Springer

Dear Author,

Here are the proofs of your article.

- You can submit your corrections **online**, via **e-mail** or by **fax**.
- For **online** submission please insert your corrections in the online correction form. Always indicate the line number to which the correction refers.
- You can also insert your corrections in the proof PDF and **email** the annotated PDF.
- For fax submission, please ensure that your corrections are clearly legible. Use a fine black pen and write the correction in the margin, not too close to the edge of the page.
- Remember to note the **journal title**, **article number**, and **your name** when sending your response via e-mail or fax.
- **Check** the metadata sheet to make sure that the header information, especially author names and the corresponding affiliations are correctly shown.
- **Check** the questions that may have arisen during copy editing and insert your answers/ corrections.
- **Check** that the text is complete and that all figures, tables and their legends are included. Also check the accuracy of special characters, equations, and electronic supplementary material if applicable. If necessary refer to the *Edited manuscript*.
- The publication of inaccurate data such as dosages and units can have serious consequences. Please take particular care that all such details are correct.
- Please **do not** make changes that involve only matters of style. We have generally introduced forms that follow the journal's style.
  Substantial changes in content, e.g., new results, corrected values, title and authorship are not allowed without the approval of the responsible editor. In such a case, please contact the Editorial Office and return his/her consent together with the proof.
- If we do not receive your corrections **within 48 hours**, we will send you a reminder.
- Your article will be published **Online First** approximately one week after receipt of your corrected proofs. This is the **official first publication** citable with the DOI. **Further changes are, therefore, not possible.**
- The **printed version** will follow in a forthcoming issue.

Please note

After online publication, subscribers (personal/institutional) to this journal will have access to the complete article via the DOI using the URL: http://dx.doi.org/[DOI].
If you would like to know when your article has been published online, take advantage of our free alert service. For registration and further information go to: http://www.springerlink.com.

Due to the electronic nature of the procedure, the manuscript and the original figures will only be returned to you on special request. When you return your corrections, please inform us if you would like to have these documents returned.

Metadata of the article that will be visualized in OnlineFirst

Please note: Images will appear in color online but will be printed in black and white.

ArticleTitle	Self-organizing maps for texture classification	
Article Sub-Title		
Article CopyRight	Springer-Verlag London Limited (This will be the copyright line in the final PDF)	
Journal Name	Neural Computing and Applications	

Corresponding Author	Family Name	**Petrov**
	Particle	
	Given Name	**Nedyalko**
	Suffix	
	Division	
	Organization	School of Computing, University of Portsmouth
	Address	PO1 3HE, Portsmouth, England, UK
	Email	Nedyalko.Petrov@port.ac.uk
Author	Family Name	**Georgieva**
	Particle	
	Given Name	**Antoniya**
	Suffix	
	Division	
	Organization	NDOG, University of Oxford
	Address	OX3 9DU, Oxford, England, UK
	Email	Antoniya.Georgieva@obs-gyn.ox.ac.uk
Author	Family Name	**Jordanov**
	Particle	
	Given Name	**Ivan**
	Suffix	
	Division	
	Organization	School of Computing, University of Portsmouth
	Address	PO1 3HE, Portsmouth, England, UK
	Email	Ivan.Jordanov@port.ac.uk

	Received	4 March 2011
Schedule	Revised	
	Accepted	26 December 2011

Abstract

A further investigation of our intelligent machine vision system for pattern recognition and texture image classification is discussed in this paper. A data set of 335 texture images is to be classified into several classes, based on their texture similarities, while no a priori human vision expert knowledge about the classes is available. Hence, unsupervised learning and self-organizing maps (SOM) neural networks are used for solving the classification problem. Nevertheless, in some of the experiments, a supervised texture analysis method is also considered for comparison purposes. Four major experiments are conducted: in the first one, classifiers are trained using all the extracted features without any statistical preprocessing; in the second simulation, the available features are normalized before being fed to a classifier; in the third experiment, the trained classifiers use linear transformations of the original features, received after preprocessing with principal component analysis; and in the last one, transforms of the features obtained after applying linear discriminant analysis are used. During the simulation, each test is performed 50 times implementing the proposed algorithm. Results from the employed unsupervised learning, after training, testing, and validation of the SOMs, are analyzed and critically compared with results from other authors.

Keywords (separated by '-')	Self-organizing maps - Texture classification - Feature extraction - Statistical analysis - PCA - LDA
Footnote Information	

Journal: 521
Article: 797

Author Query Form

Please ensure you fill out your response to the queries raised below and return this form along with your corrections

Dear Author

During the process of typesetting your article, the following queries have arisen. Please check your typeset proof carefully against the queries listed below and mark the necessary changes either directly on the proof/online grid or in the 'Author's response' area provided below

Query	Details required	Author's response
1.	Please confirm the inserted city name is correct and amend if necessary.	
2.	As per the information provided by the publisher, Fig. 8 will be black and white in print; hence, kindly consider rephrasing the caption without referring to colours	
3.	Please provide a definition for the significance of bold values in tables 4, 8.	
4.	Please check and approve the edit made in the article title.	

Neural Comput & Applic
DOI 10.1007/s00521-011-0797-x

ORIGINAL ARTICLE

Self-organizing maps for texture classification

Nedyalko Petrov · Antoniya Georgieva · Ivan Jordanov

Received: 4 March 2011 / Accepted: 26 December 2011

Abstract A further investigation of our intelligent machine vision system for pattern recognition and texture image classification is discussed in this paper. A data set of 335 texture images is to be classified into several classes, based on their texture similarities, while no a priori human vision expert knowledge about the classes is available. Hence, unsupervised learning and self-organizing maps (SOM) neural networks are used for solving the classification problem. Nevertheless, in some of the experiments, a supervised texture analysis method is also considered for comparison purposes. Four major experiments are conducted: in the first one, classifiers are trained using all the extracted features without any statistical preprocessing; in the second simulation, the available features are normalized before being fed to a classifier; in the third experiment, the trained classifiers use linear transformations of the original features, received after preprocessing with principal component analysis; and in the last one, transforms of the features obtained after applying linear discriminant analysis are used. During the simulation, each test is performed 50 times implementing the proposed algorithm. Results from the employed unsupervised learning, after training, testing, and validation of the SOMs, are analyzed and critically compared with results from other authors.

N. Petrov (✉) · I. Jordanov
School of Computing, University of Portsmouth,
Portsmouth PO1 3HE, England, UK
e-mail: Nedyalko.Petrov@port.ac.uk

I. Jordanov
e-mail: Ivan.Jordanov@port.ac.uk

A. Georgieva
NDOG, University of Oxford, Oxford OX3 9DU, England, UK
e-mail: Antoniya.Georgieva@obs-gyn.ox.ac.uk

Keywords Self-organizing maps · Texture classification · Feature extraction · Statistical analysis · PCA · LDA

1 Introduction

Analysis, recognition, and classification of texture patterns and images are topics with current surge of research interest in the field of digital image processing and pattern recognition, with wide areas of applications [1–5]. A number of different methods, algorithms, and paradigms have been or are being developed nowadays [6–9].

The investigated image classification and recognition systems may vary in their approach but most of them include data acquisition, data preprocessing, feature extraction, feature analysis, classification, and testing and evaluation stages [8–11]. The preprocessing of the raw data is difficult but important part of the whole process, whose aims are to extract useful and appropriate characteristics and features that are to be used in the later stages [8]. Often, the raw data are too large or complex to be used directly as input to a classifier, leading to the "curse of dimensionality" and other problems related to the generalization abilities of the trained systems, especially when insufficient training samples are available. Even if this is not the case, reducing the number of variables representing the data can speed up and facilitate the learning process at later stages [11]. That is why principal component analysis (PCA), for example, is a widely accepted technique in such cases [1, 2, 12].

In [12], we investigated a classification of texture images problem, using supervised neural network learning, for which a priori knowledge about the image classes was used.

The aim of this research is to extend this previous work, considering the same classification problem, but assuming there is no expert knowledge available for the texture

Journal : Large 521 | Dispatch : 31-12-2011 | Pages : 10
Article No. : 797 | LE | TYPESET
MS Code : NCA-1199 | CP | DISK

classes of the data set samples. This implies that no supervised learning can be used, and the knowledge about the texture patterns and their similarity and uniformity has to be extracted from the data set itself. Unsupervised classification of texture patterns and images is widely used approach with applications in a broad range of areas, for example: for determining water quality based on some chemical and physicochemical features [1], for classification of SAR images [2], for texture-based classification of atherosclerotic carotid plaque images for determining risk of stroke for individuals [13], for classifying volcanic ash using surface texture features [3], for automatically classifying texture structure of different fabric types using SOM [14], for classification of textures in scene images using biology inspired features [6], for classification of aerial images using SOMs [15].

In this investigation, a data set of 335 texture images, acquired via an intelligent visual recognition system, as reported in [12], is used. Each data sample of the set represents a grayscale image of an industrial cork tile that was classified in the previous paper into one of seven classes—*Beach, Corkstone, Desert, Lisbon, Pebble, Precision* and *Speckled*. The distribution of the texture classes is non-uniform and is shown in Fig. 1.

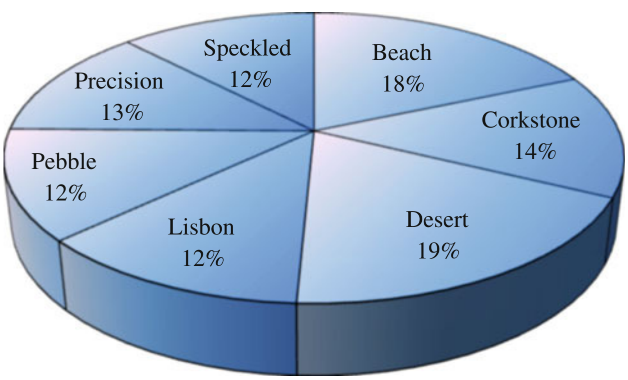


Fig. 1 Distribution of the texture classes

The simulation of the investigated system is divided in five main stages: data acquisition, feature extraction, feature analysis, classifier training, and classifier testing and evaluation.

The rest of the paper is organized as follows: Sect. 2 presents information about the data acquisition, feature extraction, and feature analysis and reduction stages, while Sect. 3 covers the classification stage. The results from the conducted tests are given and discussed in Sect. 4. Finally, Sect. 5 concludes the paper and gives some ideas for future work.

2 Data acquisition and feature extraction

The texture image data set used in this paper is acquired via an intelligent visual recognition system described in more detail in [12]. The system consists of a charge-coupled device camera, lightning devices, and scaffolding. Since the texture of the samples is of prime interest, the images are converted to a grayscale format.

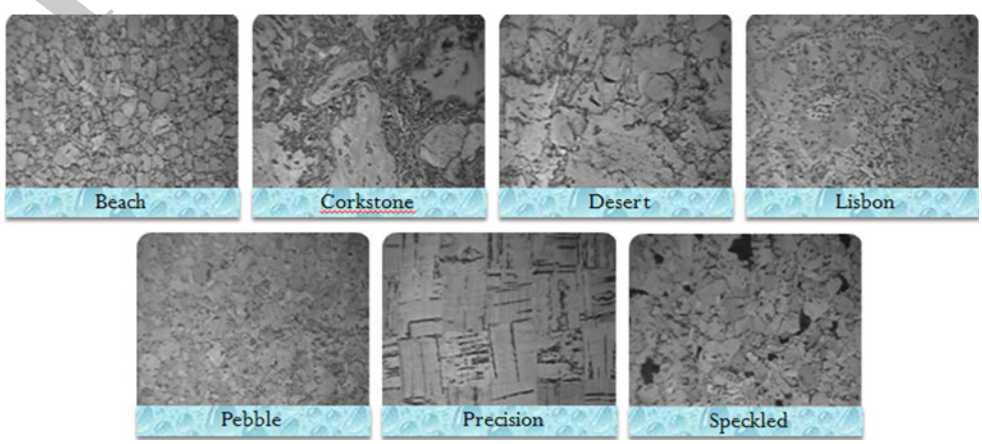
As mentioned above, a total of 335 grayscale images of size 230 × 340 pixels of cork tile samples of 7 predefined by experts types were collected (see Fig. 2).

The feature extraction phase in our investigation aims to identify characteristics and properties that make the classes of samples distinct from each other [16]. At this stage of the process, features that represent some valuable information about the texture of the images are obtained. This is preceded by image normalization.

2.1 Initial feature extraction

In order to reduce the illumination effects on the analyzed images (e.g., due to a glare), a normalization technique is applied. In this process, a small window (15 × 15 pixels) is moved within each image and the local average is subtracted from the pixels' values, in order to get images with average intensity of each neighborhood about a zero [9]. Afterward, 34 features are extracted using classical approaches.

Fig. 2 Samples of the acquired texture data—images of seven different types of wall cork tiles: *Beach*, *Corkstone*, *Desert*, *Lisbon*, *Pebble*, *Precision* and *Speckled*



Journal : Large 521 | Dispatch : 31-12-2011 | Pages : 10
Article No. : 797 | LE | TYPESET
MS Code : NCA-1199 | CP | DISK

2.1.1 Co-occurrence matrices

Co-occurrence matrices, introduced by Haralick in [17], is a commonly applied statistical approach for texture features extraction that takes into account relative distances and orientation of pixels with co-occurring values [9, 15, 18].

The MATLAB's Image Processing Toolbox is used for the computation of the co-occurrence matrices of the normalized images. As usually proposed by other authors [19], four relative orientations are used—horizontal (0°), right diagonal (45°), vertical (90°), and left diagonal (135°). In this way, the *energy*, *homogeneity*, *correlation*, and *contrast* characteristics in each direction are computed, getting as a result the rotation invariant features [9, 11].

Also, two spatial relationships are considered—the direct neighbors and the pixels with difference of five. As a result, a total of eight co-occurrence matrices are obtained—four for the direct neighbors and another four for the pixels with difference of five.

2.1.2 Laws' masks

The Laws' masks are used as a filter technique that is applied to identify points of high energy in an image [20]. Masks are derived from one-dimensional (1-D) vectors of five pixels length, proposed by Laws, to pick up the average gray *level*, *edges*, *ripples*, *spots,* and *waves* [12, 13]:

L_5 (Level) = [1 4 6 4 1] → Level detection;
E_5 (Edge) = [−1 −2 0 2 1] → Edge detection;
S_5 (Spot) = [−1 0 2 0 −1] → Spot detection;
R_5 (Ripple) = [1 −4 6 −4 1] → Ripple detection;
W_5 (Wave) = [−1 2 0 −2 1] → Wave detection.

The vectors are multiplied each other (the second vector is transposed) and this way 25 different 5 × 5 masks are produced. The masks are then applied to the normalized set of samples and the obtained filtered images are converted to texture energy maps. The aim of this process (also called smoothing) is to deduce the local magnitudes of the quantities of interest (edges, spots, etc.). A smoothing window of size 15 × 15 [9] is applied to each filtered image F_k for the k-th mask and new energy images are obtained, where each pixel in the image is given by (1):

$$E_k(r,c) = \sum_{j=c-7}^{c+7} \sum_{i=r-7}^{r+7} |F_k(i,j)|, \quad (k = 1, \ldots, 25), \tag{1}$$

where (r, c) denotes the rows and columns indices. After obtaining 25 energy maps for each image, a power metric, representing the sum of the squared absolute values for each pixel in the map is used [9], to finally obtain 25 different values for each texture sample.

2.1.3 Entropy

Entropy is a statistical measure of randomness that can be used to characterize the texture of an image [9, 14]. It takes low values for smooth images and vice versa.

The entropy for each image sample is calculated using a MATLAB's build-in function, according to (2):

$$E = -\sum_{i=1}^{G} d(i).\log_2 d(i), \tag{2}$$

where G is the number of gray levels in the image's histogram, ranging between 0 and 255 for a typical 8-bit image, and $d(i)$ is the normalized occurrence frequency of each gray level.

2.2 Statistical analysis and feature reduction

Before applying any statistical analysis, a random subset of 25% of the available data is excluded for the purposes of further testing. This subset will be referred to as the testing set from now on and the remaining 75% of the available data will be the training set.

During the feature extraction stage, a total of 34 features are obtained for each texture image (8 by the co-occurrence method, 25 by Law's masks and 1 entropy feature). The distribution of the seven classes of the training set, represented by two randomly selected from the 34 features is shown in Fig. 3. Figure 3b presents the classes' distribution according to the 2nd and the 5th features of the original data set and Fig. 3a shows the classes' means with 95% confidence interval. As it can be seen from Fig. 3, the considerable overlap between the classes makes the classification process more challenging.

In order to reduce the dimensionality of the classification problem (i.e., the number of inputs to the classifier), to reduce the redundant information (i.e., the information contained in some highly correlated features), and to improve the class separability, two statistical analysis techniques [10] are used in some of the experiments. They are described in more details in the next two subsections.

2.2.1 Principal component analysis

PCA is an eigenvalue-based multivariate technique that transforms a number of possibly correlated features into a number of uncorrelated features, called principal components (PC) [2, 9]. The number of the derived PCs is less than or equal to the number of the original features. It is an unsupervised technique and as such does not use any labeled information on the data.

The first PC accounts for as much of the variability (information) in the data, as possible, and each succeeding

Journal : Large 521 | Dispatch : 31-12-2011 | Pages : 10
Article No. : 797 | LE | TYPESET
MS Code : NCA-1199 | CP | DISK

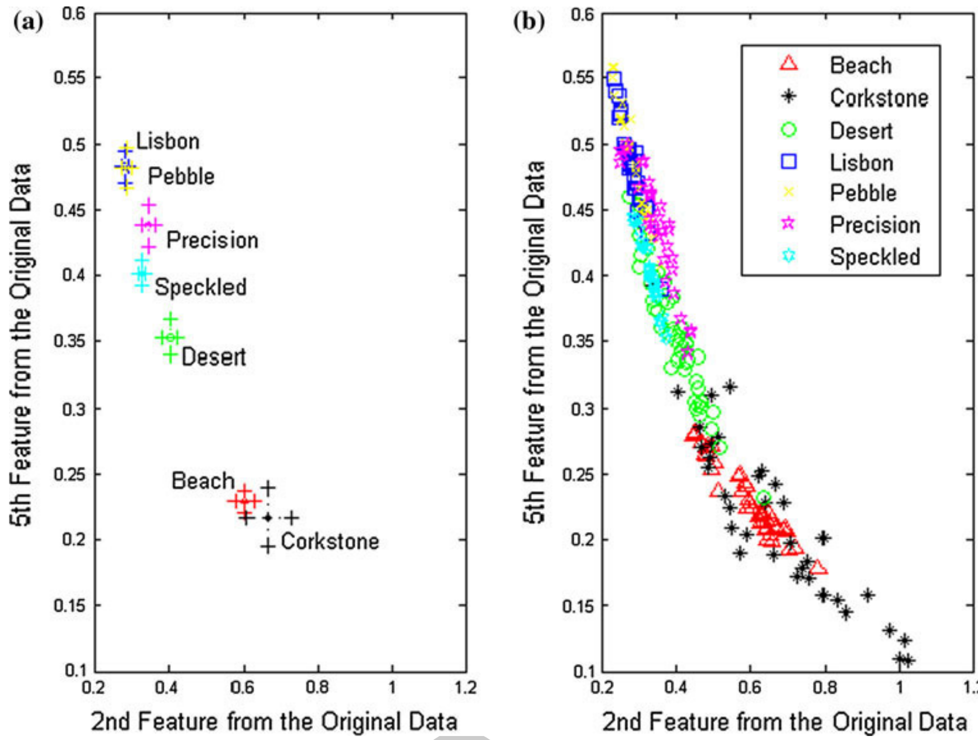


Fig. 3 Texture types distribution, according to two randomly selected features from the training set: **a** classes' means with 95% confidence intervals; **b** scatter plot of the samples

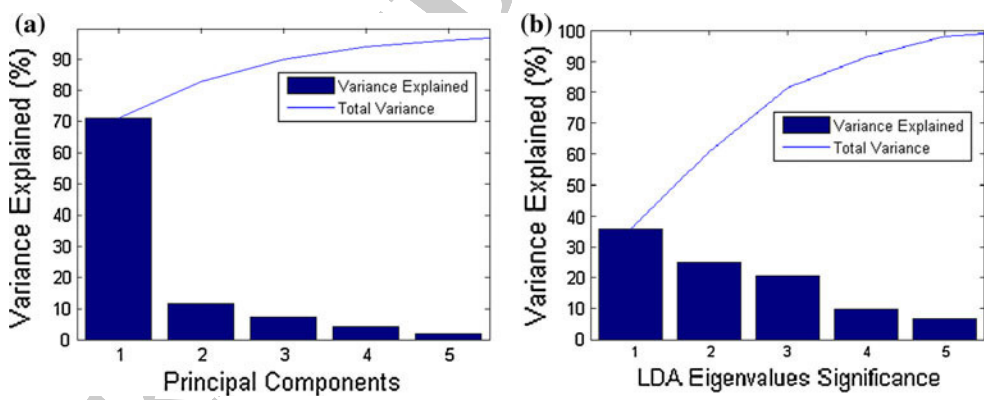


Fig. 4 Percentage of the information from the training set contained: **a** in the first five PCs for the PCA experiment; **b** in the first five eigenvalues for the LDA experiment

PC accounts for as much of the remaining variability as possible. Depending on the areas of application, PCA is also referred to as Hotelling transform, Karhunen–Loeve transform, or proper orthogonal decomposition [9].

The PCA implementation of the MATLAB's Statistics Toolbox is used for processing the extracted features of the training set. As a result, a new data set in which the first 5 features contain about 97% of the total variation (information) is obtained (Fig. 4a). The PCA transformation matrix is saved for further use in the evaluation stage.

Figure 5 shows the distribution of the seven texture classes, represented by the first and second PCs. It can be seen that four out of the seven classes (*Beach*, *Corkstone*, *Desert,* and *Pebble*) are easily separable from the others. However, the rest of the classes are too close to each other and partially overlap. This is because the PCA considers all the data samples independently, without taking into account which class they belong to. The overlapping in some of the classes however is expected to harden the classifiers' performance later on.

2.2.2 Linear discriminant analysis

Linear discriminant analysis (LDA) is an eigenvalues-based transformation technique that aims to find a linear combination of features that characterize or separate two or more classes [9, 21]. LDA is not used in this work as a classification technique, but as a data preprocessing transform, before applying the classification technique, as recommended in [10]. The number of the newly generated

Journal : Large 521 | Dispatch : 31-12-2011 | Pages : 10
Article No. : 797 | ☐ LE | ☐ TYPESET
MS Code : NCA-1199 | ☑ CP | ☑ DISK

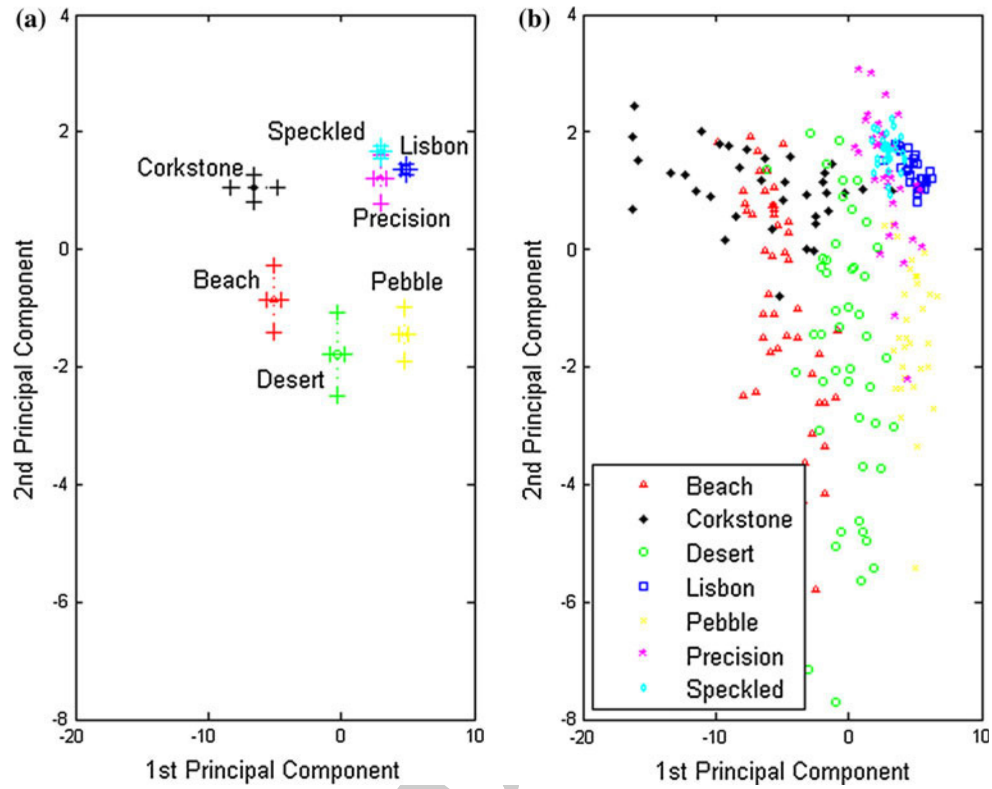


Fig. 5 Texture types distribution, according to the first two PCs: **a** classes' means with 95% confidence intervals; **b** scatter plot of the samples

features is always one less than the number of the classes. An LDA implementation in MATLAB, following the algorithm presented in [21], is employed for this research.

LDA is applied to the features extracted for each texture sample of the training set. As a result, the dimensionality of the feature space is reduced from 34 to 6 without loss of information about the class separability [11] and the LDA transformation matrix is saved for further use in the evaluation stage.

Figure 4b shows the percentage contribution of each eigenvalue to the sum of the six eigenvalues. It can be seen that about 98.5% of the eigenvalues sum is contributed by the first five eigenvalues.

The classes' means with 95% confidence intervals and the scatter plot of the processed with LDA data are shown in Fig. 6. It can be seen that the classes' separability is considerably improved.

3 Classification

For the classification of the texture samples data, self-organizing maps (SOM) are employed. As it is known, a SOM is an artificial neural network (NN) that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called map. A specific characteristic of SOMs (compared to other NNs) is that they use a neighborhood function to preserve the topological properties of the input space [22]. Like most neural networks, SOMs operate in two modes: training and testing. The MATLAB's implementation of SOM is employed for this research and the following algorithm is used for the classification:

1. Design of SOM's architecture (map topology, number of neurons, training parameters, etc.);
2. Training of the SOM with data subset, representing the extracted texture features (75% of the available data set);
3. As a result of step b), a 2D map is obtained, in which each node and its closest neighbors represent similar data samples (Fig. 7);
4. Based on the available expert knowledge for the training samples, the count of the samples belonging to a certain class is determined for each node of the map;
5. Each node is then labeled to represent just one class—the class with predominant number of associated samples. In case equal number of samples of different classes is mapped to a certain node, the node is labeled to the predominant class in its neighborhood (Fig. 7). A node gets no label if there are no data samples mapped to it (the red node in Fig. 7b);
6. The classifier's testing is performed with the remaining 25% of the available data;
7. Each testing sample label is compared to the label of the node that it is mapped to. A sample is counted as unclassified if it is mapped to an unlabeled node;

Journal : Large 521 | Dispatch : 31-12-2011 | Pages : 10
Article No. : 797 | LE | TYPESET
MS Code : NCA-1199 | CP | DISK

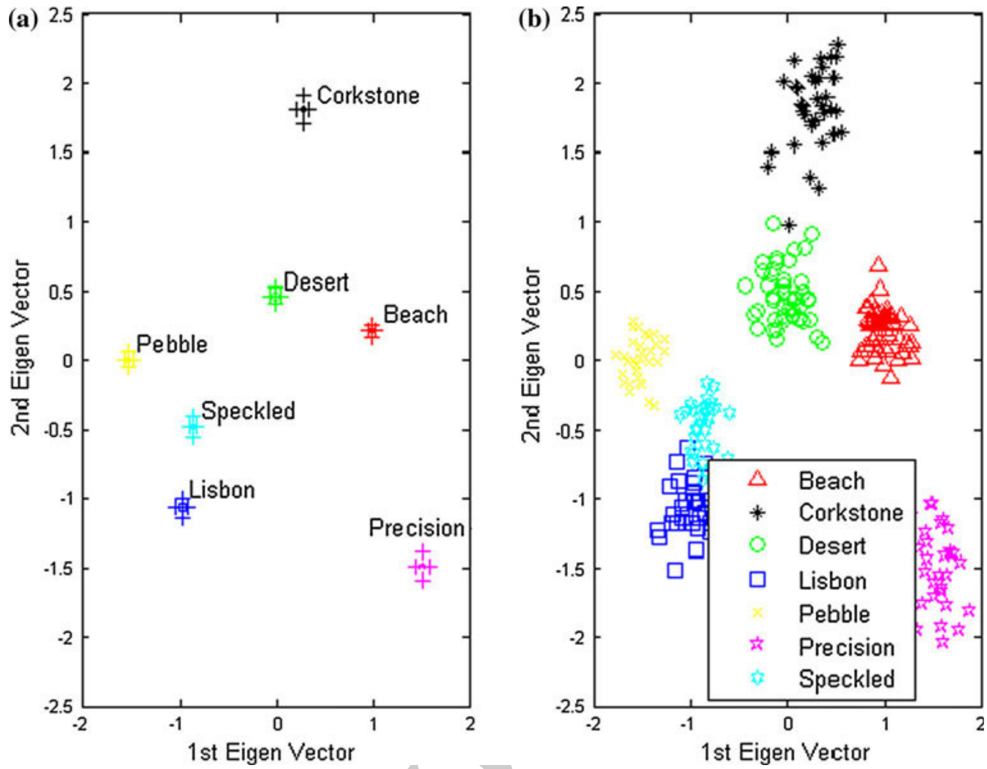


Fig. 6 Texture types distribution, according to the first two eigenvalues: **a** classes' means with 95% confidence intervals; **b** scatter plot of the samples

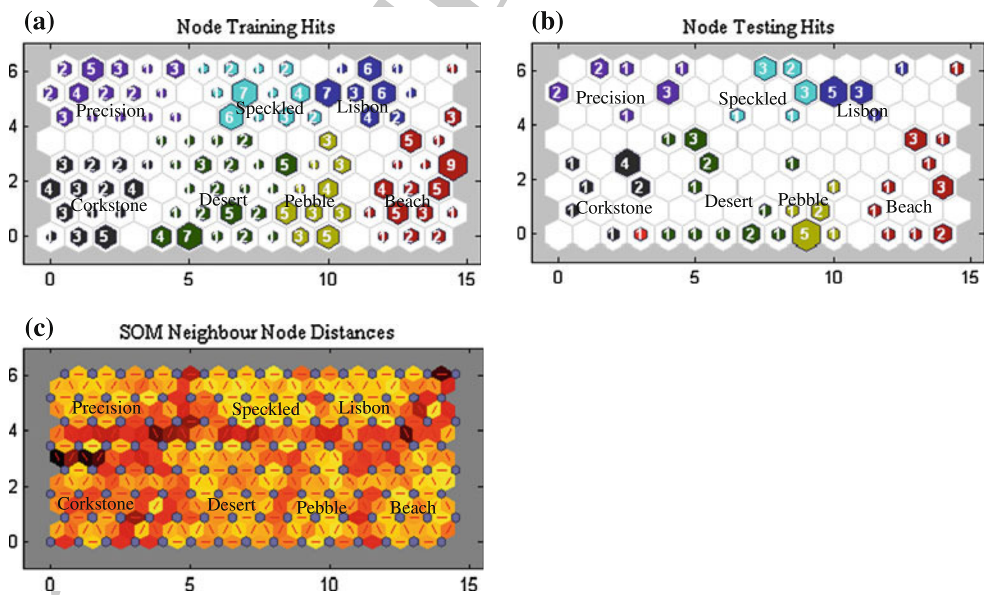


Fig. 7 Sample SOM classifier map. Image **a** presents the node hits for the samples from the training set and **b** from the testing set. The *number* in each node represents its hits. The nodes are colored according to the classes they are labeled to. Image **c** shows the relative distance between the map nodes. *Darker color* corresponds to larger distances

8. The classification accuracy rate is calculated using Eq. 3:

$$a = \frac{n_c}{n_c + n_w + n_u} \cdot 100[\%], \qquad (3)$$

where a is the accuracy of the classifier, n_c is the number of correctly classified samples, n_w is the number of wrongly classified samples and n_u is the number of unclassified samples.

4 Simulation and results

MATLAB 2010B and its Neural Network, Image Processing and Statistics Toolboxes are used for the computations and simulations presented in this paper.

Four major experiments are conducted: in the first one, the classifiers are trained using all the extracted features without any statistical preprocessing; in the second, the extracted features are normalized before being fed to a

Journal : Large 521 | Dispatch : 31-12-2011 | Pages : 10
Article No. : 797 | □ LE | □ TYPESET
MS Code : NCA-1199 | ✔ CP | ✔ DISK

classifier; in the third experiment, the trained classifiers use features obtained after preprocessing with PCA; and in the last one, features obtained after applying LDA are used.

During the simulation, each test is performed 50 times using the algorithm given in Sect. 3. The minimum, maximum, and mean percentages of successfully classified texture images from the testing set are recorded, and the mean standard deviation over the 50 runs is also calculated.

4.1 Classification without statistical preprocessing

In this experiment, SOMs are trained using all the 34 extracted features. No statistical preprocessing is performed, and random 75% (251 texture images) of the available data samples are used for training and the remaining 25% (84 texture images) for testing.

Tables 1 and 3 show results from simulations with varying number of training epochs and varying number of neurons for different SOM's topologies. The sample confusion matrix given in Table 4 shows excellent performance of the classifier for two of the classes (Lisbon and Speckled) and inferior results for the rest.

4.2 Classification with features normalization

In this experiment, all 34 features are used for the SOM's learning and the training set is normalized, so that the features have zero mean and unity standard deviation. Tables 2 and 3 show results from simulations with varying number of training epochs and varying number of neurons for different SOM's topologies. Table 4 gives a sample confusion matrix of the classifier's performance for one run. It can be seen that the classifier's performance is improved, and it is now able to better distinguish most of the classes. However, it still experiences some difficulties with the *Beach* and the *Corkstone* samples.

Table 1 Variation of the classifier's accuracy (in %) for different number of training epochs and no statistical preprocessing

Epochs	50	100	250	500	1,000	2,500	5,000	7,500
Min	48.2	58.0	70.3	70.4	75.3	75.3	74.1	75.3
Max	63.0	75.3	81.5	80.3	81.5	81.5	82.7	82.7
Mean	55.1	66.7	77.0	77.0	78.4	78.3	78.0	78.1
Std	3.6	3.9	2.6	1.9	1.4	1.6	1.9	1.8

SOMs with 120 neurons (15 × 8 map topology) are trained

Table 2 Variation of the classifier's accuracy (in %) for different number of training epochs for SOM with 120 neurons (15 × 8 map topology) after normalization

Epochs	50	100	250	500	1,000	2,500	5,000	7,500
Min	71.6	79.0	84.0	84.0	85.2	85.2	87.7	87.7
Max	86.4	90.1	93.8	93.8	93.8	93.8	95.1	93.8
Mean	77.8	84.9	88.7	89.8	89.9	89.8	90.8	90.9
Std	3.6	3.1	2.4	2.0	2.1	1.8	1.8	1.6

4.3 Classification with PCA

In this case, statistically preprocessed with PCA data is used for the training of SOMs. Again, random 75% (251 texture images) of the available data samples are used for training and the remaining 25% (84 texture images) for testing.

Similarly to the previous case, the number of training epochs, the number of neurons in the SOM, the SOM's topology, and the number of principal components (PC) used for the training are varied. Each sub-experiment is performed 50 times, and the minimal, maximal, and the mean accuracy (%) for these runs are recorded. The results are presented in Tables 5, 7, and Fig. 8a. The sample confusion matrix given in Table 8 shows that this classifier experience slight difficulties recognizing some of the *Corkstone* samples, but performs very well on the rest of the classes.

4.4 Classification with LDA

In the last experiment, SOMs are trained using data statistically preprocessed with LDA, while the same training/testing data ratio (75% training, 25% testing) is kept intact.

The parameters for this experiment are varied through the number of eigenvalues used, the number of training epochs, the number of neurons, and the SOM's topology. Each simulation is performed 50 times, and the minimal, maximal, and the mean accuracy (in %) for these runs are given in Fig. 8b, Tables 6, and 7. Table 8 presents a sample confusion matrix of the classifier's performance for one run. It can be seen that this classifier is able to distinguish all the classes, and the classification error is mainly contributed by the unclassified samples (mapped to an unlabeled node).

4.5 Analysis of the results

Figure 8a illustrates that no significant improvement of the accuracy is obtained when more than 5 principal components are used (PCA case), and for the LDA case (Fig. 8b), the first 3 eigenvalues bring the most significant improvement. This could also be concluded from the graphics given in Fig. 4.

Regarding the SOM's topology, no clear corelation between the accuracy and the number of used neurons was observed (Tables 3 and 7), but more experiments need to be done in order to investigate this in more detail.

Journal : Large 521
Article No. : 797
MS Code : NCA-1199
Dispatch : 31-12-2011
Pages : 10
☐ LE
☑ CP
☐ TYPESET
☑ DISK

Table 3 Variation of the classifier's accuracy (in %) for different number of neurons and different SOM topology (trained for 500 epochs): with no statistical preprocessing on the left side of the cells and after normalization on the right

Neurons	60			120		
Topology	3 × 20	5 × 12	6 × 10	6 × 20	10 × 12	12 × 10
Min	70.4/82.7	69.1/84.0	69.1/85.2	67.9/84.0	70.4/84.0	70.4/85.2
Max	82.7/92.6	79.0/92.6	80.3/92.6	81.5/93.8	81.5/92.6	81.5/92.6
Mean	77.9/88.0	75.2/88.1	75.1/87.9	75.5/88.1	75.9/89.0	76.6/89.1
Std	2.5/2.0	2.4/2.1	2.3/2.0	2.9/2.0	2.0/1.9	2.5/1.6

Table 4 Sample confusion matrix for SOM classifier with 120 neurons (15 × 8 map topology) and 500 training epochs: with no statistical preprocessing on the left side of the cells and after normalization on the right

Actual	Predicted							
	Beach	Corkstone	Desert	Lisbon	Pebble	Precision	Speckled	Unclassified
Beach	**14/13**	1/1	0/0	0/0	0/0	0/0	0/0	0/1
Corkstone	1/0	**8/7**	0/1	0/0	1/2	0/0	0/0	1/1
Desert	2/0	0/0	**10/15**	0/0	1/0	1/0	1/0	0/0
Lisbon	0/0	0/0	0/0	**11/11**	0/0	0/0	0/0	0/0
Pebble	0/0	1/0	0/1	2/0	**8/10**	0/0	0/0	0/0
Precision	1/0	0/0	1/0	1/1	2/0	**5/10**	1/0	0/0
Speckled	0/0	0/0	0/0	1/0	0/1	0/0	**9/9**	0/0

Table 5 Variation of the accuracy (in %) of the classifier for different number of training epochs for SOM with 120 neurons, 15 × 8 map topology, and PCA preprocessing with 5 PCs

Epochs	50	100	250	500	1,000	2,500	5,000	7,500
Min	70.4	74.1	85.2	85.2	85.2	84.0	85.2	86.4
Max	85.2	88.9	92.6	91.4	93.8	92.6	92.6	92.6
Mean	75.6	80.9	89.1	88.8	89.2	88.9	89.5	89.3
Std	2.8	3.3	2.1	1.6	2.0	1.8	1.5	1.5

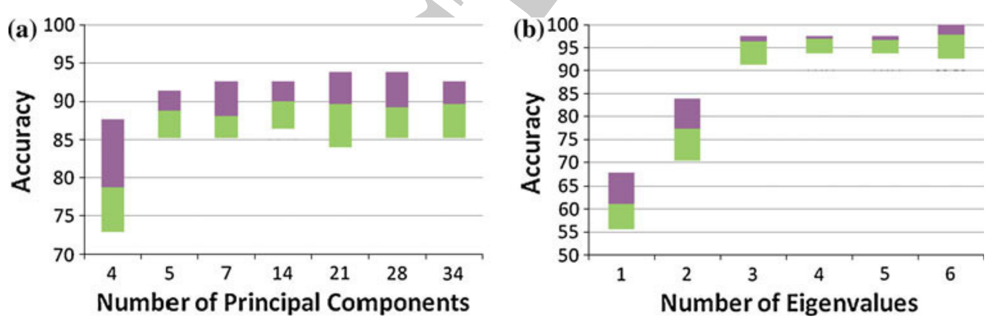


Fig. 8 Variation of the accuracy (in %) of the classifier (SOM with 120 neurons, 15 × 8 map topology, 500 epochs). The border between the *subbars* shows the mean accuracy rate for the 50 runs. The *green* and the *purple sections* show the min and max rate, respectively, for: **a** different number of PCs used for the training (after PCA); **b** different number of eigenvalues used for training (after LDA)

Table 6 Variation of the accuracy (in %) of the classifier for different number of training epochs for SOM with 120 neurons, 15 × 8 map topology, and LDA with 6 eigenvalues

Epochs	50	100	250	500	1,000	2,500	5,000	7,500
Min	85.2	86.4	92.6	92.6	95.1	96.3	95.1	95.1
Max	96.3	98.8	100.0	100.0	100.0	100.0	100.0	100.0
Mean	92.6	93.9	97.7	97.9	98.5	98.2	98.1	98.2
Std	2.9	3.0	1.5	1.3	1.2	1.1	1.1	1.3

Figure 9 summerises and illustrates the obtained results for the four cases, presented in the previous section. It can be seen from the figure that, as expected, the worst accuracy is attained for the case with no statistical preprocessing. Although the accuracy of the normalized data looks better than the obtained one for the PCA case, it has to be noted that only five principal components are considered during the training, whereas in the normalized case, all 34 extracted features are taken into account. The use of only

Journal : Large 521 | Dispatch : 31-12-2011 | Pages : 10
Article No. : 797 | LE | TYPESET
MS Code : NCA-1199 | CP | DISK

Table 7 Variation of the classifier's accuracy (in %) for different number of neurons, different SOM topology, 500 epochs after: PCA with 5 PCs on the left side of the cells and LDA with 6 eigenvalues on the right

Neurons	60			120		
Topology	3 × 20	5 × 12	6 × 10	6 × 20	10 × 12	12 × 10
Min	81.5/96.3	81.5/96.3	82.7/96.3	81.5/95.1	82.7/93.8	84.0/93.8
Max	91.4/100.0	92.6/100.0	91.4/100.0	93.8/100.0	92.6/100.0	91.4/100.0
Mean	86.7/98.7	87.8/99.2	87.4/99.1	87.1/98.6	88.7/97.9	88.4/97.6
Std	2.1/1.1	2.2/0.9	1.8/1.0	2.2/1.4	2.0/1.2	1.7/1.4

Table 8 Sample confusion matrix for SOM classifier with 120 neurons (15 × 8 map topology) and 500 training epochs: with PCA on the left side of the cells and with LDA on the right

Actual	Predicted							
	Beach	Corkstone	Desert	Lisbon	Pebble	Precision	Speckled	Unclassified
Beach	**14/15**	0/0	1/0	0/0	0/0	0/0	0/0	0/0
Corkstone	0/0	**7/10**	1/0	0/0	2/0	0/0	0/0	1/1
Desert	1/0	0/0	**14/14**	0/0	0/0	0/0	0/0	0/1
Lisbon	0/0	0/0	0/0	**11/11**	0/0	0/0	0/0	0/0
Pebble	0/0	0/0	0/0	0/0	**11/11**	0/0	0/0	0/0
Precision	0/0	0/0	0/0	1/0	0/0	**10/11**	0/0	0/0
Speckled	0/0	0/0	0/0	1/0	0/1	0/0	**9/9**	0/0

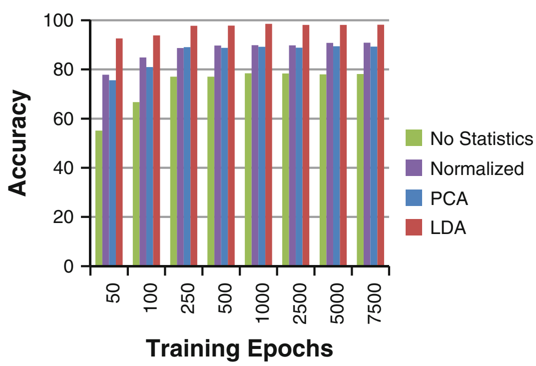


Fig. 9 *Bar graph* showing the accuracy for the four case studies with increasing the number of training epochs

five PCs in the PCA case led to significant reduction in the computational time, compared to the first two experiments.

Analyzing the sample confusion matrices for the four experiments (Tables 4 and 8), it can be said that the accuracy is improved (as expected) after applying LDA and PCA on the data sets, and this is especially valid for the *Desert* and *Precision* classes, while at the same time, the SOM kept excellent recognition rate for the *Lisbon* and *Speckled* classes.

Overall, the achieved accuracy for the LDA case is superior for all runs, outperforming the others by 9% on average. The best results for the LDA are due to the nature of this approach, which uses the samples' lables during the feature analysis. On the contrary, the PCA does not consider the classes when applying ortogonal linear transformation to convert the investigated features to principal components. It can also be observed that the increase in the number of epochs for the runs does not lead to substantial increase in the accuracy, and above 250 epochs, an accuracy plateau is normally reached (Tables 1, 2, 5, and 6).

The results for the PCA case, presented in Tables 5 and 7, are in good agreement with those given in [2], where the authors reported between 81 and 98% accuracy rate for a PCA-based unsupervised classification of SAR images. They are also very close to the [83, 95.5%] achieved in [15] and fall within the intervals with slightly larger accuracy variance, reported in [5, 6], where the results are within the [77, 100%] and [67, 92%] domains, respectively.

5 Conclusion

The investigated texture image recognition of cork tiles is considered as unsupervised classification problem, and SOMs are employed for its solution. The proposed approach includes statistical feature preprocessing techniques (for the purposes of dimensionality reduction and defining optimal number of the features used for the classification) and employing SOM as a classifier for unsupervised classification (NN architecture and topology design, investigating the complexity of the unsupervised learning and the performance of the SOM). For the purpose of comparison, the experiments and simulations of the system are also conducted using the raw data set without any statistical preprocessing. As expected, better results are obtained for the cases when statistical techniques such as PCA and LDA are used (on average about 92% accuracy rate). When LDA is applied, the trained SOMs achieve

Journal : Large 521 Dispatch : 31-12-2011 Pages : 10
Article No. : 797 LE TYPESET
MS Code : NCA-1199 CP DISK

very high accuracy rate—above 98%. This can be expected, as LDA is in fact supervised labeling technique, which makes the classification tasks for the subsequently used SOM much easier.

The comparison of the sample confusion matrices for the four experiments (Tables 4 and 8) shows that the SOM classifiers generally confirm the experts' knowledge about the seven types of texture. However, the visual closeness of some of the misclassified samples to samples from other classes could assist experts to refine the classes' boundaries or to introduce new classes.

Although a straightforward comparison of the methods' performance, based only on the accuracy, can be misleading due to the different complexity of the investigated problems (network's topology parameters, training convergence parameters, differences in the preprocessing techniques, and variations in the number of the investigated features and classes, size and quality of the datasets, etc.), it still can give some indication about the method quality. Nevertheless, as compared with results from other authors in the above paragraph, it can be concluded that while our results of 88% mean accuracy for the PCA case, and above 98% for the LDA case, are generally comparable and competitive for most of the cases, they are also superior in some of the comparisons. It is also interesting to note that in our previous paper [12], the achieved results (86% after PCA and 95% after LDA) are inferior to the ones presented here. This can be attributed to the added entropy feature and the feature normalization, applied before the analysis and classification stages, but would need further investigation in a future work.

References

1. Astel A, Tsakouski S, Barbieri S, Simeonov V (2007) Comparison of self-organizing maps classification approach with cluster and principal components analysis for large environmental data sets. Water Res 41:4566–4578
2. Chamundeeswari VV, Singh D, Singh K (2009) An analysis of texture measures in PCA-based unsupervised classification of SAR images. IEEE Geosci Remote Sens Lett 6:214–218
3. Ersoy O, Aydar E, Gourgaud A, Artuner H, Bayhan H (2007) Clustering of volcanic ash arising from different fragmentation mechanisms using Kohonen self-organizing maps. Comput Geosci 33:821–828
4. Guler I, Demirhan A, Karakis R (2009) Interpretation of MR images using self-organizing maps and knowledge-based expert systems. Digital Signal Process 19:668–677
5. Lei Q, Zheng QF, Jiang SQ, Huang QG, Gao W (2008) Unsupervised texture classification: automatically discover and classify texture patterns. Image Vis Comput 26:647–656
6. Martens G, Poppe C, Lambert P, Van de Walle R (2008) Unsupervised texture segmentation and labeling using biologically inspired features. In: IEEE 10th workshop on multimedia signal processing, vols 1 and 2, pp 163–168
7. Paniagua B, Vega-Rodriguez MA, Gomez-Pulido JA, Sanchez-Perez JM (2010) Improving the industrial classification of cork stoppers by using image processing and neuro-fuzzy computing. J Intell Manuf 21:745–760
8. Shih FY (2010) Image processing and pattern recognition: fundamentals and techniques. Wiley, Hoboken
9. Umbaugh SE (2010) Digital image processing and analysis. CRC; Taylor & Francis, Boca Raton
10. Bishop CM (2004) Neural networks for pattern recognition. Clarendon Press, Oxford
11. Theodoridis S, Koutroumbas K (2009) Pattern recognition. Elsevier/Academic Press, Amsterdam
12. Georgieva A, Jordanov I (2009) Intelligent visual recognition and classification of cork tiles with neural networks. IEEE Trans Neural Netw 20:675–685
13. Christodoulou CI, Pattichis CS, Pantziaris M, Nicolaides A (2003) Texture-based classification of atherosclerotic carotid plaques. IEEE Trans Medical Imag 22:902–912
14. Kuo CFJ, Kao CY (2007) Self-organizing map network for automatically recognizing color texture fabric nature. Fibers Polym 8:174–180
15. Salah M, Trinder J, Shaker A (2009) Evaluation of the self-organizing map classifier for building detection from lidar data and multispectral aerial images. J Spatial Sci 54:15–34
16. Liu H, Yu L (2005) Toward integrating feature selection algorithms for classification and clustering. IEEE Trans Knowl Data Eng 17:491–502
17. Haralick RM, Shanmuga K, Dinstein I (1973) Textural features for image classification. IEEE Trans Syst Man Cybern Smc 3:610–621
18. Kohonen O, Hauta-Kasari M, Parkkinen J, Jaaskelainen T (2006) Co-occurrence matrix and self-organizing map based query from spectral image database. art. no. 603305, ICO20: Illumination, Radiation, and Color Technologies, vol 6033, pp 3305–3305
19. Randen T, Husoy JH (1999) Filtering for texture classification: a comparative study. IEEE Trans Pattern Anal Mach Intell 21:291–310
20. Davies ER (2005) Machine vision: theory, algorithms, practicalities. Morgan Kaufmann, Amsterdam
21. Dillon WR, Goldstein M (1984) Multivariate analysis: methods and applications. Wiley, New York
22. Kohonen T (1990) The self-organizing map. Proc IEEE 78:1464–1480

Journal : Large 521 | Dispatch : 31-12-2011 | Pages : 10
Article No. : 797 | ☐ LE | ☐ TYPESET
MS Code : NCA-1199 | ☑ CP | ☑ DISK