Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2010, Article ID 163635, 23 pages
doi:10.1155/2010/163635

Research Article

Macroscopic Rock Texture Image Classification Using a Hierarchical Neuro-Fuzzy Class Method

Laercio B. Gonçalves[1,2] and Fabiana R. Leta[1]

[1] *Computational and Dimensional Metrology Laboratory (LMDC), Mechanical Engineering Department (PGMEC), Universidade Federal Fluminense (UFF), R. Passo da Pátria, 156, Niterói, Rio de Janeiro, 24210-240, Brazil*
[2] *Computer Department, Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET-RJ), Av. Maracanã, 229, Rio de Janeiro, 20271-110, Brazil*

Correspondence should be addressed to Fabiana R. Leta, fabiana@ic.uff.br

Received 6 October 2009; Revised 22 April 2010; Accepted 17 May 2010

Academic Editor: Panos Liatsis

Copyright © 2010 L. B. Gonçalves and F. R. Leta. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We used a Hierarchical Neuro-Fuzzy Class Method based on binary space partitioning (NFHB-Class Method) for macroscopic rock texture classification. The relevance of this study is in helping Geologists in the diagnosis and planning of oil reservoir exploration. The proposed method is capable of generating its own decision structure, with automatic extraction of fuzzy rules. These rules are linguistically interpretable, thus explaining the obtained data structure. The presented image classification for macroscopic rocks is based on texture descriptors, such as spatial variation coefficient, Hurst coefficient, entropy, and cooccurrence matrix. Four rock classes have been evaluated by the NFHB-Class Method: gneiss (two subclasses), basalt (four subclasses), diabase (five subclasses), and rhyolite (five subclasses). These four rock classes are of great interest in the evaluation of oil boreholes, which is considered a complex task by geologists. We present a computer method to solve this problem. In order to evaluate system performance, we used 50 RGB images for each rock classes and subclasses, thus producing a total of 800 images. For all rock classes, the NFHB-Class Method achieved a percentage of correct hits over 73%. The proposed method converged for all tests presented in the case study.

1. Introduction

Oil is an essential energy resource for industrial production. It can be found in a variety of geological environments. The exploitation of oil is a large-scale activity, in which the acquisition, distribution and use of expert knowledge are critical to decision making. Two sets of data are of fundamental importance in the exploitation of a new oilfield: the oil reservoir geometry and the description of the type of porous rock that holds the oil. When analyzing the oil reservoir geometry, it is possible to identify the amount of oil in the

reservoir. The second set of data consists in describing the porous rock that holds the oil, that is, the reservoir rock. The quality of a reservoir is determined by the particular characteristics of its rocks, such as the minerals that form it, the volume and shape of pores (spaces that store fluids within the rock), the connections between the pores and the physical-chemical processes that may have modified these characteristics. The study of reservoir rocks is based on a systematic description of rock samples collected from oil exploration boreholes. Petrography is an activity performed in the laboratory, which incorporates the results of macroscopic and microscopic rock analyses. In macroscopic analysis, rock samples consist of cylindrical pieces cleaved by drill bit, extracting "witness" samples. Using these samples, slices are withdrawn and prepared in thin sections (0.03 mm), which are then, in turn, analyzed with the use of optical microscopes of polarized light. In macroscopic analyses, several physical characteristics, such as color, structure, texture, grain size, mineral orientation and fossil content, when available, are described. The existence of a large number of classes and subclasses makes the task of rock classification difficult, requiring extensive training, since it depends on the identification of features based on images.

This paper introduces a Neuro-Fuzzy Hierarchical Class model based on binary space partitioning for rock texture image automatic classification, called NFHB-Class method. The system uses the following features: spatial variation coefficient, Hurst coefficient, entropy and cooccurrence matrix. Four rock types have been evaluated by the NFHB-Class method, namely: gneiss (two subclasses), basalt (four subclasses), diabase (five subclasses), and rhyolite (five subclasses). These four rock types are of great interest to evaluate oil boreholes, which is considered by geologists to be a complex task. In the present study, a computer method to solve this problem is presented.

2. NFHB-Class Method

The NFHB-Class [1] method is an extension of the NFHB Inverted method [2, 3] used in data classification. The main difference between the NFHB Inverted method and NFHB-Class method is the construction of system structure. In NFHB Inverted method, the neuro-fuzzy structure is created by NFHB method [4] and the pattern classification system is then formed. On the other hand, the NFHB-Class Method is capable of generating its own pattern classification structure, without the use of NFHB method.

2.1. Basic NFHB-Class Cell

A basic NFHB-Class cell is a mini neuro-fuzzy system that performs a fuzzy binary partitioning in a given area, according to the relevance of functions described by (2.1) and Figure 1(a). The NFHB-Class cell generates two precise outputs (crisp) after a defuzzification process. Figure 1(b) shows the basic NFHB-Class cell representation and Figure 1(c) illustrates it in detail

$$\mu = \text{sig}[a(x_i - b)],$$
$$\rho = 1 - \mu. \tag{2.1}$$

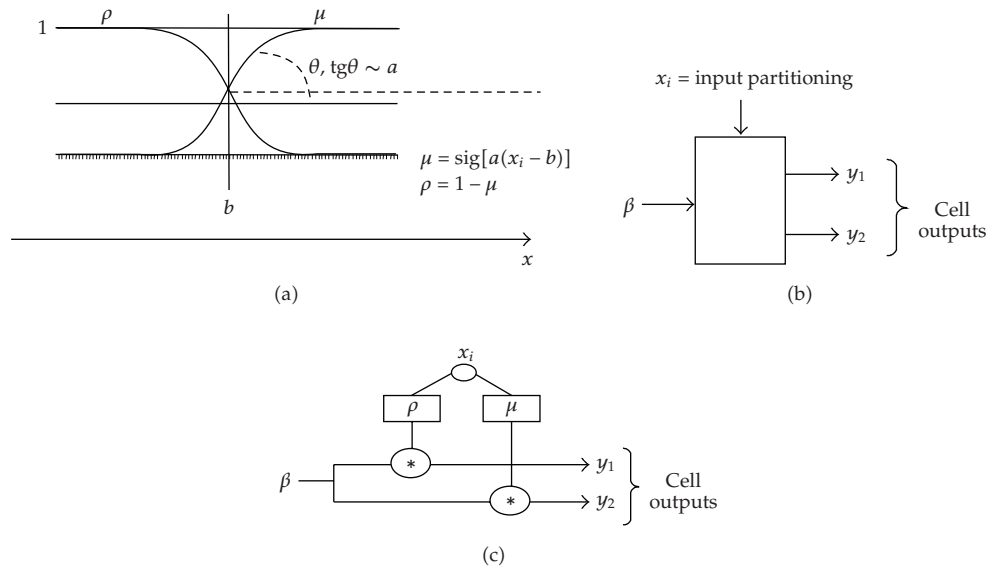

Figure 1: (a) Example of the pertinence functions of the NFHB-Class cell. (b) NFHB-Class cell schematic symbol. (c) NFHB-Class cell.

The outputs (crisp) in an NFHB-Class cell are given by (2.2)

$$y_1 = \frac{\beta * \rho(x)}{\rho(x) + \mu(x)},$$
$$y_2 = \frac{\beta * \mu(x)}{\rho(x) + \mu(x)}, \tag{2.2}$$

where β can be defined by one of the two scenarios below.

(i) The first cell input: in this case $\beta = 1$. The value "1" in the first cell input represents the entire input space, that is, the entire discussion universe of the variable x_i that is being used as an input cell.

(ii) The output of a previous level: in this case $\beta = y_i$, where y_i represents one of two outputs of a generic cell "j", whose value is also calculated by (2.2).

In NFHB-Class basic cell, the high-pertinence function (μ) is implemented by a sigmoid function and the low-pertinence function (ρ) is implemented as its complement $[1 - \mu(x)]$. The complement used leads to a simplification of the defuzzification procedure performed by (2.2), because the sum given by (2.3) (below) is equal to 1 for any "x" value

$$\rho(x) + \mu(x) = 1. \tag{2.3}$$

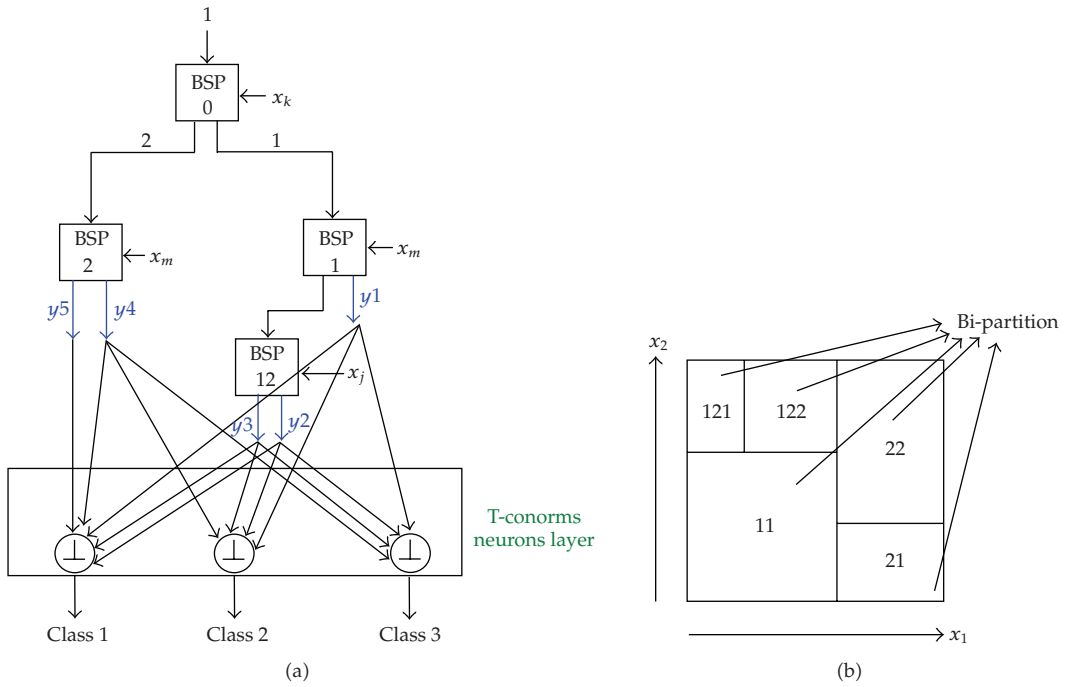

Figure 2: (a) NFHB-Class Architecture. (b) Input space partitioning of NFHB-Class Model.

Therefore, the output equations, given by (2.2), are simplified to:

$$\begin{aligned} y_1 &= \beta * \rho(x), \\ y_2 &= \beta * \mu(x), \end{aligned} \tag{2.4}$$

or

$$y_i = \beta * \alpha_i(x), \tag{2.5}$$

where α_i's are the firing levels of each rule (partition) and are given by $\alpha_1 = \rho$ *and* $\alpha_2 = \mu$.

2.2. NFHB-Class Architecture

Figure 2(a) shows an example of NFHB-Class architecture, obtained in the training system and applied to a database that has three distinct classes; while its corresponding partition is illustrated in Figure 2(b). The structure has been created automatically without the need of using NFHB method in the training stage.

In NFHB-Class architecture, the system has several outputs which are connected to T-conorm cells that define the classes (Figure 2(a)). The system output (class 1, class 2 or class 3) with the highest value defines the class to which the pattern belongs.

The outputs of the leaf-cells are listed below:

$$\begin{aligned} y1 &= \rho_0 \cdot \rho_1, \\ y2 &= \rho_0 \cdot \mu_1 \cdot \rho_{12}, \\ y3 &= \rho_0 \cdot \mu_1 \cdot \mu_{12}, \\ y4 &= \mu_0 \cdot \rho_2, \\ y5 &= \mu_0 \cdot \mu_2. \end{aligned} \tag{2.6}$$

After calculating the output of each leaf cell, the connection between these cells to the T-conorm neurons is performed, and then the final output of the pattern classifier is obtained. Each T-conorm neuron is associated with a specific class, as can be seen in the example in Figure 2(a), where there are three distinct classes, and consequently, three T-conorm neurons.

The connections between the leaf-cells with the T-conorm neurons are made, initially, by connecting all the leaf-cells with all the T-conorm neurons, according to the number of classes in which the database is structured. After this connection, it is necessary to establish weights for these connections (arcs). For determining the weight allocation, we used the least squares method, applying Gauss-Seidel iterative method [5].

Once we defined the strategy of how the leaf-cells were connected to T-conorm neurons and their corresponding weight links, it was necessary to define which T-conorm operators would be used to obtain the final output of the NFHB-Class. All the outputs of the leaf-cells were connected to all T-conorm neurons, as shown in Figure 2(a). Each of these outputs is multiplied by the weight of its corresponding connection with T-conorm neurons. The next step is then to define how all of these T-conorm neurons input data derived from the leaf-cells should be treated.

In the example of Figure 2(a), the outputs of three T-conorm neurons are calculated according to (2.7)

$$\begin{aligned} &y1 * w_{11} \oplus y2 * w_{21} \oplus y3 * w_{31} \oplus y4 * w_{41} \oplus y5 * w_{51}, \\ &y1 * w_{12} \oplus y2 * w_{22} \oplus y3 * w_{32} \oplus y4 * w_{42} \oplus y5 * w_{52}, \\ &y1 * w_{13} \oplus y2 * w_{23} \oplus y3 * w_{33} \oplus y4 * w_{43} \oplus y5 * w_{53}, \end{aligned} \tag{2.7}$$

where:

(i) *y1, y2, y3, y4,* and *y5* are the outputs of the leaf-cells;

(ii) W_{11}, W_{12}, W_{13}, W_{21}, W_{22}, W_{23}, W_{31}, W_{32}, W_{33}, W_{41}, W_{42}, W_{43}, W_{51}, W_{52}, and W_{53} are the weights of the link between the leaf cell and the T-conorm neuron;

(iii) $\oplus$ is the T-conorm operator used for processing the neuron output.

In this paper, the limited-sum T-conorm operator [6] has been used. This operator is the most appropriate in this case, since it considers all inputs in the output calculation. It is worth noting that another T-conorm operator that is very popular in the literature, the *max operator*, only considers the maximum membership value, thus ignoring the membership values of the input data.

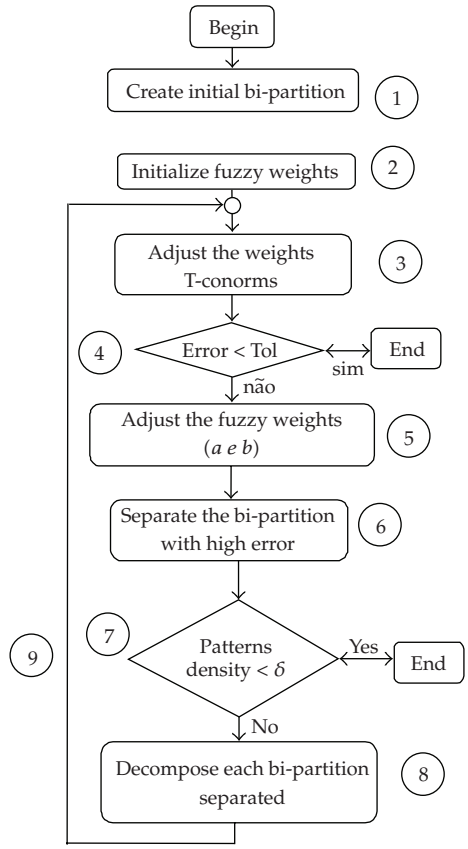

Figure 3: Learning algorithm of NFHB-Class Model.

The final output obtained in the NFHB-Class Method is determined by the highest output obtained among all the T-conorm neurons, thus indicating the class to which the input pattern belongs.

2.3. Learning Algorithm

In neuro-fuzzy literature, the learning process is generally divided in two spheres: (1) structure identification and (2) parameter adjustments. The NFHB-Class method follows the same process. However, the NFHB-Class method performs both learning tasks simultaneously. The learning algorithm of this method follows nine steps, as illustrated in Figure 3.

Learning steps.

(1) An initial bipartition is created, dividing the input space into two fuzzy sets—*high* and *low*—for the input variable x. In this step, the first NFHB-Class cell is created, which is called a root cell.

(2) The b parameter (the sigmoid inflexion point, Figure 1(a)) is initialized. The other sigmoid parameter a is heuristically initialized. Equation (2.8) illustrate the

initialization process:

$$a = \frac{2}{(\text{Lim}\, S - \text{Lim}\, I)},$$
$$b = \frac{(\text{Lim}\, S + \text{Lim}\, I)}{2}, \tag{2.8}$$

where Lim I and Lim S are the lower and upper limits of the cell's input variable.

(3) In this step, the weights of the arcs that connect the cells to T-conorm neurons are adjusted by ordinary least squares method (as described in Section 2.2). After calculating the weights, the value of each T-conorm output neuron of the NFHB-Class is computed.

(4) The total system error is then calculated for the entire training set, according to the following root-mean-square (rms) error expression, (2.9):

$$E_{\text{RMS}} = \sqrt{\frac{1}{L}\sum_{n=1}^{L}\sum_{c=1}^{\#\text{classes}}\left(y_{nc} - y_{nc}^{d}\right)^{2}}, \tag{2.9}$$

where L = number of patterns in the training set; y_{nc} is the T-conorm neuron output, which represents the "c" class of the index "n" pattern; y_{nc}^{d} is the desired T-conorm neuron output which represents the "c" class of the index "n" pattern.
For example, we can consider a database with 3 different classes (Class 1, Class 2 and Class 3). There are also "j" and "k" database index belonging to Class 2 and 3, respectively. Thus, we have the following desired outputs to the pattern "j" given by: $y_{j1}^{d} = 0$, $y_{j2}^{d} = 1$, $y_{j3}^{d} = 0$, for the pattern k, they are given by: $y_{k1}^{d} = 0$; $y_{k2}^{d} = 0$, $y_{k3}^{d} = 1$. Thus, the system error considering these two patterns (index "j" and "k") is given by (2.10):

$$E_{\text{RMS}} = \sqrt{\frac{1}{2}\left[\left(y_{j1}-y_{j1}^{d}\right)^{2}+\left(y_{j2}-y_{j2}^{d}\right)^{2}+\left(y_{j3}-y_{j3}^{d}\right)^{2}+\left(y_{k1}-y_{k1}^{d}\right)^{2}+\left(y_{k2}-y_{k2}^{d}\right)^{2}+\left(y_{k3}-y_{k3}^{d}\right)^{2}\right]}. \tag{2.10}$$

In order to calculate the total system error, we generalize the error for all patterns in the database, according to (2.9). If the error is below the minimum desired, then the learning process is finished. Otherwise, the learning process continues in step 5.

(5) In this step, the fuzzy weight adjustments can be computed with two different algorithms.

(a) Maintaining the parameters "a" and "b" from the previous pertinence functions without any change. In this case, fixed partitioning is used.

(b) Using a "Descending Gradient" method to adjust the parameters "a" and "b", which corresponds to the pertinence functions of the antecedent. In this case, an adaptive partitioning is conducted.

(6) Each bipartition is evaluated regarding its contribution to the total rms error, and also in terms of the acceptable minimum error. Each bipartition with an unacceptable error is separated. The evaluation of the error generated for the data set that falls on the partitioning ij, for example, is calculated by (2.11)

$$E_{\text{rms}}^{ij} = \sqrt{\frac{1}{L}\sum_{n=1}^{L}\alpha_i^n\alpha_{ij}^n\sum_{c=1}^{\#\text{classes}}\left(y_{nc}-y_{nc}^d\right)^2}, \tag{2.11}$$

where α_i^n and α_{ij}^n are the rules' firing levels for pattern "n".

(7) In order to prevent the system's structure from growing indefinitely, a dimensionless parameter is created and named decomposition rate (δ). During the learning process, the decomposition rate is constantly compared with the population density of patterns that fall in a specific bipartition. When the population density of patterns (the rate between the number of patterns in a bipartition and the total number of patterns) falls below the decomposition rate, this bipartition cannot be further partitioned, thus limiting the structure. Therefore, a very low value for δ can result in a very big structure, thus compromising the generalization capability. On the other hand, if a large value is chosen, the structure might be too small to learn the patterns with the desired accuracy.

(8) This step performs the decomposition of the separate partitions. For each bipartition a detached decomposition process is performed, consisting of the following process: it allocates a new node (new cell) in the BSP structure for the separate BSP bipartition (it is divided into two). Thus, two new pertinence functions are generated; which will constitute the two newly created partitions. Figure 4 presents a graphic representation of this process.

(9) Go back to step "3" to continue the learning process.

2.4. Strategies for Attributes Selection

In the area of pattern classification, it is important to define the goals and to select, from the available database, which characteristics will be used. These features are the attributes considered relevant in the pursuit of significant goals.

In the case of rock classification, there is a large number of relevant variables available, such as the Hurst coefficient for grayscale and RGB channels, grayscale and RGB channels spatial variation coefficient, in addition to the descriptors used to define texture characteristics obtained in image cooccurrence matrices. In general, we choose the most representative collection of variables, the so-called features.

The correct attribute selection strategy is that which avoids unnecessary partitioning, resulting in more compact BSP tree structures, consequently resulting in better generalization, fewer rules and greater interpretation degrees [2]. In this study, two methods for attribute selection have been tested: the Jang algorithm [7] which showed better performance, and the entropy method [8]. In addition to those methods, there are several studies using other techniques for feature selection, such as principal components analysis [9–12], machine learning [13, 14]; hierarchical clustering [15, 16] and genetic algorithms [17–19].

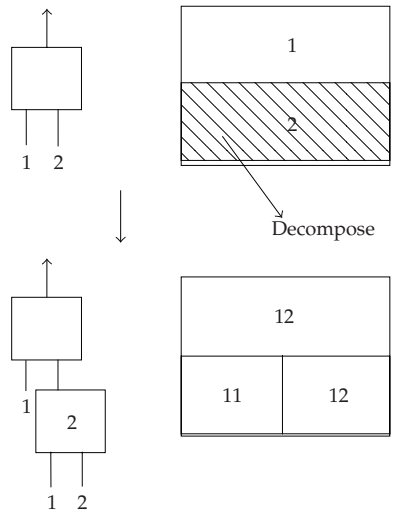

Figure 4: BSP decomposition.

Considering the Jang algorithm, two strategies for selection—fixed and adaptive—have been proposed to deal with the selection problem, that is, which input variables should be applied to partitioning inputs of each NFHB-Class cell.

In the fixed selection sets strategy, where the order of the attributes is determined by Jang algorithm, during the NFHB-Class Method learning process and architecture construction, each of these features is chosen and used as input for each level of the BSP tree. The same input (attribute) is used for all nodes in the same level. This strategy generates unnecessary partitioning due to the fact that all nodes at the same level are forced to use the same fixed input, which is not always the best characteristic for this node. One of the advantages of this strategy is that the computational cost is very small, since the feature selection is performed only once, before the learning process. The result obtained is very competitive. In many cases it results in an interesting alternative, considering time and performance.

In contrast to the methodology described above, the adaptive selection strategy chooses the best input feature (attribute) for each tree node, regardless of the level where the node is. For each node the best input is chosen using only the subset associated with this node. This strategy generates more compact neuro-fuzzy BSP structures according to the specialization of each node, resulting in better generalization performance. However, the computational cost is higher, since the selection algorithm must be run for each new node of NFHB-Class Model.

2.5. Fuzzy Rules Extraction

The NFHB-Class Method is capable of generating interpretable rules with the purpose of extracting information from a specific database. An example of the rules extracted in this model is: "If x is high and y is small and w is hot, then class is k".

Figure 6 shows an example of NFHB-Class method. In this approach, each partition of the input space (leaf node) will have an associated rule. The elements of each partitioning are associated to all existing k classes, with different membership levels.

As demonstrated by Gonçalves et al. [3], the rule extracting process consists of the following steps:

(i) routing on the NFHB-Class tree, calculating the firing level of each rule in each partitioning.

(ii) evaluation of the rules with the use of fuzzy accuracy and coverage.

Fuzzy Accuracy

The accuracy of a rule is a measure of how well it applies to the data. In order to determine how suitable a particular fuzzy rule describes a specific class k, the fuzzy accuracy measurement is presented in (2.12)

$$\text{Fuzzy\_Accuracy}_k^i = \frac{\sum_{j=1}^{P_k} \alpha_{k,j}^i}{\sum_{j=1}^{P_i} \alpha_j^i}, \tag{2.12}$$

where $\text{Fuzzy\_Accuracy}_k^i$ is the rule accuracy for class k in partition i; $\alpha_{k,j}^i$ is the membership level of pattern j for class k in partition i; α_j^i is the membership level of pattern j in partition i (regardless of the class); P_k is the total number of class k patterns; P_i is the total number of patterns in partition i.

If the database has different numbers of patterns per class, the correction factor W_k^i must be applied in order to balance the nonuniform pattern distribution. Hence, the correct fuzzy accuracy $\text{Fuzzy}^*\text{\_Accuracy}_k^i$ is calculated by (2.13)

$$\begin{aligned} \text{Fuzzy}^*\text{\_Accuracy}_k^i &= \text{Fuzzy\_Accuracy}_k^i \cdot W_k^i, \\ W_k^i &= \frac{1}{P_k \sum_{j=1}^{N_c} \left(\text{Fuzzy\_Accuracy}_j^i / P_j \right)}, \end{aligned} \tag{2.13}$$

where is W_k^i is correction factor for accuracy of class k in partition i; N_c is total number of classes; P_j is number of patterns of class j; P_k is number of patterns of class k.

The accuracy sum related to all classes in a specific partition is always equal to 1, since each node's membership function is complementary. Besides, each pattern's level of presence, in each partition, is calculated by the intersection of all firing levels in each node, using the product operator.

Fuzzy Coverage

Fuzzy coverage provides a measure of how comprehensive a rule is in relation to the total number of patterns in the base rule, that is, it measures "how many" patterns are affected by the available rule. The fuzzy coverage definition is given by (2.14)

$$\text{Fuzzy}^i\text{\_Coverage} = \frac{\sum_{j=1}^{P_i} \alpha_j^i}{P}, \tag{2.14}$$

where $\text{Fuzzy}^i\_\text{Coverage}$ = partition i fuzzy coverage; P = total number of patterns in the database; α_j^i is the membership level of the j pattern in the i partition; and P_i is the number of patterns in i partition.

Due to the aforementioned features (complementary membership functions and product operator); all rule composition covers the total number of patterns. In other words, the fuzzy coverage sum of all partitions is equal to 1.

3. Texture

According to Turceyan and Jain [20], image texture is a combination of similar patterns with a regular frequency. It is an attribute that represents a spatial pixel arrangement in a region [21]. Jain [22] defines texture as a basic pattern repetition in space. Conci et al. [23] refer to the texture as a visual pattern that has some homogeneity properties that does not result simply in a color or intensity; it can be defined as the visual appearance of a surface. Although there is no precise definition for texture, it is easily perceived by human vision due to its range of visual patterns composed by subpatterns, which have underlining properties, such as uniformity, density, roughness, regularity, intensity [24].

3.1. Rock Texture—Related Works

Lepistö et al. [25] have shown a classification method based on structural and spectral characteristics of rocks. In order to extract features that could identify the texture, a spectral feature of a set of color parameters was used, and in order to define the structural aspect they used cooccurrence matrices [26]. For rock classification, the nonhomogeneous textures were divided into blocks. Lepistö et al. [27] applied Gabor filters in rock color images for classification. Autio et al. [28] used cooccurrence matrices with Hough transform for rock classification. Starkey and Samantary [29] used morphology and color parameters to distinguish rock images. Genetic programming techniques with edge detectors were also used by Ross et al. [30] and by Starkey and Samantary [29] in petrography. Genetic programming with decision trees was used for grain segmentation by Ross et al. [31]. Thompson et al. [32] used neural networks for mineral classification. Fueten and Manson [33] used neural networks for detecting edges in petrography color images in order to discriminate the grains.

In this work, the following has been used as descriptors of texture: Spatial Variation Coefficient—SVC—[23], Hurst coefficient [34], entropy [35] and cooccurrence matrices [24, 36].

4. Case Study

To evaluate the system performance we used 50 RGB images (401 × 401) for each rock class and subclass, thus producing a total of 800 images. The rock classes and subclasses which compose the image database are: gneiss (two subclasses), basalt (four subclasses), diabase (five subclasses) and rhyolite (five subclasses). The images which have been analysed were of natural rocks, without any kind of treatment (polishing). Figure 5 shows this classification.

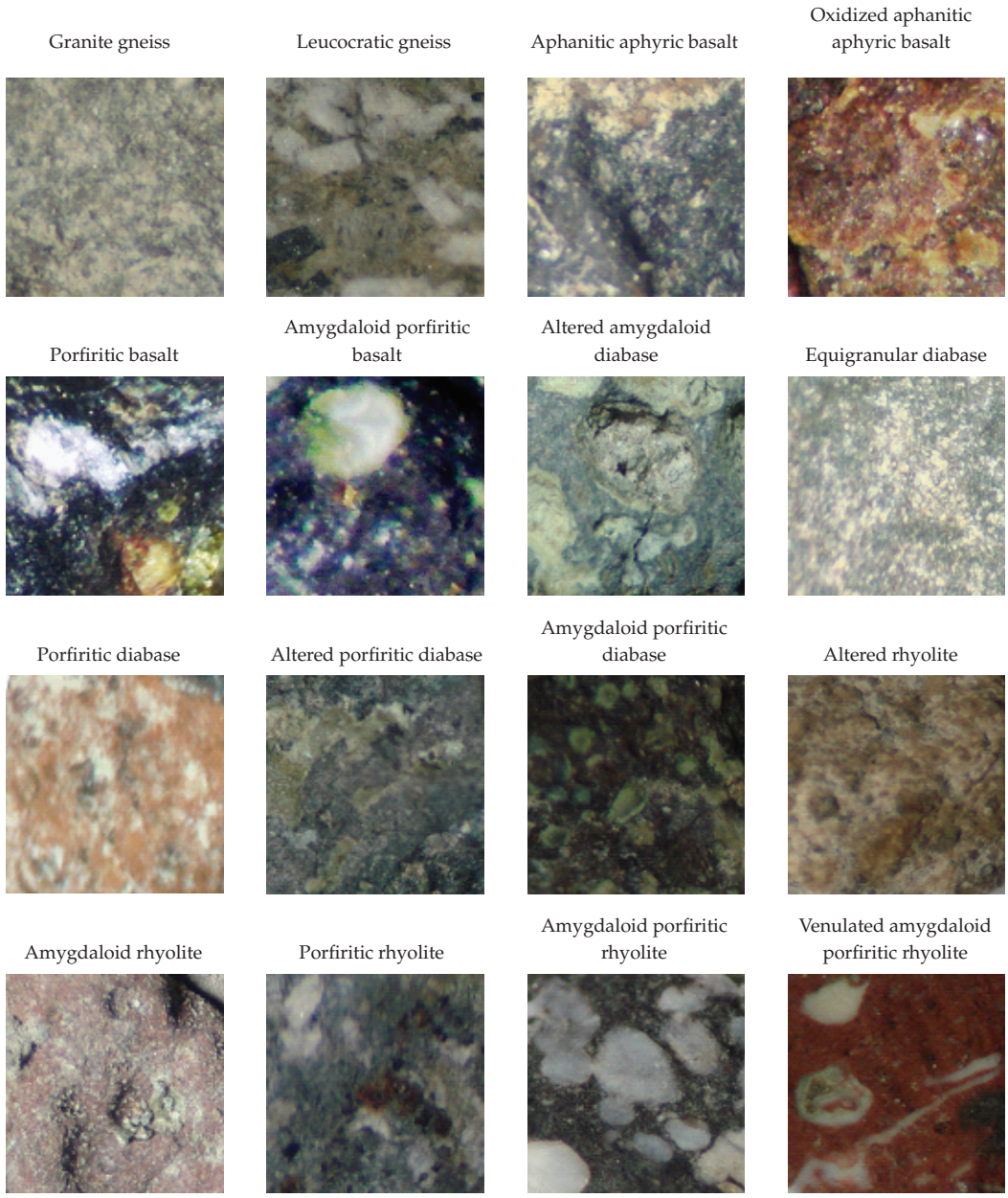

Figure 5: Rocks samples.

From each image we extracted the following: Hurst coefficient for gray and color images (a coefficient for each RGB channel); spatial variation coefficient (gray and color); entropy and cooccurrence matrix. From the 160 cooccurrence matrices, we calculated the average matrices in the 0°, 45°, 90°, and 135° directions for each distance, resulting in 40 matrices for the 40 distances. For the analysis of these 40 matrices, we used the following descriptors: contrast, homogeneity, energy, entropy and correlation. We then generated five curves for each image, and both the highest value and the area were used as attributes to determine the texture of the images. Table 1 shows all the attributes.

Table 1: Attributes used to determine the image texture.

Attribute Number	Description	
1	*Hurst coefficient* for	gray scale
2		Red channel
3		Green channel
4		Blue channel
5	*SVC* for	gray scale
6		Red channel
7		Green channel
8		Blue channel
9	*Entropy of the image* for	gray scale
10		Red Channel
11		Green Channel
12		Blue channel
13	Highest value of the contrast	Descriptor found in the cooccurrence matrix
14	Area of the contrast	
15	Highest value of the correlation	
16	Area of the correlation	
17	Highest value of the energy	
18	Area of the energy	
19	Highest value of the entropy	
20	Value of the distance where was found the highest value of the entropy	
21	Area of the entropy	
22	Highest value of the homogeneity	
23	Area of the homogeneity	

The training models stage is performed in accordance with the following steps.

(1) Initially, only the subclass classification of gneiss, basalt, diabase, and rhyolite rocks is made. With this, the following databases are created:

   (a) a database with just 2 gneiss rock subclasses: granite gneiss and leucocratic gneiss,

   (b) a database with just 4 basalt rock subclasses: aphanitic aphyric basalt, oxidized aphanitic aphyric basalt, porfiritic basalt and amygdaloid porfiritic basalt,

   (c) a database with just 5 diabase rock subclasses: altered amygdaloid diabase, equigranular diabase, porfiritic diabase, altered porfiritic diabase and amygdaloid porfiritic diabase,

   (d) a database with just 5 rhyolite rock subclasses: altered rhyolite, amygdaloid rhyolite, porfiritic rhyolite, amygdaloid porfiritic rhyolite and venulated amygdaloid porfiritic rhyolite.

(2) Then the HNFB-Class Method is tested for the rock macro classification. In this process, we create:

(a) a database with just 2 rock macro classes: gneiss and basalt, without considering subclasses,

(b) a database with just 2 rock macro classes: diabase and rhyolite, without considering subclasses,

(c) a database with just 3 rock macro classes: basalt, diabase and rhyolite, without considering subclasses,

(d) a database with just 3 rock macro classes: gneiss, basalt, diabase, and rhyolite, without considering subclasses.

In all tests, we use 50% of the database to train the HNFB-Class Method and 50% to validate it. Tables 2 and 6 summarize the performance of HNFB-Class Method with selection strategies: fixed (NFHB-Class^1) and adaptive (NFHB-Class^2), for subclass classification and macro classification, respectively.

For subclass classification, the databases are organized as follows:

(i) index 1 (Gneiss1, Basalt1, Diabase1, Rhyolite1): a database with only gray scale coefficients,

(ii) index 2 (Gneiss2, Basalt2, Diabase2, Rhyolite2): a database with only RGB channels coefficients,

(iii) index 3 (Gneiss3, Basalt3, Diabase3, Rhyolite3): a database including RGB and grayscale coefficients,

(iv) index 4 (Gneiss4, Basalt4, Diabase4, Rhyolite4): a database with the attributes extracted from cooccurrence matrices.

For macro classification, the databases are organized as follows:

(i) gneiss and basalt1; diabase and rhyolite1; basalt, diabase and rhyolite1; gneiss, basalt, diabase, and rhyolite1: database containing only gray scale coefficients,

(ii) gneiss and basalt2; diabase and rhyolite2; basalt, diabase, and rhyolite2; gneiss, basalt, diabase, and rhyolite2: database containing only RGB channels coefficients,

(iii) gneiss and basalt3: diabase and rhyolite3; basalt, diabase, and rhyolite3; gneiss, basalt, diabase, and rhyolite3: database including RGB and grayscale coefficients.

When using the attributes extracted from the cooccurrence matrix, poorer results are obtained than with the use of other attributes. Therefore, they have not been used in tests of macro classification of rocks.

With the intent of comparing with NFHB-Class Method, some backpropagation neural networks have been used. Tables 3 and 7 show the configurations used for subclass classification and macro classification, respectively. Tables 4 and 8 present the performance of neural networks for subclass classification and macro classification respectively.

Tables 5 and 9 summarize the performance of NFHB-Class Method with selection strategies: fixed (NFHB-Class^1) and adaptive (NFHB-Class^2), for subclass classification and macro classification, respectively. These tables are based on the largest percentage of the overall set validation for each method.

As observed in Table 5, the best results in rock classification have been obtained with the NFHB-Class Methods. It is also noted that NFHB-Class^2 obtained a lower number of rules than NFHB-Class^1 models.

Table 2: Classification results obtained with the NFHB-Class[1] (fixed selection strategy) and NFHB-Class[2] (adaptive selection strategy) methods, for subclass classification of gneiss, basalt, diabase, and rhyolite rocks.

Model	Database test	Training set correct percentage	Validation set correct percentage	Rules generated (number)	Decomposition Rate	Attributes used order
NFHB-Class[1]	Gneiss1	98%	96%	32	0.03	5, 9, 1
NFHB-Class[2]		98%	96%	17	0.08	
NFHB-Class[1]	Gneiss2	100%	98%	52	0.01	8, 12, 7, 6, 10, 11, 3, 2, 4
NFHB-Class[2]		96%	94%	12	0.1	
NFHB-Class[1]	Gneiss3	94%	96%	52	0.01	8, 12, 7, 5, 6, 10, 9, 11, 3
NFHB-Class[2]		100%	98%	54	0.008	
NFHB-Class[1]	Gneiss4	100%	98%	81	0.01	13, 23, 19, 22, 21, 14, 17, 18, 15, 20, 16
NFHB-Class[2]		98%	96%	9	0.1	
NFHB-Class[1]	Basalt1	79%	59%	73	0.006	1, 5, 9
NFHB-Class[2]		90%	65%	118	0.0009	
NFHB-Class[1]	Basalt2	95%	87%	81	0.006	10, 12, 11, 2, 8, 4, 3, 7, 6
NFHB-Class[2]		88%	84%	25	0.003	
NFHB-Class[1]	Basalt3	79%	83%	96	0.004	2, 8, 9, 12, 11, 6, 5, 3, 1, 10, 7, 4
NFHB-Class[2]		98%	81%	61	0.009	
NFHB-Class[1]	Basalt4	87%	74%	81	0.01	14, 15, 13, 16, 17, 18, 19, 20, 22, 21, 23
NFHB-Class[2]		91%	75%	41	0.006	
NFHB-Class[1]	Diabase1	65.6%	56%	26	0.03	1, 5, 9
NFHB-Class[2]		90.4%	56.8%	87	0.009	
NFHB-Class[1]	Diabase2	81.6%	71.2%	110	0.006	4, 7, 6, 3, 8, 2, 12, 10, 11
NFHB-Class[2]		92%	73.6%	83	0.002	
NFHB-Class[1]	Diabase3	89.6%	70.4%	62	0.003	4, 7, 6, 5, 3, 1, 8, 2, 9, 10, 11, 12
NFHB-Class[2]		85.6%	69.6%	63	0.0003	
NFHB-Class[1]	Diabase4	70.4%	67.2%	64	0.008	19, 21, 16, 17, 22, 14, 15, 23, 18, 20, 13
NFHB-Class[2]		75.2%	67.2%	35	0.009	
NFHB-Class[1]	Rhyolite1	68.8%	58.4%	97	0.008	5, 9, 1
NFHB-Class[2]		88%	60.8%	133	0.005	
NFHB-Class[1]	Rhyolite2	88%	75.2%	68	0.008	7, 8, 6, 11, 10, 4, 12, 2, 3
NFHB-Class[2]		96%	78.4%	56	0.007	
NFHB-Class[1]	Rhyolite3	93.6%	78.4%	225	0.002	7, 8, 5, 6, 11, 10, 4, 9, 12, 2, 3, 1
NFHB-Class[2]		100%	76.8%	108	0.007	
NFHB-Class[1]	Rhyolite4	76%	67.2%	85	0.009	7, 8, 5, 6, 11, 10, 4, 9, 12, 2, 3, 1
NFHB-Class[2]		64.8%	69.6%	17	0.01	

Table 3: Neural network Configurations used to classify the following rock subclasses: gneiss, basalt, diabase, and rhyolite.

Rock	Database test	Number of layers	Number of neurons in input layer	Number of neurons in first hidden layer	Number of neurons in second hidden layer	Number of neurons in output layer
Gneiss	Gneiss1	3	3	5	—	2
	Gneiss2		9	9		
	Gneiss3		12	9		
	Gneiss4		11	9		
Basalt	Basalt1	4	3	10	10	4
	Basalt2		9	10		
	Basalt3		12	10		
	Basalt4		11	10		
Diabase	Diabase1	4	3	20	15	5
	Diabase2		9	20	15	
	Diabase3		12	20	25	
	Diabase4		11	20	15	
Rhyolite	Rhyolite1	4	3	20	15	5
	Rhyolite2		9	20		
	Rhyolite3		12	20		
	Rhyolite4		11	20		

Table 4: Neural networks performance in the classification of the following rock subclassess: gneiss, basalt, diabase, and rhyolite.

Rock	Database test	Correct percentage of training set	Correct percentage of validation set
Gneiss	Gneiss1	96%	92%
	Gneiss2	100%	96%
	Gneiss3	100%	96%
	Gneiss4	100%	96%
Basalt	Basalt1	81%	55%
	Basalt2	100%	86%
	Basalt3	100%	85%
	Basalt4	96%	67%
Diabase	Diabase1	73.6%	51.2%
	Diabase2	93.6%	69.6%
	Diabase3	97.6%	68%
	Diabase4	92%	54.4%
Rhyolite	Rhyolite1	76%	49.6%
	Rhyolite2	97.6%	75.2%
	Rhyolite3	98.4%	64%
	Rhyolite4	88%	59.2%

Table 5: Best performance obtained by methods: NFHB-Class[1] (fixed selection strategy), NFHB-Class[2] (adaptive selection strategy) and neural networks. Rock subclasses: gneiss, basalt, diabase, and rhyolite.

Rock	Model	Best performance for the training set	Best performance for the validation set	Number of generated rules
Gneiss	NFHB-Class[1]	100%	98%	52
	NFHB-Class[2]	100%	98%	12
	Neural network	100%	96%	—
Basalt	NFHB-Class[1]	95%	87%	81
	NFHB-Class[2]	88%	84%	25
	Neural network	100%	86%	—
Diabase	NFHB-Class[1]	81.6%	71.2%	110
	NFHB-Class[2]	92%	73.6%	83
	Neural network	93.6%	69.6%	—
Rhyolite	NFHB-Class[1]	93.6%	78.4%	225
	NFHB-Class[2]	96%	78.4%	56
	Neural network	97.6%	75.2%	—

As exposed in Table 9, the best results in macro rock classification have been observed in the NFHB-Class Methods. The higher the number of patterns and classes, the higher the size of the NFHB-Class become, that is, it is necessary that the model "learns" greater diversities of patterns.

4.1. Fuzzy Rule Extractions

In order to illustrate the tree structure obtained by the NFHB-Class Method, we have selected the test performed with gneiss rock class. We used the database containing only the RGB channels' coefficients as attributes (Gneiss2), using the adaptive feature selection strategy and decomposition rate of $\delta = 0.1$. In this case, the success in the training set was 96% and the validation set was 94%. Figure 6 shows the complete structure presented by the HNFB-Class Method for the gneiss rock. The illustrated Class1 and Class2 represent the subclasses granite gneiss and leucocratic gneiss, respectively.

In Figure 6, the attributes are encoded by: X2—Hurst coefficient for Red channel, X3—Hurst coefficient for Green channel, X4—Hurst coefficient for Blue channel, X6—SVC for Red channel, X7—SVC for Green channel, X8—SVC for Blue channel, X10—Entropy for Red Channel, X11—Entropy for Green Channel, X12—Entropy for Blue channel.

Following the path in the tree, it is possible to extract rules that describe the database of gneiss rock. Table 10 lists the fuzzy rules extracted from the tree structure in Figure 6.

5. Conclusions

This paper presents an NFHB-Class Method for the classification of rocks, constituting an approach which creates its own architecture. Two strategies for feature selection in the database have been adopted; fixed and adaptive. Using the algorithm embedded in the

Table 6: Classification results obtained with NFHB-Class^1 (fixed selection strategy) and NFHB-Class^2 (adaptive selection strategy) methods, for macro rock classification of gneiss and basalt; diabase, and rhyolite; basalt, diabase, and rhyolite; and gneiss, basalt, diabase, and rhyolite.

Model	Database test	Correct percentage of training set	Correct percentage of validation set	Number of rules generated	Decomposition Rate	Order of the attributes used
NFHB-Class^1	Gneiss and basalt1	90.67%	88%	37	0.02	1, 5, 9
NFHB-Class^2		92%	86%	46	0.01	
NFHB-Class^1	Gneiss and basalt2	96%	96%	53	0.007	4, 7, 10, 3, 2, 11, 8, 6, 12
NFHB-Class^2		92.67%	95.33%	22	0.02	
NFHB-Class^1	Gneiss and basalt3	94.67%	90%	68	0.009	4, 7, 10, 2, 11, 5, 3, 1, 9, 8, 6, 12
NFHB-Class^2		98.67%	96.67%	68	0.008	
NFHB-Class^1	Diabase and rhyolite1	70%	68.4%	29	0.02	5, 9, 1
NFHB-Class^2		79.6%	70.4%	83	0.008	
NFHB-Class^1	Diabase and rhyolite2	81.6%	86%	22	0.02	6, 7, 8, 10, 11, 12, 2, 3,4
NFHB-Class^2		87.2%	86%	33	0.01	
NFHB-Class^1	Diabase and rhyolite3	87.2%	85.2%	59	0.009	5, 7, 6, 8, 10, 9, 11, 12, 2, 1, 3, 4
NFHB-Class^2		86%	85.2%	25	0.03	
NFHB-Class^1	Basalt, diabase, and rhyolite1	64.86%	61.43%	34	0.006	5, 9, 1
NFHB-Class^2		70.29%	60.86%	67	0.009	
NFHB-Class^1	Basalt, diabase, and rhyolite2	84%	80.57%	77	0.008	6, 10, 7, 8, 2, 11, 3, 12, 4
NFHB-Class^2		84.29%	80.29%	33	0.005	
NFHB-Class^1	Basalt, diabase, and rhyolite3	84.86%	80.57%	98	0.006	6, 10, 5, 7, 8, 9, 2, 11, 1, 3, 12, 4
NFHB-Class^2		83.14%	78.86%	50	0.006	
NFHB-Class^1	Gneiss, basalt, diabase, and rhyolite1	54%	52.25%	55	0.004	5, 9, 1
NFHB-Class^2		57.25%	56.75%	51	0.003	
NFHB-Class^1	Gneiss, basalt, diabase, and rhyolite2	77%	75.75%	105	0.007	6, 10, 7, 11, 12, 8, 2, 3, 4
NFHB-Class^2		83.75%	77.75%	79	0.006	
NFHB-Class^1	Gneiss, basalt, diabase, and rhyolite3	75.75%	74.25%	92	0.008	6, 10, 5, 7, 8, 9, 2, 11, 1, 3, 12, 4
NFHB-Class^2		79.5%	74.5%	80	0.006	

model of Jang, it is not necessary to use principal component analysis to determine the best combination of attributes which is more representative for the rock textures.

The results obtained with the NFHB-Class Method, in terms of the classification task, achieved more than 73% accuracy in the validation set for all classes of rocks, which indicates its great potential in performing this task.

Table 7: Configurations of neural networks used in the macro classification of gneiss and basalt; diabase, and rhyolite; basalt, diabase, and rhyolite; gneiss, basalt, diabase, and rhyolite

Rock	Database test	Number of layers	Number of neurons in input layer	Number of neurons in first hidden layer	Number of neurons in second hidden layer	Number of neurons in output layer
Gneiss and basalt	Gneiss and basalt1	3	3	6	—	2
	Gneiss and basalt2		9	10		
	Gneiss and basalt3		12	10		
Diabase and rhyolite	Diabase and rhyolite1	4	3	20	25	2
	Diabase and rhyolite2		9	25	20	
	Diabase and rhyolite3		12	20	25	
Basalt, diabase and rhyolite	Basalt, diabase, and rhyolite1	4	3	40	30	3
	Basalt, diabase, and rhyolite2		9			
	Basalt, diabase, and rhyolite3		12			
Gneiss, basalt, diabase, and rhyolite	Gneiss, basalt, diabase, and rhyolite1	4	3	40	40	4
	Gneiss, basalt, diabase, and rhyolite2		9			
	Gneiss, basalt, diabase, and rhyolite3		12			

For the gneiss rock class, the best result in the four databases tested was 98% in the validation set. For the basalt rock class, the result was 87%. For the diabase rock class, the best result for all databases tested was 73.6% in the validation set. For the rhyolite rock class we obtained 78.4%.

One of the advantages of NFHB-Class Method is the fact that it generates fuzzy rules that explain the extraction of knowledge, that is, it is possible to have a very satisfactory rating. Therefore, it presents a explanation for the classification, which does not happen if a neural network or a committee of neural networks are employed in a classification task.

Moreover, another advantage of NFHB-Class Method is the fact that it avoids the task of testing multiple structures in the search of a good performance to the problem. In the case of neural networks, this means empirically determining the best number of hidden layers and processing elements (neurons) for each layer. In the case of fuzzy systems, it means finding the best number of partitions in the universe of discourse for each input variable. In addition, it is important to consider the minimization of problems generated by over and under dimensioning of these structures (overfitting and no-convergence).

Table 8: Performance of neural networks for macro classes of gneiss and basalt; diabase, and rhyolite; basalt, diabase, and rhyolite; gneiss, basalt, diabase, and rhyolite.

Rock	Database test	Correct percentage of training set	Correct percentage of validation set
Gneiss and basalt	Gneiss and basalt1	92.67%	80%
	Gneiss and basalt2	100%	94%
	Gneiss and basalt3	100%	94%
Di*l*base and rhyolite	Diabase and rhyolite1	82.4%	72%
	Diabase and rhyolite2	99.2%	85.2%
	Diabase and rhyolite3	95.2%	84.4%
Basalt, diabase, and rhyolite	Basalt, diabase, and rhyolite1	75.14%	60.85%
	Basalt, diabase, and rhyolite2	97.42%	77.42%
	Basalt, diabase, and rhyolite3	99.42%	77.14%
Gneiss, basalt, diabase, and rhyolite	Gneiss, basalt, diabase, and rhyolite1	65.5%	53.75%
	Gneiss, basalt, diabase, and rhyolite2	97%	77%
	Gneiss, basalt, diabase, and rhyolite3	99.75%	68.75%

Table 9: Best performance obtained by NFHB-Class^1 (fixed selection strategy), NFHB-Class^2 (adaptive selection strategy) and neural networks, for rock classes: gneiss and basalt; diabase, and rhyolite; basalt, diabase, and rhyolite; gneiss, basalt, diabase, and rhyolite.

Rock	Model	Correct percentage of training set	Correct percentage of validation set	Number of generated rules
Gneiss and basalt	NFHB-Class^1	96%	96%	53
	NFHB-Class^2	98.67%	96.67%	68
	Neural network	100%	94%	—
Diabase and rhyolite	NFHB-Class^1	81.6%	86%	22
	NFHB-Class^2	87.2%	86%	33
	Neural network	99.2%	85.2%	—
Basalt, diabase, and rhyolite	NFHB-Class^1	84%	80.57%	77
	NFHB-Class^2	84.29%	80.29%	33
	Neural network	97.42%	77.42%	—
Gneiss, basalt, diabase, and rhyolite	NFHB-Class^1	77%	75.75%	105
	NFHB-Class^2	83.75%	77.75%	79
	Neural network	97%	77%	—

Table 10: Fuzzy rules.

	Rules	Accuracy	Coverage
Rule 1	If X8 is low and X12 is low and X6 is low then Class = 1	0.6813	0.1105
Rule 2	If X8 is low and X12 is low and X6 is high and X3 is low then Class = 1	0.548	0.06137
Rule 3	If X8 is low and X12 is low and X6 is high and X3 is high then Class = 1	0.5152	0.05707
Rule 4	If X8 is low and X12 is high and X10 is low and X2 is low then Class = 1	0.5013	0.03149
Rule 5	If X8 is low and X12 is high and X10 is low and X2 is high then Class = 2	0.5159	0.02537
Rule 6	If X8 is low and X12 is high and X10 is high and X4 is low then Class = 1	0.5801	0.01211
Rule 7	If X8 is low and X12 is high and X10 is high and X4 is high then Class = 2	0.5648	0.01475
Rule 8	If X8 is high and X7 is low then Class = 1	0.5601	0.09951
Rule 9	If X8 is high and X7 is high and X11 is low and X8 is low then Class = 1	0.5388	0.02845
Rule 10	If X8 is high and X7 is high and X11 is low and X8 is high then Class = 2	0.6614	0.03755
Rule 11	If X8 is high and X7 is high and X11 is low and X12 is low then Class = 2	0.5931	0.01464
Rule 12	If X8 is high and X7 is high and X11 is low and X12 is high then Class = 2	0.5676	0.01653

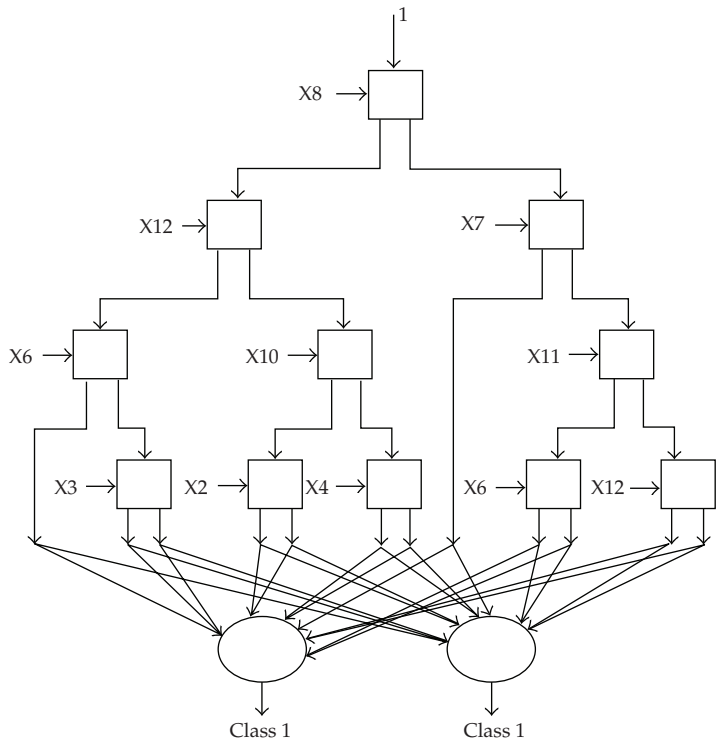

Figure 6: NFHB-Class complete tree structure with adaptive selection strategy for the test performed with gneiss rock.

As a disadvantage, it is important to point to the fact that the learning algorithm of the NFHB-Class Method requires more complex programming than the training algorithm of the neural networks. This is due to the fact that there is no fixed structure or a constant number of adjustable parameters.

Tests with the NFHB-Class Method, converged to optimum solution for the classification taking into account the fuzzy rules extraction, revealed good performance.

One of the proposals for future work is to hybridize the HNFB-Class Method so that it is capable of performing the macro classification, with the subclassification done automatically. Furthermore, executing applications of this model in other areas, such as in the classification of metals, is also intended.

Acknowledgment

The authors would like to acknowledge the financial support provided by FAPERJ (E-26/171.362/2001, E-26/110.992/2008).

References

[1] L. B. Gonçalves, F. R. Leta, and S. C. de Valente, "Macroscopic rock texture image classification using an hierarchical neuro-fuzzy system," in *Proceedings of the 16th International Conference on Systems, Signals and Image Processing (IWSSIP '09)*, pp. 1–5, Chalkida, Greece, 2009.

[2] L. B. Gonçalves, *Modelos neuro-fuzzy hierárquicos BSP para classificação de padrões e extração de regras fuzzy em banco de dados*, M.S. thesis, Electric Engineering Department, PUC-Rio, Rio de Janeiro, Brazil, 2001.

[3] L. B. Gonçalves, M. M. B. R. Vellasco, M. A. C. Pacheco, and F. J. de Souza, "Inverted hierarchical neuro-fuzzy BSP system: a novel neuro-fuzzy model for pattern classification and rule extraction in databases," *IEEE Transactions on Systems, Man and Cybernetics Part C*, vol. 36, no. 2, pp. 236–248, 2006.

[4] F. J. de Souza, *Modelos neuro-fuzzy hierárquicos*, M.S. thesis, Electric Engineering Department, PUC-Rio, Rio de Janeiro, Brazil, 1999.

[5] M. B. Barret, T. Chan, and J. Demmel, "Templates for the solution of linear systems: building blocks for iterative methods," 1994, http://www.netlib.org/.

[6] R. R. Yager and D. P. Filev, "Template-based fuzzy systems modeling," *Journal of Intelligent and Fuzzy Systems*, vol. 2, pp. 39–54, 1994.

[7] J. S. R. Jang, "Structure determination in fuzzy modeling: a fuzzy cart approach," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, vol. 1, pp. 480–485, Orlando, Fla, USA, 1994.

[8] A. I. Lanas, *Sistemas neuro-fuzzy hierárquicos BSP para previsão e extração de regras fuzzy em aplicações de mineração de dados*, M.S. thesis, Electric Engineering Department, PUC-Rio, Rio de Janeiro, Brazil, 2000.

[9] A. Aoyama and S. P. K. Walsh, "Nonlinear modelling of LPG %C5 content of catalytic reformer debutaniser column," *Computers & Chemical Engineering*, vol. 21, supplement 1, pp. S1155–S1160, 1997.

[10] D. Dong and T. J. Mcavoy, "Nonlinear principal component analysis—based on principal curves and neural networks," *Computers and Chemical Engineering*, vol. 20, no. 1, pp. 65–78, 1996.

[11] J. J. Roffel, J. F. MacGregor, and T. W. Hoffman, "The design and implementation of a multivariable internal model controller for a continuous polybutadiene polymerization train," in *Proceedings of the IFAC Symposium on Dynamics and Control of Chemical Reactors, Distillation Columns, and Batch Processes (DYCORD '89)*, pp. 9–15, Pergamon Press, Maastricht, The Netherlands, August 1989.

[12] A. Santen, G. L. M. Koot, and L. C. Zullo, "Statistical data analysis of a chemical plant," *Computers & Chemical Engineering*, vol. 21, supplement 1, pp. S1123–S1129, 1997.

[13] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 245–271, 1997.

[14] P. Langley, "Selection of relevant features in machine learning," in *Proceedings of the AAAI Fall Symposium on Relevance*, pp. 140–144, AAAI Press, November 1994.

[15] M. Dash, H. Liu, and J. Yao, "Dimensionality reduction of unsupervised data," in *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI '97)*, pp. 532–539, 1997.

[16] L. Talavera, "Feature selection as retrospective pruning in hierarchical clustering," in *Proceedings of the 3rd International Symposium on Intelligent Data Analysis*, Springer, Amsterdam, The Netherlands, 1999.

[17] W. Punch, E. Goodman, M. Pei, L. Chia-Shum, P. Hovland, and R. Enbody, "Further research on feature selection and classification using genetic algorithms," in *Proceedings of the International Conference on Genetic Algorithms*, pp. 557–564, Springer, 1993.
[18] W. Siedlecki and J. Sklansky, "A note on genetic algorithms for large-scale feature selection," *Pattern Recognition Letters*, vol. 10, no. 5, pp. 335–347, 1989.
[19] J. Yang and V. Honavar, "Feature subset selection using genetic algorithm," *IEEE Intelligent Systems and Their Applications*, vol. 13, no. 2, pp. 44–48, 1998.
[20] M. Turceyan and A. K. Jain, "Handbook of Pattern Recognition and Computer Vision," in *Texture Analysis*, pp. 235–276, World Scientific, River Edge, NJ, USA, 1993.
[21] IEEE Standard 610.4, *IEEE Standard Glossary of Image Processing and Pattern Recognition Terminology*, IEEE Press, New York, NY, USA, 1990.
[22] A. K. Jain, *Fundamentals of Image Processing*, Prentice Hall, New York, NY, USA, 1998.
[23] A. Conci, E. Azevedo, and F. R. Leta, *Computação Gráfica—Teoria e Prátic*, vol. 2, Elsevier, New York, NY, USA, 2008.
[24] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, no. 6, pp. 610–621, 1973.
[25] L. Lepistö, I. Kunttu, J. Autio, and A. Visa, "Rock image classification using non-homogenous textures and spectral imaging," in *Proceedings of the 11th International Conference in Central Europe on Computer Graphics, Visualization and Computer (WSCG '03)*, pp. 82–86, Plzen, Czech Republic, February 2003.
[26] P. Launeau, C. A. Cruden, and J.-L. Bouchez, "Mineral recognition in digital images of rocks: a new approach using multichannel classification," *Canadian Mineralogist*, vol. 32, no. 4, pp. 919–933, 1994.
[27] L. Lepistö, I. Kunttu, and A. Visa, "Rock image classification using color features in Gabor space," *Journal of Electronic Imaging*, vol. 14, no. 4, Article ID 040503, 3 pages, 2005.
[28] J. Autio, S. Luukkanen, L. Rantanen, and A. Visa, "The classification and characterisation of rock using texture analysis by co-occurrence matrices and the hough transform," in *Proceedings of the International Symposium on Imaging Applications in Geology*, pp. 5–8, University of Liege, Liege, Belgium, May 1999.
[29] J. Starkey and A. K. Samantaray, "Edge detection in petrographic images," *Journal of Microscopy*, vol. 172, no. 3, pp. 263–266, 1993.
[30] B. J. Ross, F. Fueten, and D. Y. Yashkir, "Edge detection of petrographic images using genetic programming," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '09)*, 2000.
[31] B. J. Ross, F. Fueten, and D. Y. Yashkir, "Automatic mineral identification using genetic programming," *Machine Vision and Applications*, vol. 13, no. 2, pp. 61–69, 2001.
[32] S. Thompson, F. Fueten, and D. Bockus, "Mineral identification using artificial neural networks and the rotating polarizer stage," *Computers & Geosciences*, vol. 27, no. 9, pp. 1081–1089, 2001.
[33] F. Fueten and J. Mason, "An artificial neural net assisted approach to editing edges in petrographic images collected with the rotating polarizer stage," *Computers & Geosciences*, vol. 33, no. 9, pp. 1176–1188, 2007.
[34] J. R. Parker, *Algorithms for Image Processing and Computer Vision*, John Wiley & Sons, Toronto, Canada, 1997.
[35] L. He, Q. Pan, W. Di, and Y. Li, "Anomaly detection in hyperspectral imagery based on maximum entropy and nonparametric estimation," *Pattern Recognition Letters*, vol. 29, no. 9, pp. 1392–1403, 2008.
[36] D. Morquin, M. B. Ghalia, and S. Bose, "An integrated neural network-based vision system for automated separation of clods from agricultural produce," *Engineering Applications of Artificial Intelligence*, vol. 16, no. 1, pp. 45–55, 2003.