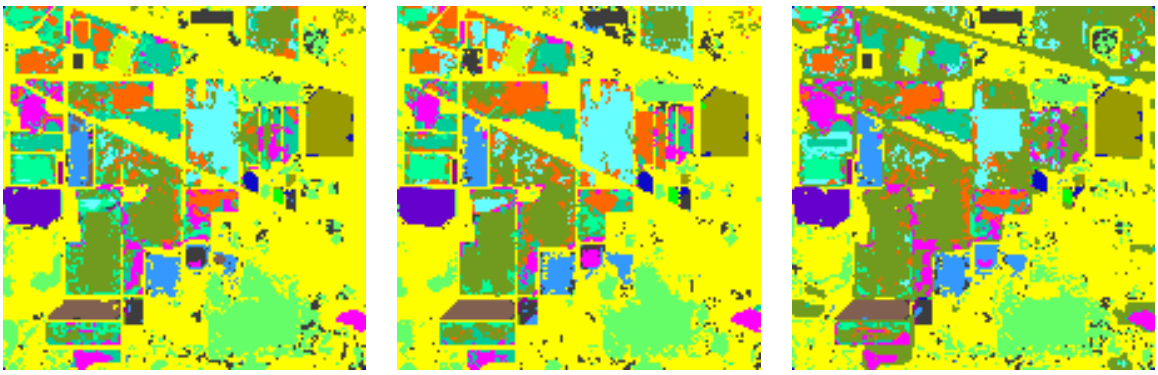
(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.4 Thematic maps generated by maximum likelihood classification with West Lafayette image.



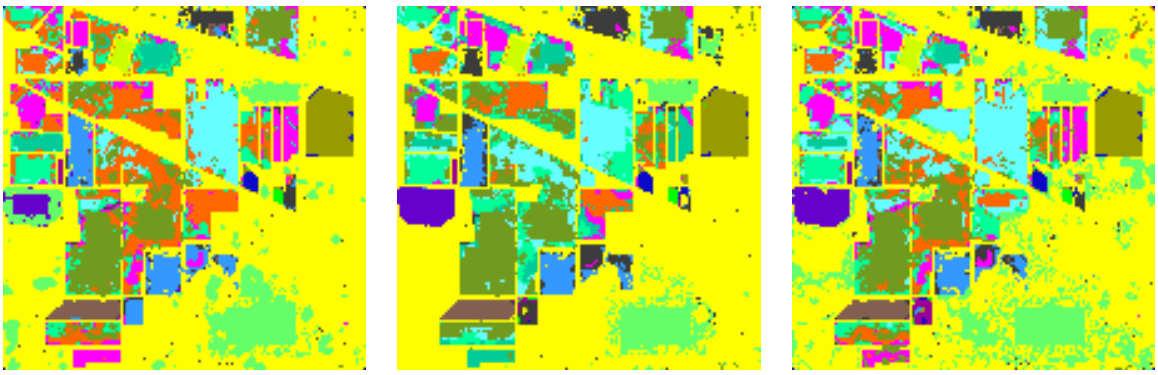
(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.5 Thematic maps generated by ECHO with West Lafayette image.



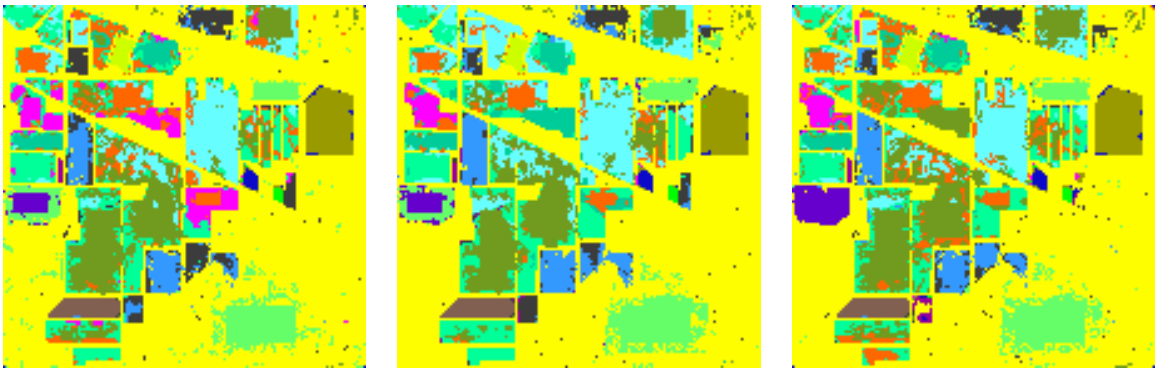
(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.6 Thematic maps generated by hierarchical competitive learning with West Lafayette image.



(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.7 Thematic maps generated by hierarchical SOM with West Lafayette image.



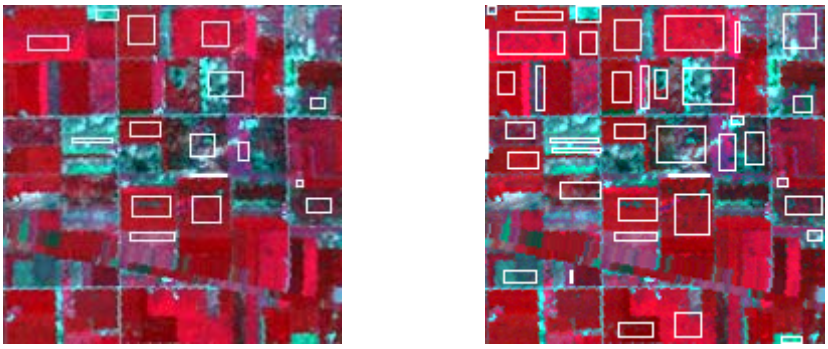
(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.8 Thematic maps generated by hierarchical SOGR with West Lafayette image.

6.1.2. Experiments with Tippecanoe County image

This is a small segment (169×169) of a Thematic Mapper scene of Tippecanoe County, Indiana gathered on July 17, 1986 [21]. Each pixel in this image consists of 8 bits per channel. There are 6 classes (background, corn, soybean, wheat, alfalfa/oats, pasture) in this image. There is a wrongly measured part in this image. This part is labeled as sensor distortion. Sensor distortion refers to a part of the image where the pixels do not line up correctly. The pixels were out of sync when originally recorded when being downloaded from the Landsat sensor because the sensor lost sync signals for those lines. Thus, this part is not used in generating thematic map and masked out on thematic map. The Tippecanoe County image is a 7 channel data and channel description is given in Table 6.6.

2024 and 6346 pixels were selected for training and testing fields, respectively, by referring to the ground reference data. These are 7 % and 22 % of the total numbers of pixels in the image. Selected training and testing fields of the Tippecanoe County image



(a) Training field (b) Testing field

Figure 6.9 Training and testing fields in the Tippecanoe County image.

Table 6.6 Channel description of Tippecanoe County image.

Channel	Spectral Band (μm)
1	0.45-0.52
2	0.52-0.60
3	0.63-0.69
4	0.76-0.90
5	1.55-1.75
6	2.08-2.35
7	10.4-12.5

are shown in Figure 6.9. Channels 2, 3, and 4 are selected to display the input image in Fig 6.9. As in the experiments with the West Lafayette image, three nonlinear filters were applied for all classifiers prior to classification. Improvement of class separability by nonlinear filtering is given in Table 6.7.

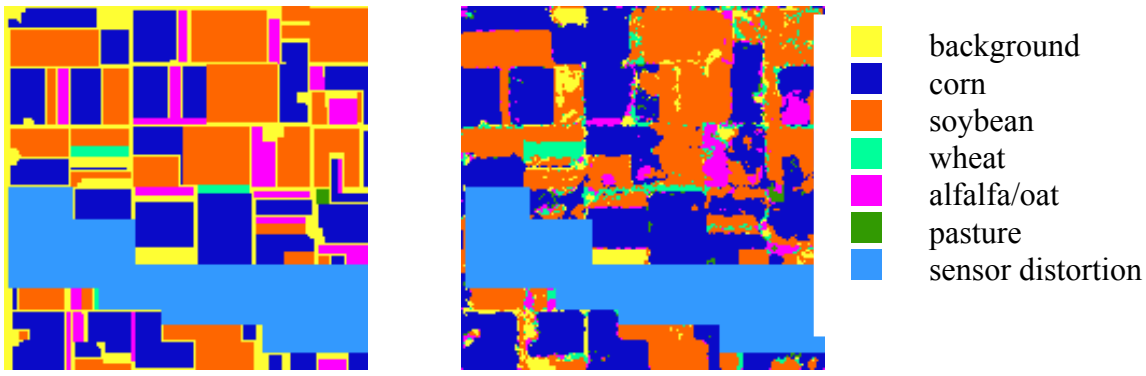
Only one SNN is used for hierarchical competitive learning, hierarchical SOM, and hierarchical SOGR and all training vectors belonging to each class are used as weight vectors. Since few training data belonging to one class are rejected from SNN, the number of SNN was set to one and rejected data were classified to the class of nearest weight vectors. Experimental results with consensual classifiers and single classifiers are given in Table 6.8. The ground reference thematic map and thematic maps generated by all single classifiers and consensual classifiers are shown in Figure 6.10 to Figure 6.15. When median filtered Tippecanoe County image is classified by hierarchical SOM, testing accuracy was 82.73 %. This is the best result among those of single classifiers. By combining fifteen classification results, the testing accuracy increased to 83.00 %. This is higher than the result of any other single classifier.

Table 6.7 Class seperability of Tippecanoe County image.

Image type	Transformed divergence		J-M distance	
	Training data	Testing data	Training data	Testing data
Original image	1989.68	1966.02	1.925	1.712
Median filtered	1996.48	1986.40	1.960	1.799
GMFed	1998.22	1993.08	1.968	1.828
Spatial filtered	1998.29	1992.98	1.964	1.826

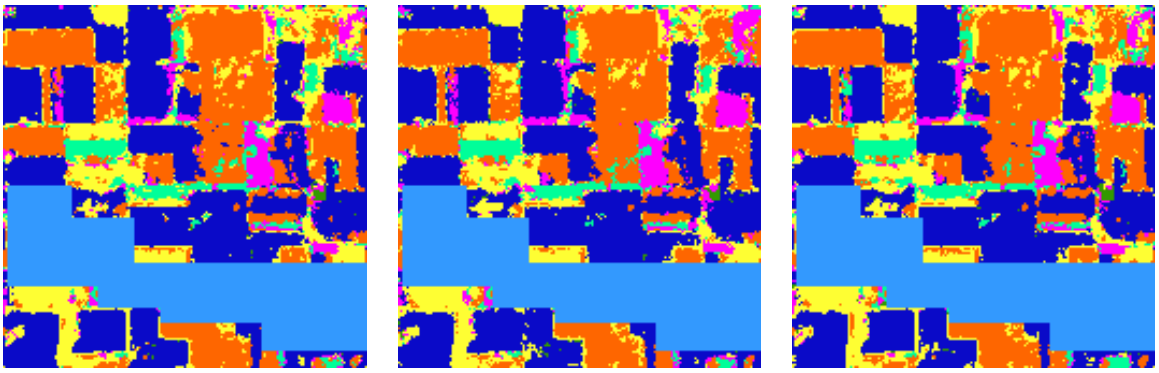
Table 6.8 Experimental results with the Tippecanoe County image.

Classification Methods	Filter	Training accuracy (%)	Testing accuracy (%)
Maximum likelihood	Median	95.21	76.99
	GMF	96.99	78.24
	Spatial	95.31	77.59
ECHO	Median	97.08	78.06
	GMF	98.27	78.96
	Spatial	97.63	78.82
Hierarchical Competitive learning	Median	100.00	82.00
	GMF	100.00	81.48
	Spatial	100.00	81.15
Hierarchical SOM	Median	100.00	82.73
	GMF	100.00	80.51
	Spatial	100.00	82.70
Hierarchical SOGR	Median	100.00	81.97
	GMF	100.00	81.61
	Spatial	100.00	81.12
Combined by consensus rule		100.00	83.00



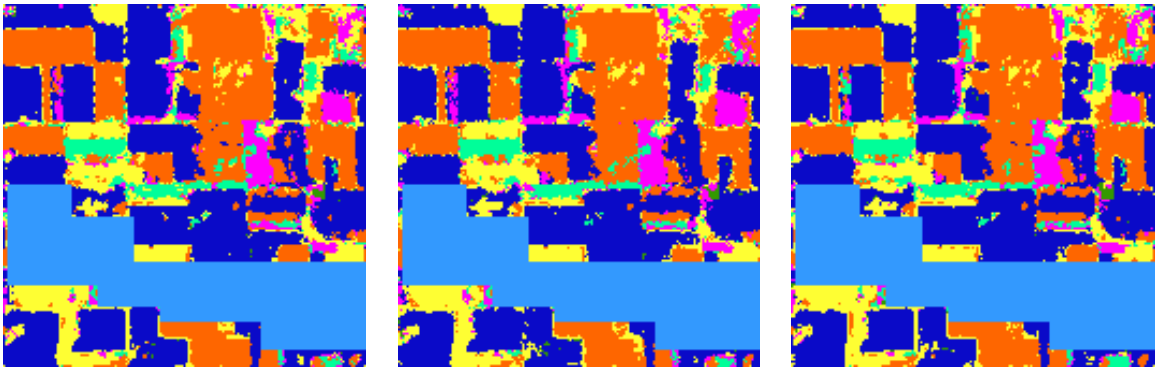
(a) Ground reference (b) Combined result

Figure 6.10. Thematic maps generated by consensual classification with Tippecanoe County image.



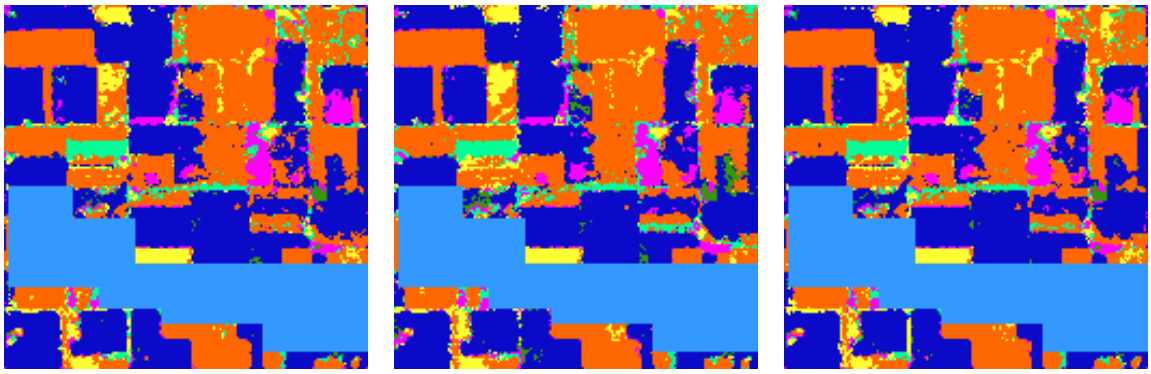
(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.11 Thematic maps generated by maximum likelihood classification with Tippecanoe County image.



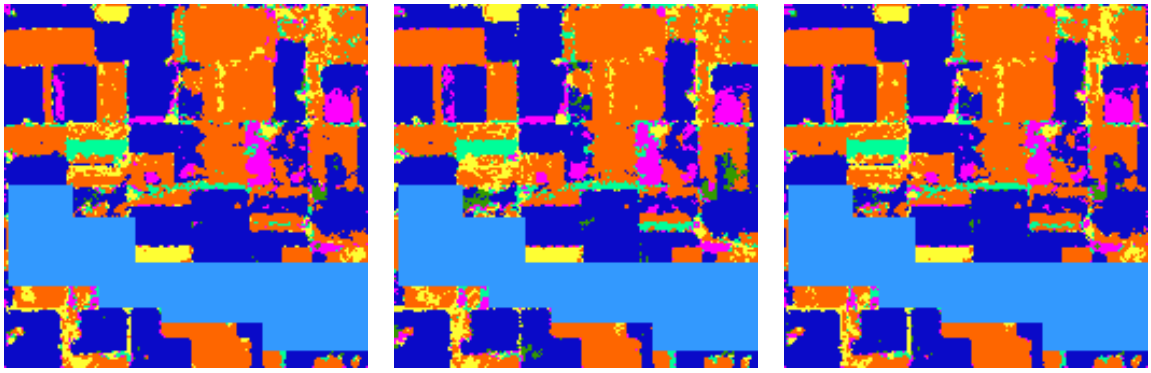
(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.12 Thematic maps generated by ECHO with Tippecanoe County image.



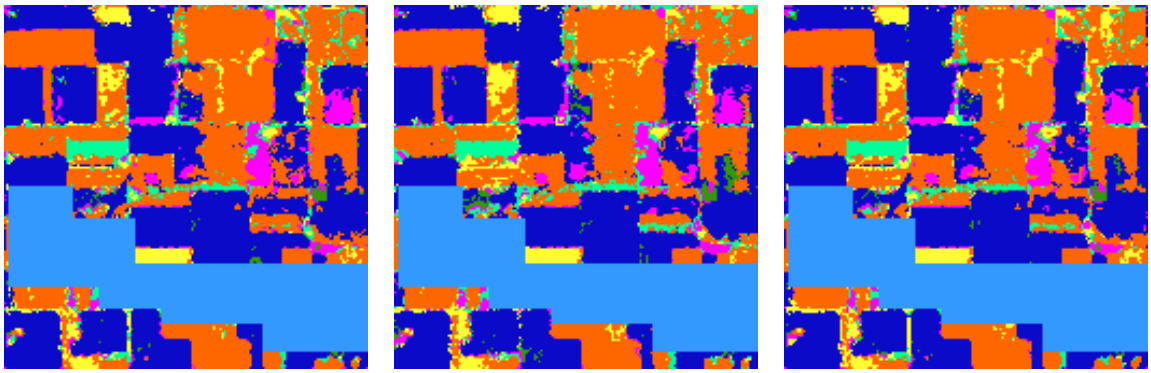
(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.13 Thematic maps generated by hierarchical competitive learning with Tippecanoe County image.



(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.14 Thematic maps generated by hierarchical SOM with Tippecanoe County image.



(a) Median filtered (b) GMFed (c) Spatial filtered

Figure 6.15 Thematic maps generated by hierarchical SOGR with Tippecanoe County image.

6.1.3. Experiments with Colorado data set

Colorado data set is a multisource data set. It is a remote sensing data covering a mountainous area in Colorado [6]. The Colorado data set consists of the following four data sources: Landsat MSS data (4 data channels), elevation data (in 10 m contour intervals, 1 data channel), slope data (0-90 degrees in degree increments, 1 data channel), aspect data (1-180 degrees in 1 degree increments, 1 data channel). It has ten land cover classes (water, Colorado blue spruce, Montane/Subalpine meadow, Aspen, Ponderosa pine, Ponderosa pine/Douglas fir, Engelmann Spruce, Douglas fir/White fir, Douglas fir/Ponderosa pine/Aspen, Douglas fir/White fir/Aspen). All classes are forest types except water. Since the forest classes show very similar spectral response, it is difficult to distinguish among the forest types using the Landsat MSS data only [95]. So other geographical sources such as slope and elevation data are used with the Landsat MSS

data. Colorado data set has seven channels totally. 1188 training samples and 831 testing samples were selected from the image of 135 rows and 131 columns for each channel referring to ground reference data.

Classification was performed with four different methods, which are maximum likelikood classification, hierarchical competitive learning, hierarchical SOM, and hierarchical SOGR. Maximum likelikood classification was done using only 4 channel data, which are image data. If all multisource data are used, determinant and inverse of covariance matrix in (2.9) can not be computed since the covariance matrix is singular. The four classification results were combined using delta rule. Experimental results with the Colorado data set is given in Table 6.9.

Table 6.9 Experimental results with the Colorado data set

Classification methods	Training accuracy (%)	Testing accuracy (%)
Maximum likelihood	74.92	49.58
Hierarchical competitive learning	98.06	58.12
Hierarchical SOM	97.90	58.12
Hierarchical SOGR	97.56	56.68
Combined by consensus rule	97.90	60.41

6.2 Combining Multiple Classifiers Generated with the Same Classification Algorithm

To investigate the effectiveness of combining multiple classifiers based on a single classification algorithm, hierarchical competitive learning, hierarchical SOM, and hierarchical SOGR were used as the single classification algorithm in each experiment to make multiple classifiers classify West Lafayette image. To inject randomness into the learning algorithm, random matrix whose elements are uniformly distributed between -1 and 1 was used to transform each pixel vector of the input image and the transformed image was filtered using median filter prior to classification. This preprocessed input image was classified by hierarchical competitive learning. This classification was performed 30 times. Each time, a different random matrix was used to transform the pixel vectors of input image. By hypothesis, this is equivalent to 30 different classifiers being used for consensus.

For hierarchical SOM and hierarchical SOGR, the same approach was applied. By combining the classification results of individual classifiers, combined testing accuracy

was observed to increase as the number of classifiers to be combined increases until saturation occurs as in shown Figure 6.16 to Figure 6.18. Figure 6.16 shows testing accuracies of various classification results generated by the same classifier using hierarchical competitive learning which was trained in different ways and testing accuracies of combined results. Even poor result is helpful in increasing the testing accuracy of combined result since it could offer complementary information when multiple results are combined. Thus, the test accuracy of combined result is improved as shown in Figure 6.16. When hierarchical SOM and hierarchical SOGR are used for classification, testing accuracy is improved if the multiple classification results generated by preprocessing and varying learning parameters are combined as in Figure 6.17 and Figure 6.18. Thematic maps of combined multiple classifiers of single classification algorithm are shown in Figure 6.19. Even though the same classification algorithm is used to generate multiple classification results, thematic maps look similar with the those of combined multiple different classifiers.

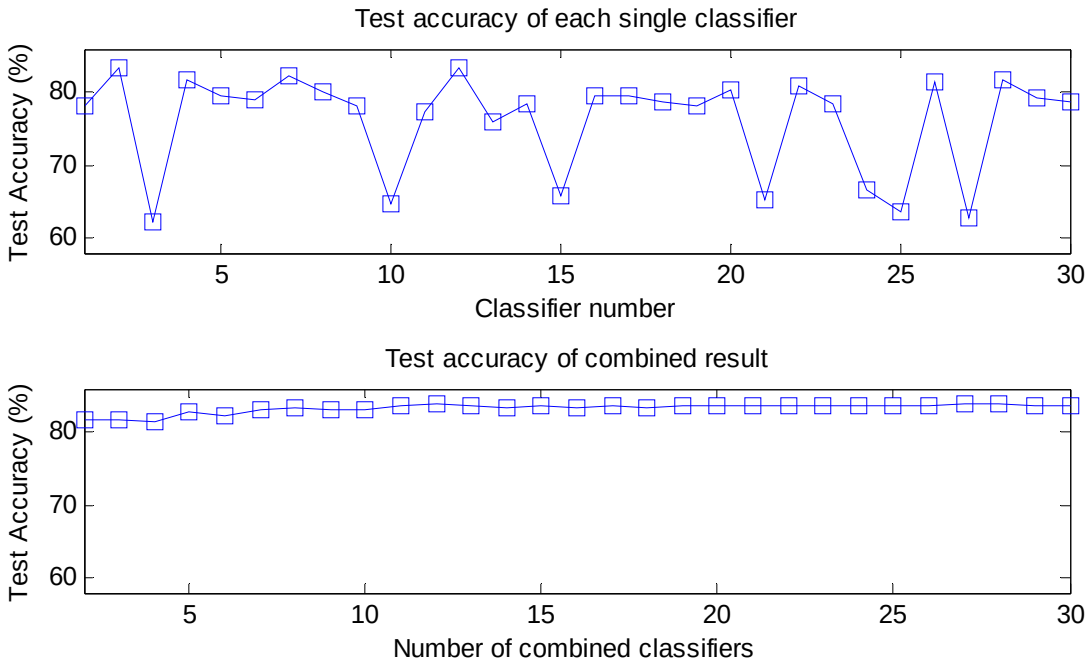


Figure 6.16. Test results with hierarchical competitive learning.

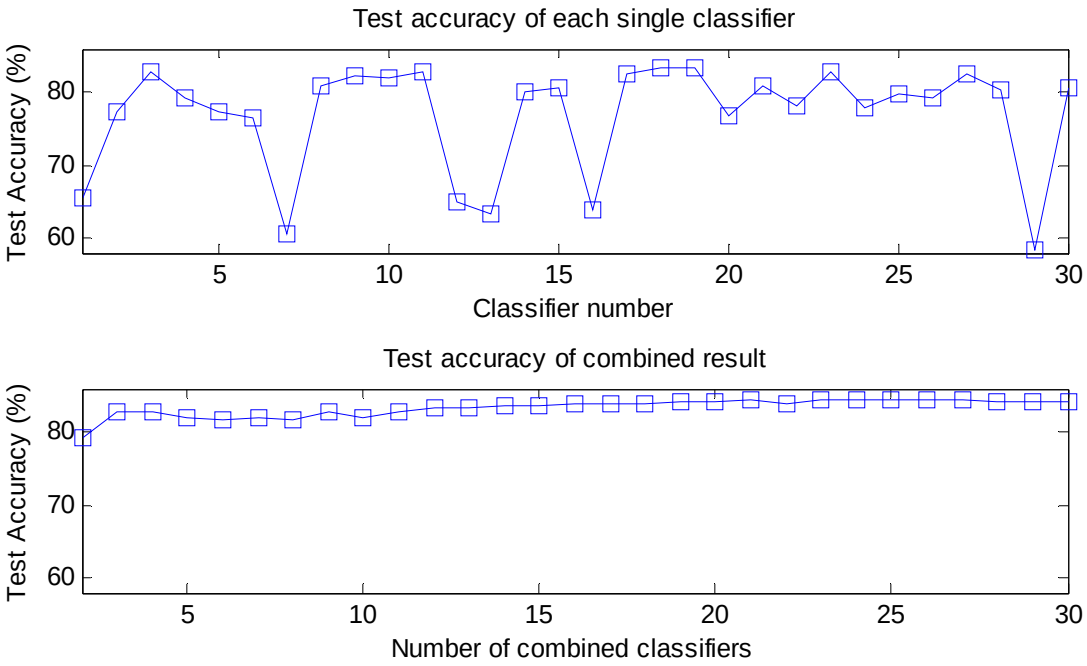


Figure 6.17. Test results with of hierarchical SOM.

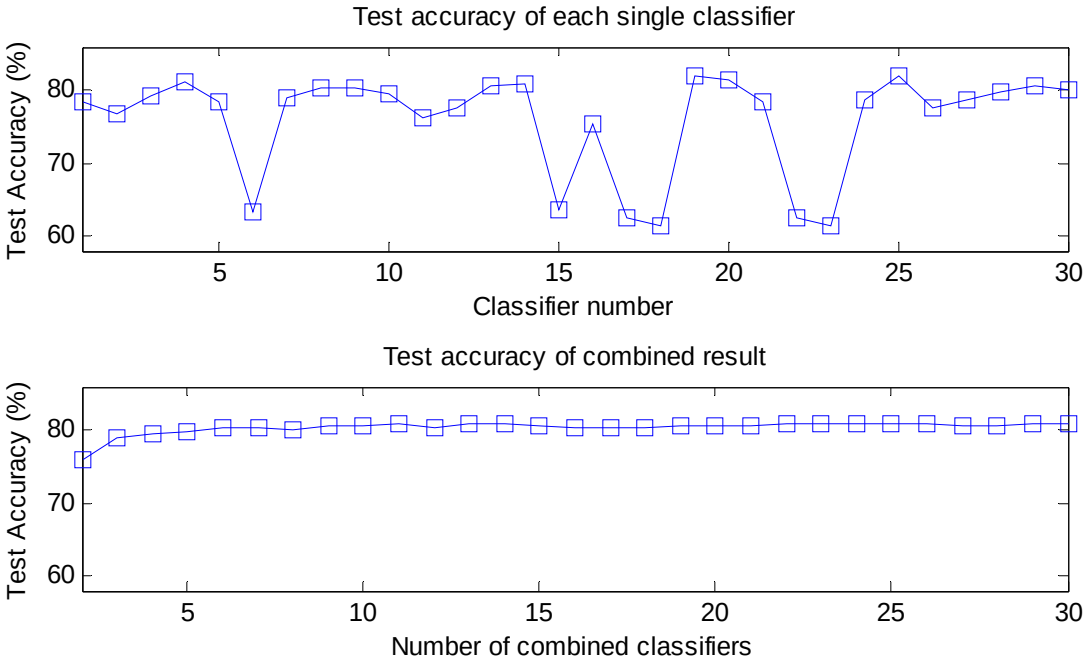
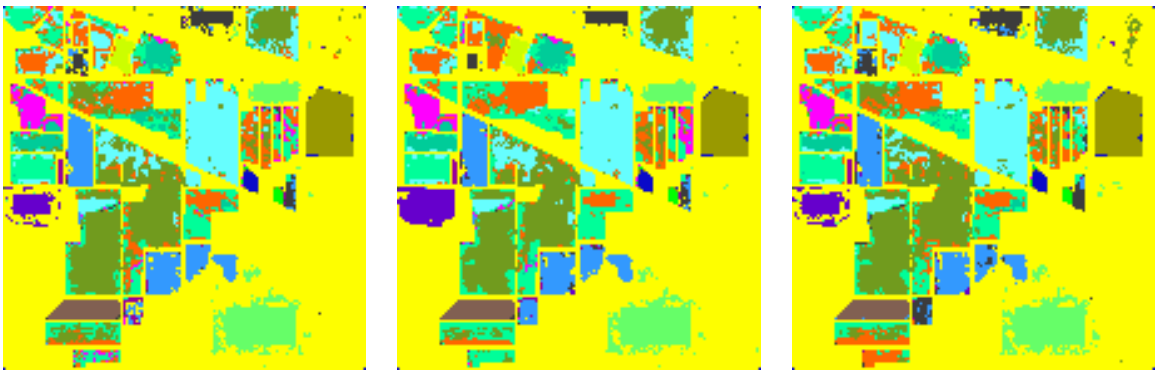


Figure 6.18. Test results with hierarchical SOGR.



(a)Hierarchical competitive learning (b)Hierarchical SOM learning (c)Hierarchical SOGR

Figure 6.19 Thematic maps generated by combining multiple classifiers generated with the same classification algorithm with the West Lafayette image.

7. CONCLUSIONS AND FUTURE WORK

Classification performance is improved by applying hierarchical strategy in neural network classifiers and combining the classification results of multiple classifiers by consensus rule. Nonlinear filtering is applied to reduce variance of homogeneous regions and improves spectral separability. Median filter, generalized morphological filter, and spatial filter are used for this purpose. Improvement of class separability is shown by computing class separability measures, which are transformed divergence and J-M distance. Maximum likelihood, ECHO, hierarchical competitive learning, hierarchical SOM, and hierarchical SOGR are used for classification algorithms of single classifiers. By using different classifiers, classification errors caused by single classifiers are sufficiently independent. This allows improvement of overall classification accuracy when the multiple classification results are combined. Delta rule is used for combining the results of multiple classifiers. By delta rule, optimal weight based on the goodness of the result of each single classifier are generated, and the classification accuracy of the combined result is better than that of any other single classifier.

Hierarchical approach using rejection schemes are applied to competitive learning, SOM, and SOGR. This approach improves classification performance by successive classifiers which are tuned to reduce remaining error. Hard vectors which are difficult to classify are detected before classification and processed differently from non-hard vectors. This results in reducing the misclassification rate. For this purpose, rejection schemes are constructed. The input vectors inside rejection boundary are classified in current SNN, and the input vectors outside rejection boundary are rejected from the current SNN and fed into the next SNN for further processing. Rejection boundaries which are constructed during training are stored to use in testing. Testing is performed using stored rejection boundaries. If there is no rejected training vector or the training accuracy of current SNN reaches 100 % or training accuracy of the network with the current SNN is less than that of the network without the current SNN, the training procedures is stopped. When training is stopped, if vectors to be classified are still remaining, those are classified to the class of the closest weight vector. Testing is done

similarly to training. Two multispectral images and one multisource remote sensing were used to test the proposed algorithm.

The improvement of classification performance by consensus is achieved when the errors produced by multiple classifiers are different and there is little correlation between them. That is, each classifier needs to make independent errors. When multiple classifiers using different types of classification algorithms are used to generate multiple classification results, the errors generated by different classifiers can be assumed to be sufficiently independent.

If only one classification algorithm is used, it is possible to make independent errors by varying input pattern and training parameters. Since the classification is done with different input pattern and training parameter, the classification result is varied whenever classification is performed. This has the similar effect with using multiple different types of classifiers which use different classification algorithms. This is simple and has similar performance as the case with multiple different classifiers.

In this research, optimal weights which are calculated by the delta rule are used for combination of the results of single classifiers. Combination of multiple results to produce a consensual result is based on the minimum squared error in this case. Development of new consensus rules can be a topic of further research. Consensus theory is based on that a group decision with multiple data sources is better in terms of mean square error than a decision from a single data source. Since no single method can be good for all kinds of multispectral images, if the classification results of single classifiers are combined by some consensus rule, classification performance can be improved. Similarly, since the complexity of class boundaries is not necessarily uniform over all classes in a feature space, a situation may develop such that one classification method works best for the data belonging to one class while another classification method works best for the data belonging to another class. In this case, best consensual classifier, which works best for each class, can be found. In this approach, different classifiers are used for each class depending on its performance for the data belonging to a certain class. Since this approach has similarities to data or image fusion techniques such as wavelet and transform approaches, and other techniques for fusion can also be investigated for best performance in the future.

There are other classification methods using other neural network algorithms such as support vector machines and backpropagation networks which are not covered in this thesis. Hierarchical approach using rejection scheme used in this thesis also can be applied to these classification methods for further research.

One major source of classification error in neural networks is the nonseparability of the classes. To reduce or eliminate classification errors, it is desirable to find a transformation which maps the input vectors into another set of vectors that can be classified more accurately. In this thesis multiplying with a uniformly distributed random matrix is used to vary the each pixel vector. Finding a more proper transformation would help to improve separability of classes and may result in further improvement of classification accuracy.

When competitive neural networks are used for classification, classification accuracy relies on how correctly the reference vectors are computed. However, it is difficult to compute the weight vectors which produce globally minimum errors because weight vectors depends on initial weight vectors, learning rate, the order of training samples, etc. Hierarchical approach using rejection schemes improves classification performance since hard vectors are detected before classification and processed separately in the next SNN. Further refinement of the hierarchical approach is a future research topic.

Using spectral variations as the primary source to derive information from the images is for avoiding the need for very high spatial resolution. If the high spatial resolution images are available, more accurate classification can be achieved. Thus image fusion is useful to obtain higher resolution images preserving spectral features [91] [92]. By injecting the high resolution of panchromatic image measured by panchromatic sensor into the multispectral band, multispectral image can have high resolution as the panchromatic image maintaining its spectral resolution. This image fusion combined with the approaches discussed in this thesis is another future work for improvement of classification performance.

REFERENCES

[1] T. M. Lillesand, *Remote Sensing and Image Interpretation*, 4th ed., John Wiley & Sons, New York, 2000.

[2] C. H. Chen, *Information Processing for Remote Sensing*, World Scientific, River Edge, N.J., 1999.

[3] J. A. Benediktsson, J.R. Sveinsson, O. K. Ersoy, and P.H. Swain, "Parallel consensual neural networks," *IEEE Trans*. *Neural Networks*, vol. 8, no.1, pp. 688-704, Jan. 1997.

[4] N. Ueda, "Optimal linear combination of neural networks for improving classification performance," *IEEE Trans*. *Pattern Analysis and Machine Intelligence*, vol. 22, no. 2, pp. 207-215, Feb. 2000.

[5] S. S. Haykins, *Neural Networks*: *A Comprehensive Foundation*, 2nd ed., Prentice Hall, Upper Saddle River, N.J., 1999.

[6] J. A. Benediktsson, P.H. Swain, and O. K. Ersoy, "Neural network approaches versus statistical methods in classification of multisource remote sensing data," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 28, no. 4, pp. 540-552, Jul. 1990.

[7] C. J. S. Weber, "Competitive learning, natural images and cortical cells," *Network* : *Computation in Neural Systems*, vol. 2, no. 2, pp. 169-187, May, 1991.

[8] O. K. Ersoy, and D. Hong, "Parallel, self-organizing, hierarchical neural networks," *IEEE Trans*. *Neural Networks*, vol. 1, no. 2, pp. 167-178, Jun. 1990.

[9] S. Cho, O. K. Ersoy, and M. R. Lehto, "Parallel, self-organizing, hierarchical neural networks with competitive learning and safe rejection schemes," *IEEE Trans*. *Circuits and Systems II*: *Analog and Digital Signal Processing*, vol. 40, no. 9, pp. 556-567, Sep. 1993.

[10] T. Kohonen, *Self-Organizing Maps*, 2nd ed. Berlin, Springer, 1997.

[11] M. I. Saglam, O. K. Ersoy, and I. Erer, "Self organizing global ranking algorithm and its application," in *Proc*. *of the Artificial Neural Networks in Engineering Conference*, St. Louis, Missouri, Nov. 2004, pp. 893-898.

[12] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, 3rd. ed., New York, McGraw-Hill, 1991.

[13] W. Sun, V. Heidt, P. Gong, and G. Xu , ,"Information fusion for rural land-use classification with high-resolution satellite imagery," *IEEE*. *Trans*. *Geoscience and Remote Sensing*, vol. 41, no. 4, pp. 883-890, Apr.2003.

[14] R. L. Kettig and D. A. Landgrebe, "Computer classification of remotely sensed multispectral image data by extraction and classification of homogeneous objects," *IEEE Trans*. *Geoscience Electronics*, vol. GE-14, no. 1, pp. 19-26, Jan. 1976.

[15] S.M. De Jong, T. Hornstra, and H.G. Maas, "An integrated spatial and spectral approach to the classification of Mediterranean land cover types: The SSC method," *Int'l Journal of Applied Geosciences*, vol. 3, pp. 176-183, 2001.

[16] H. Karakahya, B. Yazgan, and O. K. Ersoy, "A spectral-spatial classification algorithm for multispectral remote sensing data," in *Proc*. *the 13th Int'l Conf*. *Artificial Neural Networks*, Istanbul,Turkey, Jun. 2003, pp. 1011-1017.

[17] D. A. Landgrebe, *Signal Theory Methods in Multispectral Remote Sensing*, JohnWiley & Sons, New York, 2003.

[18] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[19] P. Soille, *Morphological Image Analysis*: *Principles and Applications*, 2nd ed. Springer, Berlin, 2003.

[20] J. Song, and E. J. Delp, "A study of the generalized morphological filter," *Circuits*, *Systems*, *and Signal Processing*, vol. 11, no. 1, pp. 229-252, Mar. 1992.

[21] David A. Landgrebe and Larry Biehl, *Multispec*, May 2003, [online]. Available: http://dynamo.ecn.purdue.edu/~biehl/MultiSpec/

[22] J. A. Richards, and X., Jia, *Remote Sensing Digital Image Analysis*: *An Introduction*, 3rd. ed., Springer, Berlin, 1999.

[23] T. N. Tran, R. Wehrens, and L. M. C. Buydens, "SpaRef: A clustering algorithm for multispectral images," *Analytica Chimica Acta*, vol. 490 no. 1-2, pp.303-312, Aug. 2003.

[24] B. Guindon, Y. Zhang, C. Dillabaugh, "Landsat urban mapping based on a combined spectral-spatial methodology," *Remote Sensing of Environment*, vol. 92 no. 2, pp. 218-232, Aug. 2004.

[25] G. Fumera and F. Roli, "A theoretical and experimental analysis of linear combiners for multiple classifier systems," *IEEE Trans*. *Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 942-956, June 2005.

[26] R.M. Haralick and L.G. Shapiro, "Image Segmentation Techniques," *Computer Vision, Graphics, and Image Computing*, vol. 29, no.1, pp.100-132, Jan. 1985.

[27] L. Xu, A. Krzyzak, and C . Y. Suen, "Methods for combining multiple classifiers and their applications to handwriting recognition," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 22, no. 3, pp. 418-435, May 1992.

[28] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226-239, Mar., 1998.

[29] G. Giacinto, F. Roli, and G. Fumera, "Design of effective multiple classifier systems by clustering of classifiers," in *Proc*. 15th *Int'l Conf. on Pattern Recognition*, Barcelona, Spain, Sep. 2000, vol. 2, pp. 160-163.

[30] A. J. C. Sharkey, "On combining artificial neural nets," *Connection Science*, vol. 8, no. 3, pp. 299-314, Dec., 1996.

[31] D. Partridge, "Network generalization differences quantified," *Neural Networks*, vol. 9, no. 2, pp. 263-271, Mar. 1996.

[32] D. W. Opitz and J. W. Shavlik, "Actively searching for an effective neural network ensemble," *Connection Science*, vol. 8, no. 3-4, pp. 337-353, 1996

[33] D. Partridge and W.B Yates, "Engineering multiversion neural-net systems," *Neural Computation*, vol. 8, no. 4, pp. 869-893, May, 1996.

[34] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans*. *Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993-1001, Oct. 1990.

[35] G. Giacinto and F. Roli, "Design of effective neural network ensembles for image classification purposes," *Image and Vision Computing*, vol. 19, no. 9-10, pp. 699-707, Aug. 2001.

[36] G. Giacinto, F. Roli, and L. Bruzzone, "Combination of neural and statistical algorithms for supervised classification of remote-sensing images," *Pattern Recognition Letters*, vol. 21, no.5, pp. 385-397, May 2000.

[37] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*, Hoboken, NJ; Wiley, 2004.

[38] A. Al-Ani and M. Deriche, "A New Technique for Combining Multiple Classifiers using The Dempster-Shafer Theory of Evidence," *Journal of Artificial Intelligence Research*, vol. 17 pp. 333-361, 2002.

[39] T. K. Ho, J. J. Hull, and S. N. Srihari, "Decision combination in multiple classifier systems," *IEEE Trans*. *Pattern Analysis and Machine Intelligence*, vol. 16, no. 1, pp. 66-75, Jan. 1994.

[40] R. Battiti and A. M. Colla, "Democracy in neural nets: Voting schemes for classification," *Neural Networks*, vol. 7, no. 4, pp. 691-707, 1994.

[41] P. D. Heermann and N. Khazenie, "Classification of Multispectral remote sensing data using a back-propagation neural network," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 30, no. 1, pp. 81-88, Jan. 1992.

[42] S. B. Serpico and F. Roli, "Classification of multisensor remote-sensing images by structured neural networks," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 33, no. 3, pp. 562-578, May 1995.

[43] B. C. K. Tso and P. M. Mather, "Classification of multisource remote sensing imagery using a genetic algorithm and Markov random fields," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 37, no. 3, pp. 1255-1260, May 1999.

[44] Y. Hara, R. G. Atkins, and S. H. Yueh, "Application of neural networks to radar image classification," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 32, no. 1, pp. 100-109, Jan. 1994..

[45] P. H. Swain and R. C. King, "Two Effective Feature Selection Criteria for Multupspectral Remote Sensing," Laboratory for Applications of Remote Sensing, Purdue University, West Lafayette, IN, Technical Note 042673, Nov, 1973.

[46] P. H. Swain and S. M. Davis, *Remote Sensing*: *The Quantitative Approach*, McGraw-Hill, New York, 1978.

[47] T. Kailath, "The Divergence and Bhattacharyya Distance Measures in Signal Selection," *IEEE Trans*. *Communication Technology*, vol. COM-15, no.1, pp. 52-60, Feb. 1967.

[48] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognition*, vol. 26, no.9, pp.1277-1294, Sep. 1993.

[49] J. Stuckens, P. R. Coppin, and M. E. Bauer, "Integrating contextual information with per-pixel classification for improved land cover classification," *Remote Sensing of Environment*, vol. 71, no.3, pp.282-296, Mar. 2000.

[50] D. Landgrebe, "The development of a spectral-spatial classifier for Earth observational data," *Pattern Recognition*, vol. 12, no.3, pp.165-175, 1980.

[51] R. O. Duda, P.E. Hart, and D. G. Stork, *Pattern classification*, 2nd ed. Wiley, New York, 2001.

[52] P. Hsieh and D. Landgrebe, "Lowpass filter for increasing class separability," in *Proc*. *Int'l Geoscience and Remote Sensing Symposium*, vol. 5, July 1998, pp. 2691-2693.

[53] M. Lennon, G. Mercier, and L. Hubert-Moy, "Classification of hyperspectral images with nonlinear filtering and support vector machines," in *Proc*. *Int'l Geoscience and Remote Sensing Symposium*, vol. 3, June 2002, pp.1670-1672.

[54] J. Serra and L. Vincent, "An overview of morphological filtering," *Circuits*, *Systems*, and *Signal Processing*, vol. 11, no. 1, pp. 47-108, Mar. 1992.

[55] P. Maragos and R. W. Schafer, "Morphological filters-Part I: Their set-theoretic analysis and relations to linear shift-invariant filters," *IEEE Trans*. *Acoustics*, *Speech*, *and Signal Processing*, vol. ASSP-35, no. 8, pp.1153-1169, Aug. 1987.

[56] H. J. A. M. Heijmans, "Composing morphological filters," *IEEE Trans*. *Image Processing*, vol. 6, no. 5, pp.713-723, May 1997.

[57] P. Soille and H. Talbot, "Directional morphological filtering," *IEEE Trans*. *Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp.1313-1329, Nov. 2001.

[58] R. A. Peters II, "A new algorithm for image noise reduction using mathematical morphology," *IEEE Trans*. *Image Processing*, vol. 4, no. 5, pp. 554-568, May 1995.

[59] P. Soille and M. Pesaresi, "Advances in mathematical morphology applied geoscience and remote sensing," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 40, no. 9, pp. 2042-2055, Sep. 2002.

[60] C. C. Hung, Y. He, T. Coleman, and K. Qian, "A spatial classification algorithm using peer group pixels," in *Proc*. *International Geoscience and Remote Sensing Symposium*, vol. 6, June 2002, pp. 3405-3407.

[61] G. B. Coleman and H. C. Andrews, "Image Segmentation by Clustering," *Proc*. *IEEE*, vol. 67, no. 5, pp. 773-785, May 1979.

[62] C. Bouman and M. Shapiro, "A multiscale random field model for Bayesian image segmentation," *IEEE Trans*. *Image Processing*, vol. 3, no. 2, pp. 162-177, Mar. 1994.

[63] G. Mercier and S. Derrode, and M. Lennon, "Hyperspectral image segmentation with Markov chain model," in *Proc*. *Int'l Geoscience and Remote Sensing Symposium*, vol. 6, July 2003, pp. 3766-3768.

[64] J. Ton, J. Sticklen, and A. K. Jain, "Knowledge-based segmentation of Landsat images," *IEEE Trans*. *Image Geoscience and Remote Sensing*, vol. 29, no. 2, pp. 222-232, Mar. 1991.

[65] A. Sarkar, M. K. Biswas, B. Kartikeyan, V. kumar, K. L. Majumder, and D. K. Pal, "A MRF model-based segmentation approach to classification for multispectral imagery," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 40, no. 5, pp. 1102-1113, May 2002.

[66] C. Evans, R. Jones, I. Svalbe, and M. Berman, "Segmenting multispectral Landsat TM images into field units," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 40, no. 5, pp. 1054-1064, May 2002.

[67] A. H. S. Solberg, T. Taxt, and A. K. Jain, "A Markov random field model for classification of multisource satellite imagery," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 34, no. 1, pp. 100-113, Jan. 1996.

[68] H. Bischof, W. Schneider, and A. J. Pinz, "Multispectral classification of Landsat-images using neural networks," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 30, no. 3, pp. 482-490, May 1992.

[69] J. Vesanto and E. Alhoniemi, "Clustering of the self-organizing map," *IEEE Trans*. *Neural Networks*, vol. 11, no. 3, pp. 586-600, May 2000.

[70] S. Hashem, "Optimal linear combination of neural networks," *Neural Networks*, vol. 10, no. 4, pp. 599-614, June 1997.

[71] K. Tumer and J. Gosh, "Analysis of decision boundaries in linearly combined neural classifiers," *Pattern Recognition*, vol. 29, no. 2, pp. 341-348, Feb. 1996.

[72] S. Hashem and B. Schmeiser, "Improving model accuracy using optimal linear combinations of trained neural networks," *IEEE Trans*. *Neural Networks*, vol. 6, no. 3, pp. 792-794, May 1995.

[73] J. Kittler and F. M. Alkoot, "Sum versus vote fusion in multiple classifier systems," *IEEE Trans*. *Pattern Analysis and Machine Intelligence*, vol. 25, no. 1, pp. 110-115, Jan. 2003.

[74] L. I. Kuncheva, "A theoretical study on six classifier fusion strategies," *IEEE Trans*. *Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 281-286, Feb. 2002.

[75] D. M. J. Tax, M. V. Breukelen, R. P. W. Duin, and J. Kittler, "Combining multiple classifiers by averaging or by multiplying," *Pattern Recognition*, vol. 33 no. 9, pp. 1475-1485, Sep. 2000.

[76] L. P. Cordella, C. D. Stefano, F. Tortorella, and M. Vento, "A method for improving classification reliability of multilayer perceptrons," *IEEE Trans*. *Neural Networks*, vol. 6, no. 5, pp. 1140-1147, Sep. 1995.

[77] G. C. Vasconcelos, M. C. Fairhurst, and D. L. Bisset," Investigating feedforward neural networks with respect to the rejection of spurious patterns," *Pattern Recognition Letters*, vol. 16, no. 2, pp. 207-212, Feb. 1995.

[78] C. D. Stefano, C. Sansone, and M. Vento, "To reject or not to reject: That is the question-An answer in case of neural classifiers," *IEEE Trans*. *Systems, Man*, *and Cybernetics-Part C*: *Applications and Reviews*, vol. 30, no. 1, pp. 84-94, Feb. 2000.

[79] M. E. Hellman, "The nearest neighbor classification rule with reject option," *IEEE Trans*. *Systems Science and Cybernetics*, vol. SSC-6, no. 3, pp. 179-185, July 1970.

[80] Y. V. Venkatesh and S. K. Raja, "On the classification of multispectral satellite images using the multilayer perceptron," *Pattern Recognition*, vol. 36, no. 9, pp. 2161-2175, Sep. 2003.

[81] R. Adams and L. Bischof, "Seeded region growing," *IEEE Trans*. *Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 641-647, June 1994.

[82] A. Mehnert and P. Jackway, "An improved seeded region growing algorithm," *Pattern Recognition Letters*, vol. 18, no. 10, pp. 1065-1071, Oct. 1997.

[83] K. V. Mardia and T. J. Hainsworth, "A spatial thresholding method for image segmentation," *IEEE Trans*. *Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 919-927, Nov. 1988.

[84] S. K. Pal and P. Mitra, "Multispectral image segmentation using the rough-set-initialized EM algorithm," *IEEE Trans*. Geoscience and Remote Sensing, vol. 40, no. 11, pp. 2495-2501, Nov. 2002.

[85] Y. Liu, S. Yan, and T. Wang, "Study on image-segmented classification," in *Proc*. *Int'l Conf*. *Info-tech and Info-net*, vol. 1, Nov. 2001, pp. 296-301.

[86] R. G. Caves and S. Quegan, "Land cover classification using SAR:A comparison of the segmentation and filtering approaches," in *Proc*. *Int'l Geoscience and Remote Sensing Symposium*, vol. 2, July 1995, pp. 907-909.

[87] K. C. Yao, M. Mignotte, C. Collet, P. Galerne, and G. Burel, "Unsupervised segmentation using a self-organizing map and noise model estimation in sonar imagery," *Pattern Recognition*, vol. 33, no. 9, pp. 1575-1584, Sep. 2000.

[88] Y. Jiang, K. J. Chen, and Z. H. Zhou, "SOM based image segmentation," in *Lecture Notes in Artificial Intelligence*, vol. 2639, Berlin: Springer, 2003, pp. 640-643.

[89] H. Y. Huang, Y. S. Chen, and W. H. Hsu, "Color image segmentation using a self-organizing map algorithm," *Journal of Electronic Imaging*, vol. 11, no. 2, pp. 136-148, Apr. 2002.

[90] S. H. Ong, N. C. Yeo, K. H. Lee, Y. V. Venkatesh, and D. M. Cao, "Segmentation of color images using a two-stage self-organizing network," *Image and Vision Computing*, vol. 20, no. 4, pp. 279-289, Apr. 2002.

[91] X. Otazu, M. Gonzalez-Audicana, O. Fors, and J. Nunez, "Introduction of sensor spectral response into image fusion methods. Application to wavelet-based methods," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 43, no. 10, pp. 2376-2385, Oct. 2005.

[92] J. Nunez, X. Otazu, O. Fors, A. Prades, V. Pala, and R. Arbiol, "Multiresolution-based image fusion with additive wavelet decomposition," *IEEE Trans*. *Geoscience and Remote Sensing*, vol. 37, no. 3, pp. 1204-1211, May 1999.

[93] T. G. Dietteruch, "Ensemble methods in machine learning," in *Lecture Notes in Computer Science*, vol. 1857, Berlin: Springer, 2000, pp. 1-15.

[94] L. I. Kuncheva and C. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine Learning*, vol. 51, no. 2, pp. 181-207, May 2003.

[95] J. A. Benediktsson and J. R. Sveinsson, "Consensus based classification of multisource remote sensing data," in *Proc*. *Int'l Workshop on Multiple Classifier Systems*, June 2000, pp. 280-289.