CHAPTER 3

EFFICIENT NEURAL NETWORK-BASED METHODOLOGY FOR THE DESIGN OF MULTIPLE CLASSIFIERS

N. Vassilas
Institute of Informatics and Telecommunications
National Research Center "Demokritos," Ag. Paraskevi, Attiki
Greece

A neural network-based methodology for time and memory efficient supervised or unsupervised classification in heavily demanding applications is presented in this chapter. Significantly increased speed in the design (training) of neural, fuzzy and statistical classifiers as well as in the classification phase is achieved by: (a) using a self-organizing feature map (SOFM) for vector quantization and indexed representation of the input data space; (b) appropriate training set reduction using the SOFM prototypes followed by necessary modifications of the training algorithms (supervised techniques); (c) clustering of neurons on maps instead of clustering the original data (unsupervised techniques); and (d) fast indexed classification. Finally, a demonstration of this methodology involving the design of multiple classifiers is performed on Land-Cover classification of multispectral satellite image data showing increased speed with respect to both training and classification times.

1 Introduction

Within the last decade, advances in space, sensor and computer technology combined with the launch of new sophisticated satellites have made it possible to amass huge amounts of data about the earth and its environment daily [1]. Applications such as environmental monitoring and resource management as well as geological and geophysical data analysis involve processing large amounts of spatial

© 2000 by CRC Press LLC

data. Often, these data are collected from multiple sources, stored in Geographical Information Systems (GIS), and may include multispectral satellite images (e.g., Landsat TM, SPOT, NOAA AVHRR), grid data (e.g., digital elevation maps, geological/geophysical maps) and point or local measurements (e.g., data from local stations, drill data). Therefore, it is evident that more powerful methodologies are needed for efficient data processing in heavily demanding applications such as the one considered in this paper, namely, satellite image classification.

Recently, several techniques have been proposed for multispectral satellite image classification. These include traditional statistics, neural networks and fuzzy logic and can be divided into two general categories: (a) supervised techniques in which labeled training samples are used for optimizing the design parameters of the classification system [2]-[7], and (b) unsupervised techniques (automatic classification) using a data clustering algorithm [8]-[12].

Although supervised techniques generally perform better in the production of thematic maps (e.g., classification in land-cover categories, geological categories, etc.), unsupervised techniques are mainly used when no training sets are available and constitute a valuable objective alternative as they do not depend on previous knowledge or a photointerpreter's experience. These algorithms first cluster the data according to a similarity criterion, then assign a label to each cluster (usually a grey level or color) that corresponds to a (thematic) category and, finally, substitute each pixel of the original image with the cluster label to which it belongs. The traditional classification scheme using a supervised algorithm or a clustering algorithm is shown in Figure 1.

In a number of recent works [2]-[7], neural network models have successfully been applied for the classification of multispectral satellite images and, more generally, of multisource remotely sensed data. However, for training sets consisting of several thousand patterns belonging to many (often more than ten) categories and large volumes of data, the neural network training and/or classification times reported are quite long, ranging in some cases from a few hours to a few weeks on a conventional computer [4]. The inherent computational complexity in the training, clustering or classification phase of several

other algorithms such as the Pal-Majumder's fuzzy classifier [13], [14], the *k*-nearest neighbors algorithm [15], the various hierarchical clustering procedures [15] and clustering based on scale-space analysis [11], to name a few, also prohibit their use in heavily demanding applications. Therefore, taking into account that additional training and classification trials must usually be performed before selecting a particular classification model, its architecture, and, its design parameters, the need for a methodology for fast model design and classification is evident.

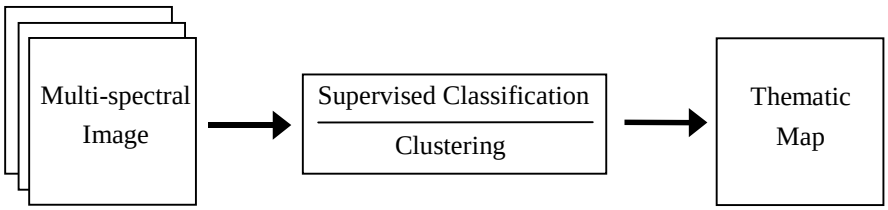


Figure 1. Traditional supervised/unsupervised classification.

In this paper, a methodology based on self-organizing feature maps and indexing techniques is proposed and demonstrated for classifying multispectral satellite images in land-cover categories. The aim is to improve memory requirements for storing the satellite data and at the same time increase training and classification speed without a significant compromise of final performance. This can be accomplished through quantization of the input space, indexed representation of image data, reduction of training (and, possibly, validation) sets, appropriate modification of the training algorithms and, finally, indexed classification.

Results using neural, fuzzy and statistical classifiers show that it is possible to obtain good land-cover maps with the proposed methodology in much shorter times than the classical method of pixel-by-pixel classification. Furthermore, the increased speed achieved allows the design of multiple independent classifiers needed by multimodular systems that combine the decisions of individual classifiers through a voting scheme. One such system that resolves "don't know" cases (i.e., classifiers not in agreement) through local spatial voting is shown to further improve the final result.

2 Proposed Methodology

In this section, a methodology for efficient classification of multi-spectral data is presented with the following advantages:

- memory savings through data quantization,
- increased training speed in supervised algorithms, due to training set size reduction achieved by redundancy removal,
- increased clustering speed in automatic classification, due to the relatively small number of prototypes (quantized data points),
- increased classification speed by using fast indexing techniques.

Although the method can be applied to multisource data without loss of generality, we will restrict the presentation to multispectral satellite images consisting of n bands with $M \times N$ pixels each. The image is represented in the n-dimensional euclidean space $\mathbf{R}^n$ by a set of $M \times N$ points, whereby the grey level values (intensities) in each band at a particular pixel are interpreted as the coordinates of the corresponding point in $\mathbf{R}^n$ (see Figure 2). In other words, the grey levels of each pixel are stacked into a vector (also called the spectral signature of the pixel) that specifies a point in $\mathbf{R}^n$.

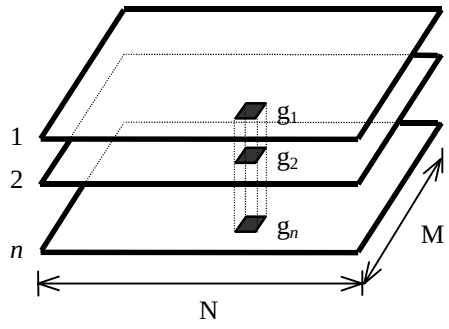


Figure 2. Representation of multispectral images of $M \times N$ pixels and n bands. The spectral signature of a pixel is represented by a point $(g_1, g_2, \ldots, g_n) \in \mathbf{R}^n$ where g_i corresponds to the grey level of the i-th spectral band, $i = 1, 2, \ldots, n$.

2.1 Data Quantization Using Self-Organizing Maps

The first stage of the proposed methodology involves quantization of the input data space using Kohonen's self-organizing feature maps (SOFM) [16]. Using the euclidean distance metric, the SOFM algorithm performs a Voronoi tessellation of the input space and the asymptotic weights of the network (usually a 1-D or 2-D lattice of neurons) can then be considered as a catalogue of vectors or prototypes, with each such prototype representing all data from its corresponding Voronoi cell.

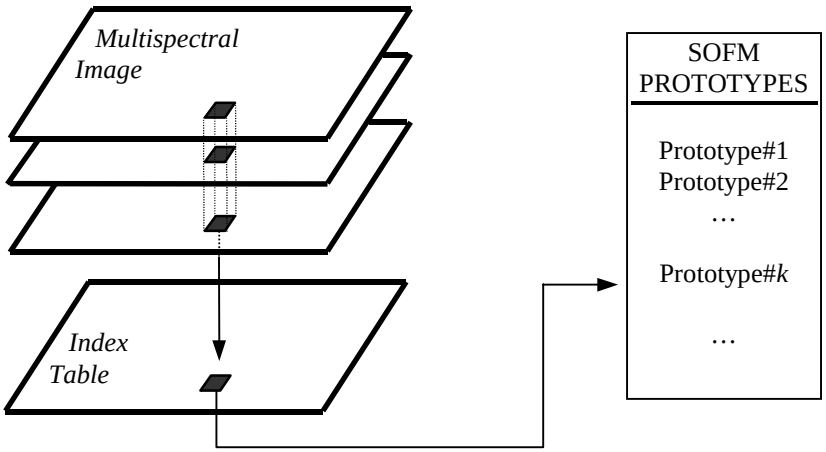


Figure 3. Representation of multispectral images by the index table and SOFM prototypes. The index table stores pointers from pixels to their nearest prototypes.

Following input data quantization, the next step is to derive an indexing scheme that maps each input sample (pixel) to its corresponding prototype. This is achieved by storing, in a 2-D array of the same dimensions as the original image, a pointer to the pixel's closest prototype. This array will be called the *index table* and, along with the SOFM prototypes, constitutes an indexed (compressed) representation that can be used in place of the original image (see Figure 3). Although SOFM training as well as index table production are performed with off-line computations, it is worth noting that the speed can be significantly increased a) by using the branch-and-bound [17] or partition [5] variants of the nearest neighbor algorithm for sequential implementations, or b) through parallel implementations. Recent works on systolic array implementations of Kohonen's algorithm, which

exploit synaptic-level parallelism and allow for fast computations needed in SOFM training and index table construction, can be found in [18], [19].

In general, the larger the number of neurons on the map the better the approximation of the original data space and the smaller the quantization distortion (provided that the map self-organizes). However, from experience, map sizes of up to 16×16 neurons (i.e., 256 prototypes) should suffice in most applications. In the case of large volumes of multispectral data from *n* bands with 256 grey levels/band, compression ratios of approximately *n*:1, when 256 prototypes are used, are readily attainable.

2.2 Training Set Reduction and Classification of SOFM Prototypes for Supervised Techniques

In this section, we demonstrate how fast tuning the parameters of supervised algorithms can be performed using the SOFM prototypes. The training phase involves the use of appropriately selected training and, possibly, validation samples of known classification, with the latter being used to avoid *overtraining* [20]. In satellite image classification applications, these data sets are usually composed of several thousands of pixels and, along with the complexity of the classification task (i.e., the number of categories as well as the optimal shapes of class boundaries), are responsible for the long training times observed. Therefore, it is plausible to seek a reduction of these sets through quantization, preserving at the same time most of the information contained in the original sets. There are two main reasons for this: a) it removes redundancy from the training set, and b) such a reduction speeds up the validation set performance evaluation computed at regular intervals during the training phase.

Using the proposed methodology, a reduction of the training and validation sets can be achieved as follows. First, both sets are quantized by substitution of each sample (spectral signature) with its closest SOFM prototype. In general, the number of prototypes is much smaller than the size of either data set, therefore leading to the existence of many duplicated quantized samples (they fall in the same Voronoi cell). Second, for each class label, we partition the data sets in groups of

identical samples and compute the multiplicity of each group. The reduced sets will have as many different samples as the number of groups under each label with each sample followed by its multiplicity in the group. These multiplicity counts are used in order to preserve the between- and within-class relative frequencies needed to specify optimal boundary placement in overlapping regions (note that identical samples belonging to different classes will both exist in the reduced set). As will be shown in Section 3, simple algorithmic modifications, taking into account sample multiplicities, allow for fast training of supervised models with reduced training and validation sets.

Finally, in the next stage of the proposed methodology the so trained, supervised models are used to classify the weight vectors associated with the neurons of the map (i.e., the catalogue of SOFM prototypes) rather than the original multispectral data. The result obtained is a catalogue of labels (e.g., grey levels or colors) following the same order as the SOFM prototypes (see Figure 4).

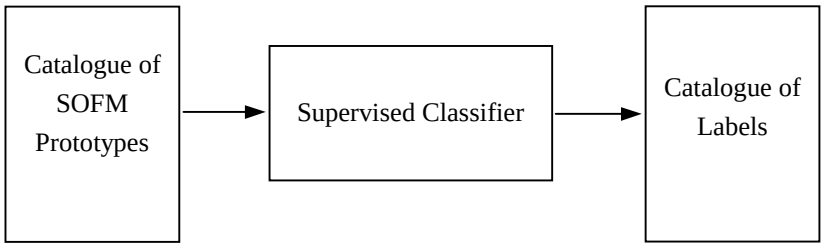


Figure 4. Creation of the catalogue of labels by supervised models based on the reduced training and validation sets.

2.3 Fast Clustering and Labeling of SOFM Prototypes for Unsupervised Techniques

Typically, automatic classification involves clustering of the data space followed by label assignment. However, due to the large number of data points (up to $M \times N$ different spectral signatures), clustering performed on the original image data is inefficient in terms of both memory and time.

In the proposed methodology clustering is performed on the neurons of the map (i.e., the catalogue of SOFM prototypes), thus achieving an increased speed of orders of magnitude, allowing the use of even the

most computationally demanding algorithms such as hierarchical algorithms [15].

Following clustering, the next step is the assignment of arbitrary labels to each cluster. These clusters, along with their labels, will represent the automatic classification categories (see Figure 5).

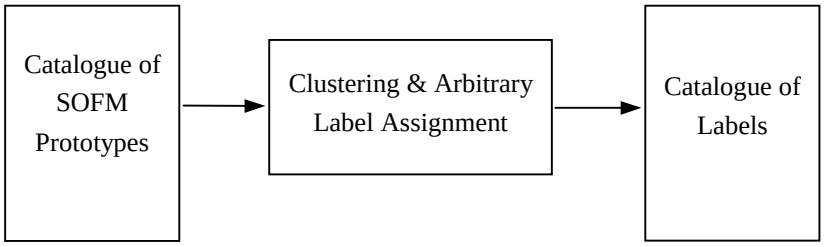


Figure 5. Creation of the catalogue of labels by unsupervised models based on fast clustering and arbitrary label assignment.

2.4 Efficient Indexed Classification

The traditional pixel by pixel classification using supervised or unsupervised techniques requires computational time proportional to the original image dimensions (see Figure 1). The classification of SOFM prototypes and production of the catalogue of labels allows for fast indexed classification by avoiding expensive computations. The final result (thematic map) is now obtained by following the pointers of the index table and accessing the corresponding labels as shown in Figure 6. For large satellite images, an increase in speed of two or three orders of magnitude is possible at this stage.

3 Modifications of Supervised Algorithms

Next, we present the neural, fuzzy and statistical supervised classifiers used in this work as well as the modifications needed in order to take advantage of reduced training and validation sets. In particular, we show how to take into account sample multiplicities when updating weights of neural networks and how to accommodate these multiplicities into fuzzy and statistical classifiers in order to accelerate their computational performance.

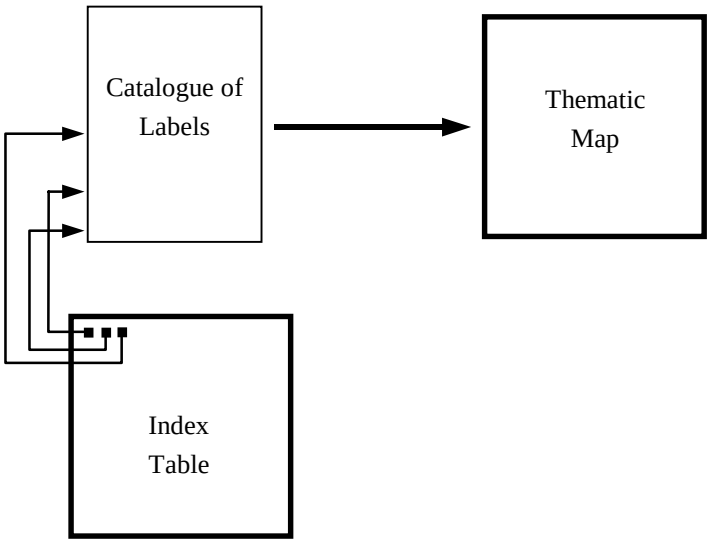


Figure 6. Fast indexed classification using the index table and catalogue of labels.

As far as neural networks are concerned, the goal is for the weight updates to be equivalent to those that correspond to the unreduced but quantized training sets. Hence, under these modifications and assuming that the complexity of the classification task is not significantly affected by the quantization process, training time is reduced approximately in proportion to the ratio between the original (redundant) training set size and the number of prototypes (provided that most of the prototypes exist in the reduced set with few belonging to different classes).

3.1 Classification Using the BP Algorithm

The first classifier used is a multi-layered feedforward network trained with the back propagation (BP) algorithm [21]. In order to speed up the convergence to a local minimum of the error surface, by allowing for relatively large adaptation gain (learning rate) parameters, the *on-line* back propagation version [22] is used in the simulations. In on-line BP, the weights are updated following each input presentation, whereby input patterns are provided in random order. Moreover, by multiplying the weight updates that correspond to a training pattern $\boldsymbol{x}$ with its multiplicity $mult(\boldsymbol{x})$, pattern multiplicities are easily incorporated into the learning procedure. In fact, such a technique implicitly assumes that a group of $mult(\boldsymbol{x})$ inputs is presented in succession to the network and

can be considered a hybrid algorithm, as all patterns within the group cause a batch weight update (in the batch BP version, identical patterns contribute the same amount of weight change) while different groups affect learning in an on-line fashion.

This technique can also be used to balance classes in the training set when they are not equally represented. To speed up learning, one can induce larger weight changes for patterns of poorly represented categories than those of well represented categories. If N_i signifies the number of patterns in class i and $N_{max} = max\{N_i\}$, then N_{max}/N_i can be used as the amplification factor of weight changes induced by patterns from class i. For the unbalanced training set of our simulations, an increase of speed by approximately 4 times was achieved using the above technique. As is the case with the batch and on-line BP versions, selection of network parameters (e.g., adaptation gain) is problem dependent and must be performed after experimentation. However, from experience, the adaptation gain for hybrid algorithms can be selected to be the same as that for the on-line BP version without destroying stochastic convergence.

3.2 Classification Using the LVQ Algorithm

The second neural classifier used in this work is a single-layered network trained with the LVQ algorithm [16], [23]. Although the LVQ algorithm is reputably fast, further improvement on training time can be achieved by following the training set reduction procedure suggested in Section 2.2. To incorporate the multiplicities of the input patterns when updating the reference vectors, the adaptation gain $\alpha(t)$ at time t must be changed to $\alpha'(t) = [1 - (1 - \alpha(t))^{mult(\boldsymbol{x})}]$. It can easily be verified that this modification: a) is equivalent to assuming a repetitive presentation of pattern $\boldsymbol{x}$ $mult(\boldsymbol{x})$ times, considering a constant adaptation gain throughout these repetitions, and b) does not significantly affect the convergence properties of the algorithm (groups are randomly presented and the adaptation gain follows a staircase decaying function with flat portions corresponding to patterns of the same group).

3.3 The Pal-Majumder Fuzzy Classifier

The fuzzy classifier used in this work is similar to the one proposed by Pal and Majumder [13], [14] for vowel and speaker recognition and will be denoted as the PM classifier. To make it easier for readers who are unfamiliar with this, a short presentation of this algorithm follows.

Let, $\boldsymbol{x}^p = (x_1^p, x_2^p, \ldots, x_n^p)$ denote the p-th original pattern ($p = 1, \ldots, P$) of the training set, with x_i^p being its i-th feature element ($i = 1, \ldots, n$). The patterns $\boldsymbol{x}^p$ are transformed into the *fuzzy patterns* $\boldsymbol{y}^p = (y_1^p, y_2^p, \ldots, y_n^p)$ by assigning a membership function $\mu_i(\cdot\ )$ to each of the features x_i^p (this is called the *fuzzification* process):

$$y^p{}_{,i} = \mu_i(x^p{}_{,i}) = (\ 1 + |\ (\bar{x}_{,i} - x^p{}_{,i})\ /\ E\ |^F)^{-1} \quad \forall i = 1, \ldots, n,\ \ \forall p = 1, \ldots, P \tag{1}$$

where $\bar{x}_i = (1/P)\ \sum_{p=1}^{P} x_i^p$ is the mean value of the i-th feature in the training set and E, F are constants determining the shape of the membership functions (spread and steepness of the symmetric membership function).

A new test pattern, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ presented to the fuzzy classifier, is first transformed to a fuzzy pattern $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$ using equation (1) and in the sequel it is compared to each class through *fuzzy similarity scores*. Let $\mathsf{C}_1, \mathsf{C}_2, \ldots, \mathsf{C}_C$ signify the sets of training pattern indices p that correspond to each of the C classes and let $N_1, N_2, \ldots, N_C$ be their respective cardinalities. The similarity scores are found by first computing the *similarity vectors* $\boldsymbol{s}_c(\boldsymbol{y}) = (s_{c1}, s_{c2}, \ldots, s_{cn})$, $c = 1, 2, \ldots, C$, between pattern $\boldsymbol{y}$ and each of the C classes, where

$$s_{ci} = (1/N_c)\ \Sigma_p\ s^p{}_{,ci} \qquad \forall p \in \mathsf{C}_c \quad \text{and} \quad c = 1, 2, \ldots, C \tag{2}$$

and

$$s^p{}_{,ci} = (\ 1 + W_i\ |\ 1 - y_i\ /\ y^p{}_{,i}\ |\ )^{-2z}. \tag{3}$$

The positive constants indicate the relative sensitivity of the classification process to the i-th feature (the lower this value, the higher the sensitivity), and z is a positive integer.

Having computed $\boldsymbol{s}_1(\boldsymbol{y})$, $\boldsymbol{s}_2(\boldsymbol{y})$, …, $\boldsymbol{s}_C(\boldsymbol{y})$ we then classify $\boldsymbol{y}$ to class c if $|\boldsymbol{s}_c(\boldsymbol{y})| < |\boldsymbol{s}_j(\boldsymbol{y})|$ $\forall j \neq c$ (this is the *defuzzification* process) with $|\boldsymbol{s}_c(\boldsymbol{y})| = \sum_{i=1}^{n} s_{ci}$ for $c = 1, 2, \ldots, C$.

It is evident that, by classifying the SOFM prototypes instead of the original pixels ($M \times N$ spectral signatures), this algorithm can efficiently be used in multispectral satellite image classification with the proposed methodology. The reduced training sets result in a further increase in speed of the classification process by P'/P where $P' = \sum_{p=1}^{P} mult(p)$ and P are the sizes of the original and reduced training sets respectively. The only modifications needed on the original algorithm, to incorporate the group multiplicities, are in the computation of the feature means,

$$\bar{x}_i = (1/P') \sum_{p=1}^{P} mult(p)\, x^p_i, \quad \forall i = 1, 2, \ldots, n, \tag{4}$$

in the cardinalities $N_c = \sum_{p \in \mathbb{C}_c} mult(p)$ and in equation (2):

$$s_{ci} = (1/N_c) \sum_{p} mult(p)\, s^p_{ci} \qquad \forall p \in \mathbb{C}_c \quad \text{and} \quad c = 1, 2, \ldots, C. \tag{5}$$

3.4 Classification Using the *k*-NN Algorithm

The final supervised classifier used in this work is one of the simplest and most popular statistical classification methods, namely, the *k*-nearest neighbors (*k*-NN) algorithm [15]. According to the *k*-NN algorithm, a new input pattern $\boldsymbol{x}$ is assigned to the class voted by the majority of its k nearest (in euclidean distance sense) training patterns $\boldsymbol{x}^p$, $p = 1, 2, \ldots, P$.

As was the case with neural and fuzzy classifiers, instead of classifying the original image on a pixel-by-pixel basis, the reduced training set is used to classify the catalogue of SOFM prototypes to a corresponding catalogue of labels followed by the fast indexed classification of Section 2.4. The necessary modifications to incorporate the group multiplicities of the reduced training set in the *k*-NN algorithm are the following: if p_i signifies the index of the i-th nearest neighbor to $\boldsymbol{x}$ then we need only find the $k' \leq k$ nearest neighbors such that

$$\sum_{i=1}^{k'} mult(p_i) \geq k \tag{6}$$

and k' has the minimum value that satisfies equation (6). Since $k' \leq k$, a further increase of the speedup factor is expected.

4 Multimodular Classification

The design of several supervised classification models with independent decisions allows for further improvement of final classification results through the use of multimodular decision making architectures. In a sense, the process of combining the power of different classifiers to obtain optimal classification results simulates the common practice followed by some patients who visit several "independent" doctors in order to obtain uncorrelated diagnoses and then follow the treatment suggested by the majority of them. By analogy, classification results obtained by a single classifier may be absolutely dependent on the particular design and properties of the classifier. Such dependence may have serious effects on final performance, especially when there is significant overlap of the categories and the optimum (in the Bayesian sense) boundaries are non-linear. Taking into consideration the computational complexity of the overall multimodular system, the increase in speed achieved by the proposed methodology in the design of each individual classifier can prove quite beneficial.

The multimodular architecture considered in this work utilizes the simple voting schemes suggested by Battiti and Colla [24]. In their experiments on optical character recognition, multiple neural network classifiers were used and independence of their individual decisions was guaranteed by using different: a) input features, b) number of hidden units, c) initial random weights, and/or d) network models. Each classifier (module) was then allocated a vote and the final decision was made by following a relative or absolute majority rule. Their results with different combinations of modules show an overall superiority of the multimodular system in terms of classification accuracy, with respect to individual classifiers.

In this work, we extend the above to multimodular classification systems that incorporate not only neural but also fuzzy and statistical classifiers. Independence of individual decisions is guaranteed by using different classification models. Absolute majority rules are then applied

for a *primal* classification. Depending on the majority rule, input patterns may be rejected from classification (*don't know* cases resulting from classifier disagreement) and performances can be plotted in the *accuracy/ rejection plane*, whereby an increase in the rejection rate should result in an increase of classification accuracy provided that rejected patterns are close to class boundaries. This added flexibility offers design options that can be exploited when desired performance levels must be obtained even at the expense of an increased rejection rate.

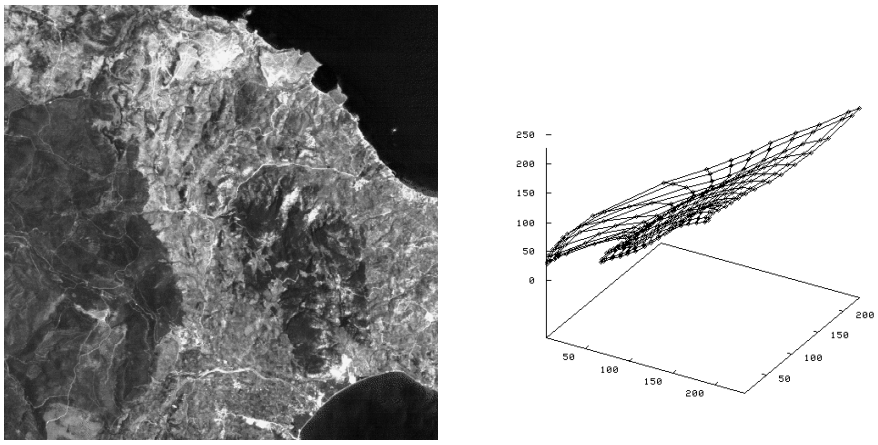
To resolve the problem of *don't know* pixels in the primal classification result we may exploit the spatial property of the image data. To this end, a *don't know* pixel may be given the label of the majority of its local neighbors found in a window centered around the *don't know* pixel. Such a technique can be viewed as spatial *noise filtering*. This cleans the final image and homogenizes its classification regions.

5 Land-Cover Classification

In this section we apply the proposed methodology for supervised and unsupervised classification of a multispectral Landsat TM 512×512 image over Lesvos island in Greece, with a spatial resolution of 30m, into the following four land-cover categories: a) *forest*, b) *sea*, c) *agricultural* and d) *bare rock-inhabited areas*. The satellite data consisted of the first 3 bands (256 grey levels each) that correspond to the red, green and blue regions of the visible spectrum. The original image is shown in Figure 7a.

After delineation of small polygonal regions from each land-cover category by an expert, two sets consisting of 6011 and 3324 samples (3-D spectral signatures) were selected for training the supervised models and testing their classification performance, respectively. In order to assess the generalization capabilities of the supervised algorithms during training, the first of these sets was further randomly split to generate a training set of 4209 samples and a validation set consisting of 1802 samples. The number of patterns in the selected four categories of each of the three sets are shown in Table 1. For the unsupervised models, the same sets of labeled pixels can be used to assess their performance. In Section 5.3, we compare the performance

of the Fuzzy Isodata automatic classifier with its supervised counterparts.



(a) (b)

Figure 7. (a) Original multispectral satellite image used in the simulations, and (b) the self-organized map of 16×16 neurons.

Table 1. Number of patterns in each of the four categories for the three data sets.

Set	Total	Forest	Sea	Agric.	Rock
Training	4209	1375	1054	1434	346
Validation	1802	589	451	614	148
Test	3324	1098	637	944	645

All programs were run on a SUN ULTRA II Enterprise workstation (64MB, 167MHz). A map of 16×16 neurons was trained with 10^5 random presentations of the 3-D spectral signatures in 23.12 sec. The adaptation gain $\alpha(t)$ of Kohonen's SOFM algorithm was selected to tend to zero according to $\alpha(t) = \alpha_0 /(1 + K_\alpha \cdot t)$ with an initial gain $\alpha_0 = 0.3$ and a rate of decay $K_\alpha = 0.002$. The popular rectangular neighborhood function was considered, with a neighborhood size $d(t)$ shrinking with time according to $d(t) = d_{min} + d_0/(1 + K_d \cdot t)$ where $d_{min} = 1$, $d_0 = 7$ and $K_d = 0.0025$. The self-organized map produced with the above settings is depicted in Figure 7b.

Following SOFM training, the storage of asymptotic weights into the catalogue of SOFM prototypes (256 prototypes × 3 floats/prototype × 4

bytes/float × 8 bits/byte = 3×2^{13} bits) and the index table construction (512^2 indices × 8 bits/index = 2^{21} bits) required 54.30 sec.

The SOFM prototypes and index table can also be used for representing the original satellite image (3 bands × 512^2 greys/band × 8 bits/grey = 3×2^{21} bits) in a compressed form. The compression ratio achieved in this case is $3 \times 2^{21} / (3 \times 2^{13} + 2^{21}) = 2.96$ while higher ratios can be obtained for more data bands and/or smaller maps, although caution should be exercised with the latter since small maps may lead to large quantization distortions. The quantized image produced by following the indices and rounding the elements of the prototype vectors to their nearest integers is visually almost indistinguishable from Figure 7a.

5.1 Supervised Classification

The next step is to compress the training and validation sets as explained in Section 2.2. The new sets have 284 and 252 patterns respectively. This is very close to the number of SOFM prototypes. The typical strategy followed in the design of supervised classifiers is to stop training when performance on the validation set is maximized. This approach is used to avoid the well known problem of overtraining [20] and may require experimentation that involves several trials with different parameter values. In the results shown below, a slightly different strategy was used. Instead of maximizing the performance of the validation set alone, we maximized a linear combination of the training and validation performances with coefficients of the linear combination, the 0.3 and 0.7, respectively. The reason for this modification was to assure not only overtraining avoidance but also a good model performance on the training set. Parameter tuning through repetitive experimentation and assessment of training and validation performance at regular intervals adds to "real" training time and provides an additional reason for the importance of the proposed methodology.

Table 2 shows the results obtained with the BP, LVQ, PM and *k*-NN classifiers in terms of final performance on the training, validation and test sets of Table 1, while Table 3 shows the respective increases in training speeds achieved by the proposed methodology. A 3-10-4 feedforward network was trained with the BP algorithm using an

adaptation gain of 0.5 (2.0) and momentum parameter of 0.7 (0.8), both following a staircase decay by a factor of 0.7 every 500 epochs, for the classical (proposed) methodology. Training and validation set performances were assessed every 20 epochs for both methodologies. The training times shown in Table 3 do not take into account the parameter tuning phase and correspond to 1500 epochs (6313500 presentations) for the classical methodology and 220 epochs (62480 presentations) for the proposed methodology. The increase in training speed for the BP classifier, achieved with the proposed methodology, was more than 500 with no significant change in classification accuracy.

Table 2. Performance of the four supervised classifiers for the classical and proposed methodologies (F-*forest*, S-*sea*, A-*agricultural*, R-*rock*).

		Classical Method			Proposed Method		
Model	Category	Training	Valid.	Test	Training	Valid.	Test
BP	F	98.25%	98.64%	94.72%	96.51%	97.62%	91.80%
	S	100.00%	100.00%	100.00%	99.91%	100.00%	98.27%
	A	96.44%	94.79%	93.22%	96.58%	95.11%	94.92%
	R	89.60%	91.22%	95.19%	93.64%	92.57%	97.36%
	Total	**97.36%**	**97.06%**	**95.40%**	**97.15%**	**96.95%**	**95.01%**
LVQ	F	97.16%	98.47%	93.90%	97.53%	98.64%	94.26%
	S	100.00%	100.00%	100.00%	99.91%	99.78%	98.12%
	A	97.07%	95.77%	94.70%	96.37%	95.60%	94.70%
	R	90.46%	91.22%	96.12%	89.31%	89.86%	94.26%
	Total	**97.29%**	**97.34%**	**95.73%**	**97.05%**	**97.17%**	**95.13%**
PM	F	96.58%	97.45%	93.99%	92.80%	92.02%	86.25%
	S	97.82%	96.45%	84.77%	99.62%	99.33%	89.48%
	A	94.42%	92.83%	92.06%	88.84%	87.95%	85.06%
	R	95.09%	94.59%	98.45%	94.22%	93.92%	97.52%
	Total	**96.03%**	**95.39%**	**92.54%**	**93.28%**	**92.62%**	**88.72%**
k-NN	F	97.75%	97.96%	92.35%	97.82%	98.13%	92.53%
	S	100.00%	100.00%	100.00%	99.91%	99.78%	98.12%
	A	97.84%	94.79%	92.58%	97.42%	95.60%	93.64%
	R	91.62%	92.57%	96.12%	87.57%	87.16%	93.49%
	Total	**97.84%**	**96.95%**	**94.61%**	**97.36%**	**96.78%**	**94.10%**

Training of the LVQ classifier was performed with 24 reference vectors (6 reference vectors per category) and a linearly decaying adaptation gain (initial gain equal to 0.3, decay slope = –0.3/5000) for both methodologies. Performances on the training and validation sets were

assessed every 100 iterations. From Tables 2 and 3, we can infer that with no significant change in classifier performance, the increase in training speed achieved was about 4.5. The entries in these tables show that the LVQ algorithm is one of the most appropriate for satellite image classification due to its high speed and good generalization capabilities as long as the dimensionality of the data is relatively small (e.g., 3-D in this application).

Table 3. Increase in training speed for the four classifiers.

	Classical Method		Proposed Method		
Classifier	Number of Presentations	Time (sec)	Number of Presentations	Time (sec)	Speedup
BP	6313500	2847.47	62480	5.31	536.25
LVQ	1000	0.58	2200	0.13	4.51
PM	-	75.25	-	0.55	136.82
k-NN	-	14.57	-	0.26	56.04

Unlike the stochastic training nature of neural classifiers, the fuzzy PM and *k*-NN classifiers use static (non-adaptive) training. However, optimum selection of the PM and *k*-NN design parameters can only be achieved through repetitive classification of the training and validation sets. As with neural classifiers, optimum model selection corresponded to maximization of the combined training and validation performance index by assessing it for various sets of design parameters. In this way, the parameters selected for the PM model were $E = 0.1$, $F = 9.0$, $z = 9$ and $W_i = 0.8$ ($i = 1, 2, 3$) for the original data sets and $E = 0.1$, $F = 12.0$, $z = 12$ and $W_i = 0.8$ ($i = 1, 2, 3$) for the reduced data sets. For the *k*-NN classifier, $k = 5$ was found to be the optimal value of k for both the original and reduced data sets. Table 2 shows that the non-adaptive nature of these algorithms results in a worse generalization than their neural counterparts for either methodology. The increase in speed due to reduced training and validation sets was about 136 and 56 times for the PM and k-NN classifiers respectively. The training times shown in Table 3 correspond to the time needed to assess training and validation performance for a given parameter set of the PM algorithm and for $k = 5$ for the *k*-NN algorithm.

Finally, Table 4 shows the increase in speed achieved in classifying the original 512×512 satellite image. The times reported for the classical method correspond to pixel-by-pixel classification using those

classifiers that have been trained with the original data sets. The times reported for the proposed method include the times for classifying the SOFM prototypes (20ms, 20ms, 370ms and 160ms for the BP, LVQ, PM and *k*-NN classifiers respectively) and the time for the indexed classification (120ms for all classifiers). The increase in speed achieved for the BP and LVQ algorithms was about 72 and 24 times respectively, while that for the PM and *k*-NN algorithms was about 7112 and 2926 respectively, a very impressive result. The final classification results using the original data sets are shown in Figure 8 and those for the reduced data sets are shown in Figure 9.

Table 4. Increase in classification speed of the four algorithms.

Classifier	Classical Method Time (sec)	Proposed Method Time (sec)	Speedup
BP	10.11	0.14	72.21
LVQ	3.45	0.14	24.64
PM	3485.00	0.49	7112.25
k-NN	819.38	0.28	2926.35

5.2 Multimodular Classification

The design of multimodular classification systems in such demanding applications is often prohibited by the time it takes to train the individual classifiers and then classify the data with each one in turn. For the four classifiers used in this work, the total training time (not counting the necessary repetitive trials) is 2937.87 sec while the total classification time is 4317.94 sec.

Using the proposed methodology, the total training and classification times are 6.25 sec and 1.05 sec, respectively, thus encouraging the use of such multimodular classification systems. In this work, the combined decision is based on an absolute majority voting scheme. In particular, we require at least three out of the four classifiers to be in agreement in order to accept the decision. Pixels for which there is not enough agreement (i.e., no three individual classifiers agree on a common class label) are labeled temporarily as "*don't knows*". *Don't know* pixels are then resolved locally at the next stage by giving them the label of the majority of their labeled spatial neighbors found in a 3×3 window centered around them.

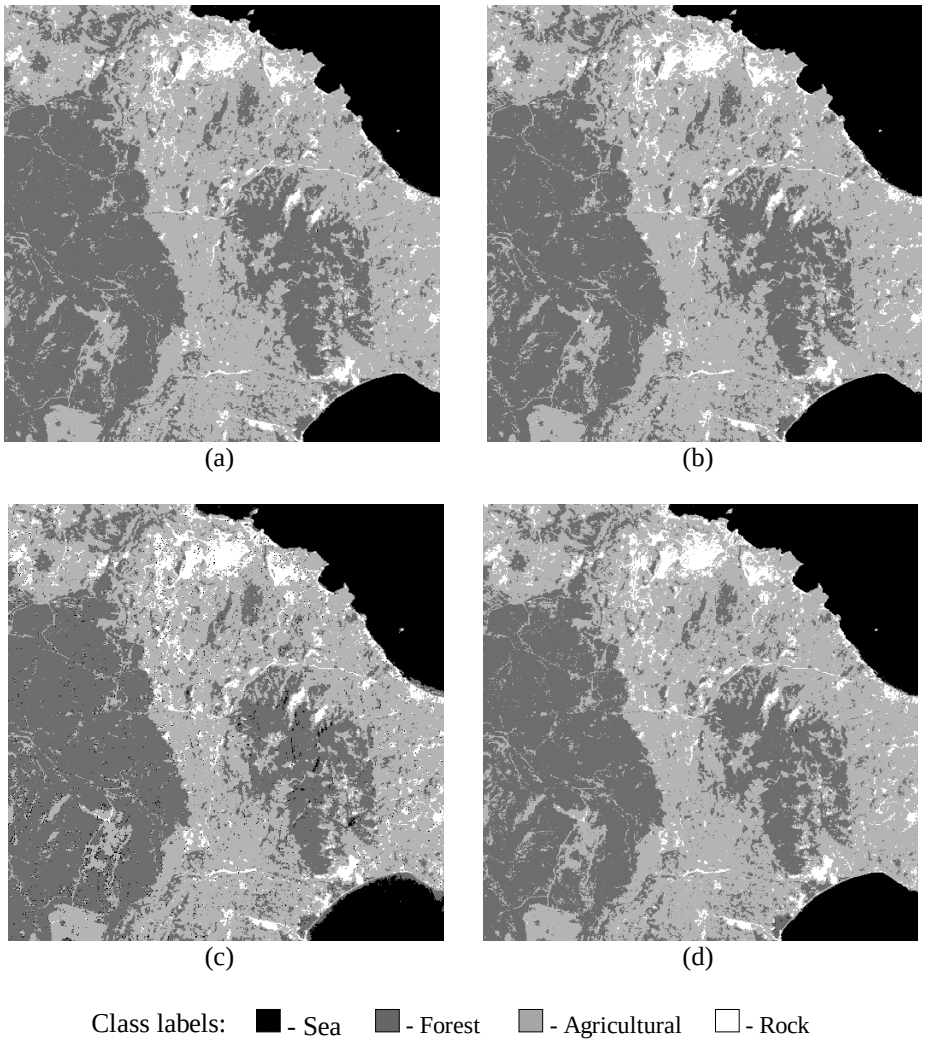


Figure 8. Classification results using the original data sets for the following algorithms: (a) BP, (b) LVQ, (c) PM, and (d) *k*-NN.

Figures 10a and 10b show the classification results obtained using a multimodular classifier on the original and reduced data sets, respectively. Qualitative evaluation and comparison with Figures 8 and 9 shows superior classification quality due to more homogeneous regions.

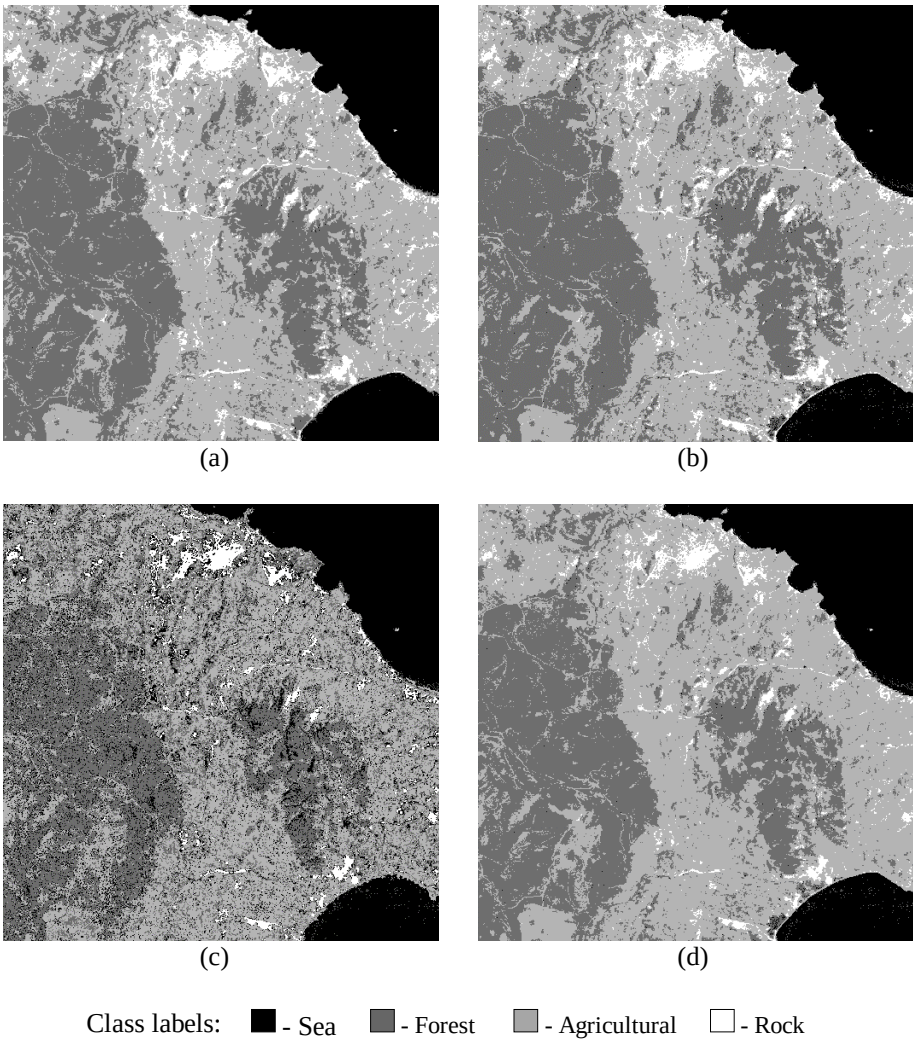


Figure 9. Classification results using the reduced data sets for the following algorithms: (a) BP, (b) LVQ, (c) PM, and (d) *k*-NN.

The time needed to combine the decisions of the four classifiers (only for the SOFM prototypes), indexed classification, and resolving 9309 *don't know* pixels through local information was 0.15 sec for the proposed methodology. The corresponding time for the classical methodology was 0.42 sec (5790 *don't know* pixels).

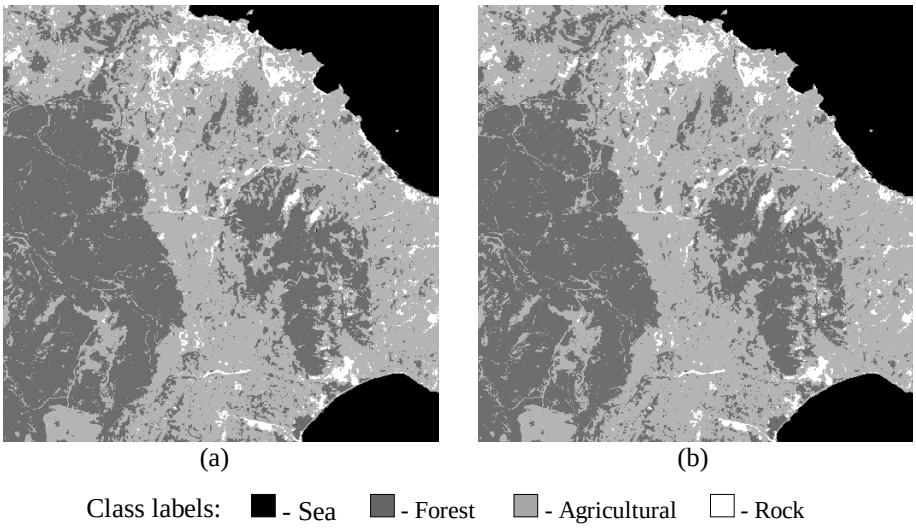


Figure 10. Multimodular classification results using: (a) the classical methodology, and (b) the proposed methodology.

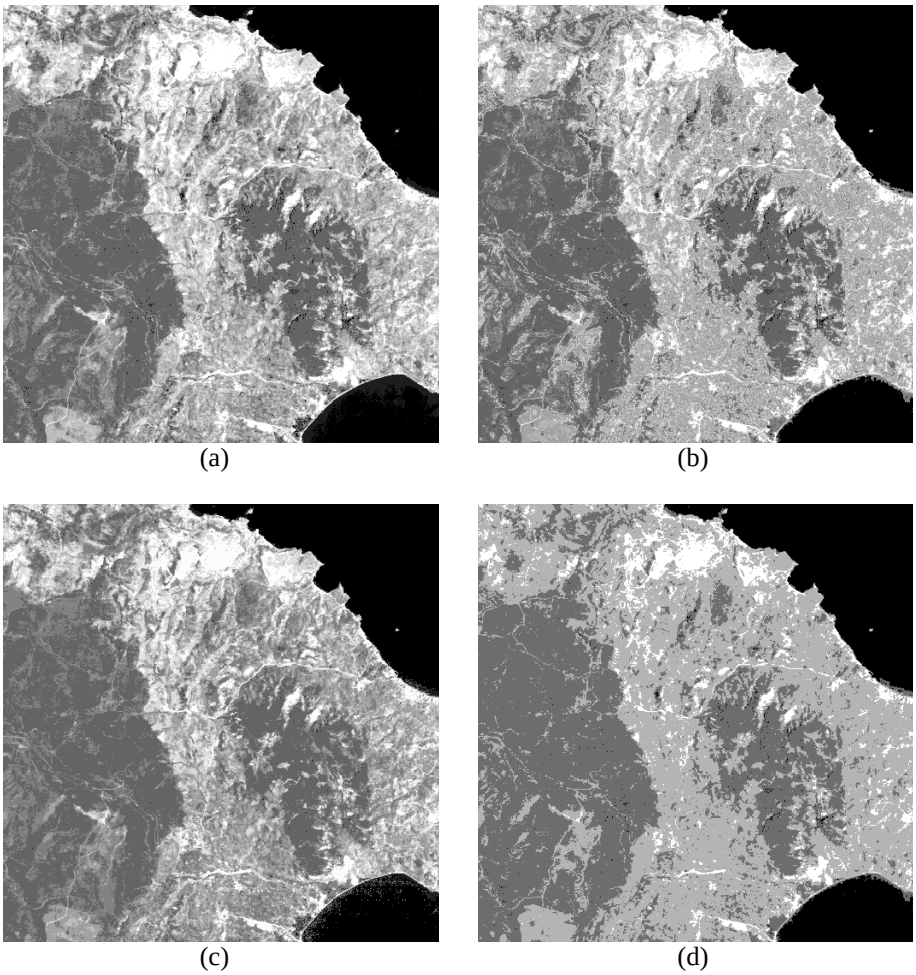
5.3 Unsupervised Classification

The experiments performed in this section mainly consider two of the most popular clustering algorithms, namely the Isodata [15], [25] and Fuzzy Isodata [26] algorithms. However, in order to stress the efficiency of the proposed methodology some practical remarks on the hierarchical min-max statistical algorithm [15] have also been included.

Classification results for 8 categories (subclusters) using the proposed methodology for the Isodata and Fuzzy Isodata algorithms are shown in Figures 11a and 11b, respectively. Convergence of the Isodata algorithm was achieved in 34.71 msec (13 iterations) while Fuzzy Isodata was terminated in 0.967 sec, at 100 (preselected maximum number) iterations. The additional indexed classification time, common to all algorithms, was 0.12 sec.

Figure 11c shows the classification result obtained in 11.90 sec with the hierarchical min-max algorithm. The computational complexity prohibits direct use of this algorithm on the original data (this is a disadvantage when compared with the proposed methodology). On the other hand, direct application of the Isodata and Fuzzy Isodata algorithms to the original data is possible (see Figures 12a and 12b) at a

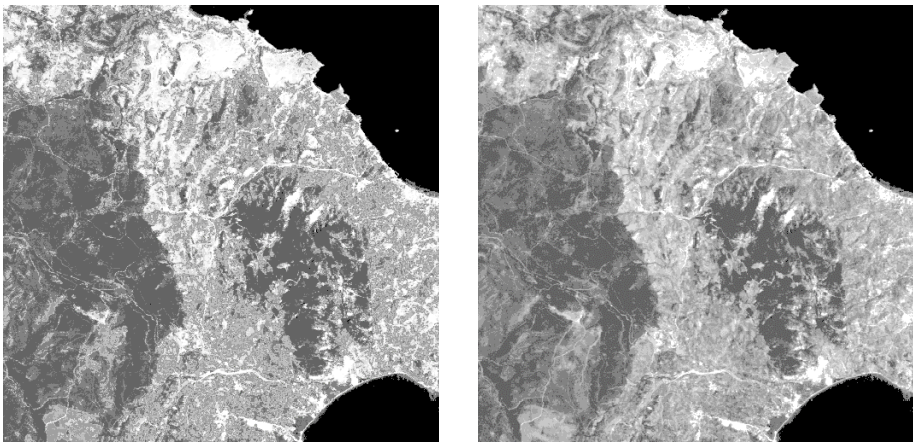
cost of about 1024 times ($512^2/256$) longer clustering time per iteration. In fact, clustering in 8 categories required 172.63 sec for Isodata (63 iterations) and 542.31 sec for Fuzzy Isodata (50 iterations), while the time for classification was 1.02 sec for both algorithms.



(a) (b)

(c) (d)

Figure 11. Classification results in 8 categories with the proposed methodology using: (a) Isodata, (b) Fuzzy Isodata, and (c) hierarchical min-max algorithm. The classification result in 4 categories using Fuzzy Isodata with the proposed methodology is shown in (d).



(a) (b)

Figure 12. Unsupervised classification in 8 categories with the classical methodology using: (a) Isodata, and (b) Fuzzy Isodata.

From the above, the increase in clustering speed per iteration, achieved by using the proposed methodology, is about $2.74/(2.67\times10^{-3}) = 1026$ for Isodata and $10.84/(9.67\times10^{-3}) = 1121$ for Fuzzy Isodata (see Table 5). On the other hand, comparisons in terms of classification speed show an increase in speed, due to indexing techniques, of 1.02/0.12 = 8.5 for both algorithms. At this point, it is important to note that if SOM training and index table construction (requiring 77.42 sec) are not off-line computations, the increase in speed in the first user trial will be smaller. However, for any additional classification trials (with different numbers of clusters) performed by the user for optimizing results, the increase in speed will be as stated above.

Table 5. Computational times and clustering gain (per iteration) for a map of 16×16 neurons. The symbol ∞ means extremely large clustering time.

Clustering Algorithm	Classical Method	Proposed Method	Speedup
Isodata	2.74 sec	2.67 msec	1026
Fuzzy Isodata	10.84 sec	9.67 msec	1121
Hierarchical	∞	11.90 sec	∞

Finally, as far as classification performance is concerned, qualitative evaluation of Figures 11 and 12 through photointerpretation shows very satisfactory results. Quantitative evaluation of the results is also possible through the labeled sets used in the supervised case. For

example, Tables 6-8 show classification performances (in the form of confusion matrices) on training, validation and test data sets using the proposed methodology with the Fuzzy Isodata algorithm. Such a tabular display of the results is useful as it conveys information about the percentage of correct or incorrect data classifications per category (rows of the confusion matrix) and about class overlapping in each cluster (columns of the confusion matrix). Confusion matrices assist the user in finding the optimum number of clusters and/or merging clusters to larger ones so as to satisfy the needs of a particular application.

Table 6. Confusion matrix for the training data set.

Category	Total	SC1	SC2	SC3	SC4	SC5	SC6	SC7	SC8
F	1375	4	0	0	0	1046	0	291	34
S	1054	1043	0	0	0	0	0	11	0
A	1434	0	463	255	30	1	460	37	188
R	346	0	1	13	326	0	6	0	0

Table 7. Confusion matrix for the validation data set.

Category	Total	SC1	SC2	SC3	SC4	SC5	SC6	SC7	SC8
F	589	0	0	0	0	458	0	124	7
S	451	445	0	0	0	0	0	6	0
A	614	0	217	108	16	1	178	21	73
R	148	0	1	5	139	0	3	0	0

Table 8. Confusion matrix for the test data set.

Category	Total	SC1	SC2	SC3	SC4	SC5	SC6	SC7	SC8
F	1098	1	7	0	0	728	1	308	53
S	637	556	0	0	0	1	0	80	0
A	944	0	296	151	28	1	228	39	201
R	645	0	0	16	629	0	0	0	0

The first column of Tables 6-8 refers to the category, the second contains the number of data samples per category and the last 8 columns show the classification results for the 8 subclusters. For example, 255 out of 1434 training samples of category A and 13 out of 346 training samples of category R were classified in subcluster SC3 (see Table 6).

Next, the 8 subclusters are merged to form 4 new clusters (C1, C2, C3, C4) being in 1-1 correspondence with the 4 categories (F, S, A, R) as follows: all initial subclusters with the majority of their data in category F (i.e., columns SC5 and SC7 of Tables 6-8) are merged to form cluster C1, those with data majority in S (i.e., SC1) form cluster C2, and so on. The result of merging and the new confusion matrices are shown in Tables 9-11.

Table 9. Cluster merging for the training set.

Category	Total	C1 = F	C2 = S	C3 = A	C4 = R
F	1375	1337	4	34	0
S	1054	11	1043	0	0
A	1434	38	0	1366	30
R	346	0	0	20	326
Total	4209	1386	1047	1420	356
Performance	**96.75%**	**97.24%**	**98.96%**	**95.26%**	**94.22%**

Table 10. Cluster merging for the validation set.

Category	Total	C1 = F	C2 = S	C3 = A	C4 = R
F	589	582	0	7	0
S	451	6	445	0	0
A	614	22	0	576	16
R	148	0	0	9	139
Total	1802	610	445	592	155
Performance	**96.67%**	**98.81%**	**98.67%**	**93.81%**	**93.92%**

Table 11. Cluster merging for the test set.

Category	Total	C1 = F	C2 = S	C3 = A	C4 = R
F	1098	1036	1	61	0
S	637	81	556	0	0
A	944	40	0	876	28
R	645	0	0	16	629
Total	3324	1157	557	953	657
Performance	**93.17%**	**94.35%**	**87.28%**	**92.80%**	**97.52%**

The last row of Tables 9-11 shows the total classification accuracy as well as the individual accuracies in the four categories for the respective data sets. The individual accuracies are computed as 100% times the ratio of the correctly classified pixels over the total number of

pixels in each category. For example, since 1366 out of the 1434 pixels of category A are correctly classified, the percentage in the last row will be 95.26%. The overall accuracy is found by adding the diagonal elements and then dividing by the total pixels in the data set. A direct comparison of the results shown in Tables 9-11 with those of Table 2 shows that unsupervised algorithms may compete in terms of performance with their supervised counterparts. However, a direct clustering to 4 categories (thus, avoiding the merging steps) would result in representing forest and sea pixels in the same category, since their spectral signatures differ less (in euclidean distance) than pixels within the agricultural category. The result would be a map with two agricultural subcategories, one category for the rock and one category for sea and forest.

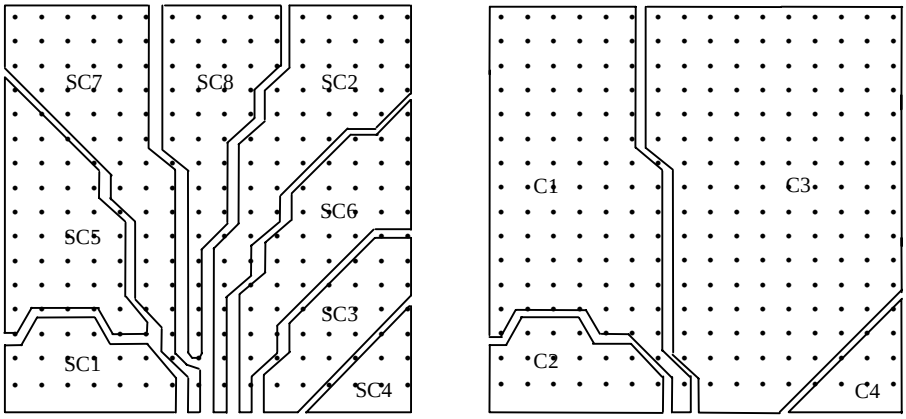


Figure 13. (a) Initial subclusters, and (b) final clusters, for a 16×16 map.

Figure 13a shows the 8 subclusters on the map that correspond to those of Tables 6-8, with dots specifying the position of neurons in the lattice. Figure 13b shows the result of merging and Figure 11d the final classification into 4 categories. Cluster/sub-cluster connectedness is a result of self-organization, while their relative position on the map follows the similarity of their spectral signatures. For example, cluster C2 that corresponds to the sea category is the exclusive neighbor of cluster C1 (forest) and, therefore, most incorrect classifications will be in the forest category rather than in the other categories. This is in agreement with the entries of Tables 9-11.

6 Summary

The methodology described in this paper offers time and memory savings for supervised and unsupervised model design and classification of large volumes of multi-dimensional spatial data using self-organizing maps and indexing techniques. The catalogue of SOFM prototypes together with the index table can be used as a compressed representation of the original data.

An increase in speed in the neural network training phase as well as in selecting the design parameters of fuzzy and statistical supervised classifiers is achieved by size reduction and redundancy removal from the training (and validation) sets in such a way as to preserve most of the information contained in the original data sets. On the other hand, an increase in clustering speed for unsupervised algorithms is achieved due to the relatively small number of SOFM prototypes that represent the original data space, permitting the use of even the most computationally complex algorithms.

Finally, efficient indexed classification, leading to increased speed, is possible by first classifying the “few” SOFM prototypes (relative to the original image data) followed by fast indirect addressing through the index table. Results on land-cover classification of multispectral satellite data show significant increases in speed of training, clustering and classification for a variety of neural, fuzzy and statistical algorithms.

References

[1] Richards, J.A. (1993), *Remote Sensing Digital Image Analysis: An Introduction*, 2nd ed., Springer-Verlag, Berlin-Heidelberg.

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

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

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

[5] Salu, Y. and Tilton, J. (1993), "Classification of Multispectral image data by the binary diamond neural network and by nonparametric, Pixel-by-Pixel Methods," *IEEE Trans. Geosci. Remote Sensing*, vol. 31, no. 3, pp. 606-617.

[6] Charou, E., Ampazis, N., Vassilas, N., Perantonis, S., Feizidis, C., and Varoufakis, S. (1994), "Land-use classification of satellite images using artificial neural network techniques," *Proceedings of Integration, Automation and Intelligence in Photogrammetry, Remote Sensing and GIS - LIESMARS*, Wuhan, P.R.China, pp. 368-377.

[7] Cappellini, V., Chiuderi, A., and Fini, S. (1995), "Neural networks in remote sensing multisensor data processing," *Proceedings of the 14th EARSeL Symposium'94 (Sensors and Environmental Applications of Remote Sensing)*, J. Askne (ed.), Rotterdam: A.A. Balkema, Geteborg, pp. 457-462.

[8] Narendra, P.M. and Goldberg, M. (1977), "A non-parametric clustering scheme for LANDSAT," *Pattern Recognition*, vol. 9, pp. 207-215.

[9] Goldberg, M. and Shlien, S. (1978), "A clustering scheme for multispectral images," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 8, no. 2, pp. 86-92.

[10] Cannon, R.L., Dave, J.V., Bezdek, J.C., and Trivedi, M.M. (1986), "Segmentation of a thematic mapper image using the fuzzy c-means clustering algorithm," *IEEE Trans. Geosci. Remote Sensing*, vol. 24, no. 3, pp. 400-408.

[11] Wong, Y.F. and Posner, E.C. (1993), "A new clustering algorithm applicable to multispectral and polarimetric SAR images," *IEEE Trans. Geosci. Remote Sensing*, vol. 31, no. 3, pp. 634-644.

[12] Baraldi, A. and Parmiggiani (1995), F., "A neural network for unsupervised categorization of multivalued input patterns: an application to satellite image clustering," *IEEE Trans. Geosci. Remote Sensing*, vol. 33, no. 2, pp. 305-316.

[13] Pal, S.K. and Majumder, D.D. (1977), "Fuzzy sets and decision making-approaches in vowel and speaker recognition," *IEEE Trans. Systems, Man and Cybernetics*, vol. 7, pp. 625-629.

[14] Pao, Y.H. (1989), *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, Reading, Massachusetts.

[15] Duda, R.D. and Hart, P.E. (1973), *Pattern Classification and Scene Analysis*, Wiley, New York.

[16] Kohonen, T. (1989), *Self-Organization and Associative Memory*, 3rd ed., Springer, Berlin-Heidelberg-New York.

[17] Niemann, H. and Goppert, R. (1988), "An efficient branch-and-bound nearest neighbour classifier," *Pattern Recognition Letters*, vol. 7, pp. 67-72.

[18] Lehmann, C., Viredaz, M., and Blayo, F. (1993), "A generic systolic array building block for neural networks with on-chip learning," *IEEE Trans. Neural Networks*, vol. 4, no. 3, pp. 400-407.

[19] Ienne, P., Thiran, P., and Vassilas, N. (1997), "Modified self-organizing feature map algorithms for efficient digital hardware implementation," *IEEE Trans. Neural Networks*, vol. 8, no. 2, pp. 315-330.

[20] Haykin, S. (1994), *Neural Networks: A Comprehensive Foundation*, MacMillan, Englewood Cliffs, New Jersey.

[21] Rumelhart, D.E., Hinton, G.E., and Williams, R.J. (1986), "Learning representations by back propagating errors," *Nature*, vol. 323, pp. 533-536.

[22] Fogelman Soulie, F. (1991), "Neural network architectures and algorithms: a perspective," in T. Kohonen, K. Makisara, O. Simula, and J. Kangas (eds.), *Artificial Neural Networks*, pp. 605-615, Elsevier, Amsterdam, The Netherlands.

[23] Kohonen, T., Kangas, J., Laaksonen, J., and Torkkola, K. (1992), LVQ_PAK: The Learning Vector Quantization Program Package, Helsinki University of Technology, Espoo, Finland.

[24] Battiti, R. and Colla, A.M. (1994), "Democracy in neural nets: voting schemes for classification," *Neural Networks*, vol. 7, pp. 691-707.

[25] Ball, G.H. and Hall, D.J. (1967), "A clustering technique for summarizing multivariate data," *Behavioral Science*, vol. 12, pp. 153-155.

[26] Bezdeck, J.C. (1976), "A physical interpretation of fuzzy ISODATA," *IEEE Trans. Systems, Man and Cybernetics*, vol. 6, pp. 387-389.