Unsupervised and Neural Hybrid Techniques for Audio Signal Classification

Andrés Ortiz, Lorenzo J. Tardón, Ana M. Barbancho and Isabel Barbancho
Dept. Ingeniería de Comunicaciones, ETSI Telecomunicación-University of Malaga, Campus Universitario de Teatinos s/n, Malaga Spain

1. Introduction

Audio signal analysis and classification have arisen as an important research topic that has been developed by many authors in different areas over the time. A main development context has been speech recognition Holmes & Huckvale (1994); Juang & Rabiner (2005); Kimura (1999). This specific topic is, in fact, an important source of references to applications of artificial intelligence techniques Juang & Rabiner (2005); Prasad & Prasanna (2008). Many classical problems encountered in this research field have been addressed from this perspective by many authors Farahani & Ahadi (2005); Minematsu et al. (2006).

However, in this same context, a different point of view can be adopted to deal with the analysis and classification of music signals. The tasks in this framework are varied, including the detection of pitch, tempo or rhythm Thornburg et al. (2007), but also other tasks like the identification of musical instruments Müller et al. (2011) or musical genre recognition Tzanetakis & Cook (2002) can be considered. Also the classification of audio samples as music or speech has been thoroughly considered in the literature Panagiotakis & Tziritas (2005), Tardón et al. (2010).

In this specific context of analysis of musical signals, we will expose some ideas regarding signal classification and their application to a real task.

2. Models and applications

Audio signal classification involves the extraction of a number of descriptive features from the sound and the proper utilization of them as input for a classifier. Artificial Intelligence (AI) techniques provide a way to deal with signal processing and pattern classification tasks from a different point of view of classical techniques.

AI techniques play an important role in the signal processing and classification context as they have been widely used by a number of authors in the literature for very different tasks Haykin (1999); Kohonen et al. (1996); Ortiz, Górriz, Ramírez & Salas-Gonzalez (2011a); Ortiz, Gorriz, Ramirez & Salas-Gonzalez (2011b); Ortiz, Ortega, Diaz & Prieto (2011); Riveiro et al. (2008). Also, we must be aware of the fact that the current trend in artificial intelligence systems points to the utilization of both classical and AI-based methods in conjunction to improve the overall system performance. This leads to build hybrid artificial intelligence systems. These systems can include neural-based techniques, evolutive computation, or statistical classifiers as well as other statistical techniques such as multivariate or stochastic methods.

Through this chapter, we will present the problem of classification of musical signals. The problem will be defined to be handled in a supervised or unsupervised way. It is be important to point out that a preprocessing stage to determine the features to be used for the classification task must be done although the classifiers can be able to properly deal with the different discrimination capability of the different features. Afterwards, a classifier will assign a different label related to the features employed to any different sound class in a process performed in a supervised or unsupervised way.

Taking into account that unsupervised classification strategies are able to organize training samples into suitable groups according to certain classes without using any a priori information (the samples to be classified are not labelled), we will describe how to apply these ideas in this context. We will present features related to the analysis framework and samples of the performance.

Unsupervised techniques include the Self-Organizing Maps (SOM) Kohonen (2001), a vector quantization method with a competitive learning process. SOMs provide a generalization of the input space through the prototypes computed during the training phase. In this model, each prototype represents a set of input vectors on the basis of a given similarity measurement. Very often, the similarity measure selected is the Euclidean distance. Moreover, it will be specially interesting to observe that SOMs group the prototypes maintaining the more similar ones close to each other in the output space while the less similar prototypes ones are kept apart. In this way, SOMs provide valuable topological information that can be exploited to make them more flexible than other vector quantization methods. This means that a description of a SOM model for a certain application must be complemented by additional details. Specifically, we are referring to two different aspects: the first one consists on the modelling of the activation process of the neurons of the SOM which can be done by means of statistical techniques such as Gaussian Mixture Models (GMM) to account for a probabilistic behaviour of the map. The second aspect to consider is related to the techniques that allow the extraction of valuable topological information from the SOM; this process can be accomplished by classical clustering techniques such as k-means or other techniques Therrien (1989) specially developed for clustering the SOM.

On the other hand, supervised classification Murtagh (1991) techniques use a priori information of at least some of the samples. It means that the training samples have labels according to their corresponding class. Classical methods include different clustering techniques or statistical classifiers. We will introduce some labels and related features for classification and describe the utilization of them to attain certain classification objectives.

3. Audio signal features for classification: Clarinet music example

In this section we describe five features extracted from audio signal. These features will compose the input data used in the unsupervised classifiers shown in the following sections. The examples shown aims to detect four different playing techniques from clarinet music. The features extracted from the audio signal attempts to characterize the signal in both, time and frequency domains in a simple way, being effective enough to perform classification experiments.

Classification examples have been performed using three different clarinets from real recordings taken from the Musical Instrument Sound Data Base RWC-MDB-1-2001-W08 Goto (2004). Clarinet 1 is a french clarinet made by Buffet, Clarinet 2 is a french clarinet made by Selmer and Clarinet 3 is a japanese clarinet many by Yamaha. For each clarinet, we have 120 note samples that contain the whole clarinet note range played with different dynamics for each playing technique (Normal, Staccato, Vibrato and Trill). This gives a total of 1440 note samples.

3.1 Time domain characterization

In this Section, we describe the features used to characterize the audio signal in time domain. The duration and the shape of the envelope of the clarinet audio signals contain information about the technique used to play the notes.

Thus, as in Barbancho et al. (2009); Jenses (1999), the attack time (T_a) is considered from the first sample that reaches a 10% of the maximum of the amplitude of the waveform of the note until it reaches the 90% of that amplitude. The release time (T_r) is considered from the last sample that reaches 70% of the maximum of the amplitude of the waveform to the last one over the 10% of the amplitude. On the other hand, the time between the attack time and the release time is called the sustain time T_s. T_a, T_r and T_s depend on the playing technique.

The signal envelope is obtained by filtering the signal with a 5th order Butterworth filter with cut-off frequency of 66 Hz. After the application of the low-pass filter, the signal is normalized so that the amplitude is 1. On the other hand, the samples with amplitude under 2.5% of the maximum are removed.

Additionally, we include another time domain feature T_f based on the presence of signal fading (i.e.: if the signal envelope is fading, $T_f = 1$, otherwise $T_s = 0$).

3.2 Frequency domain characterization

In order to characterize the clarinet in the frequency domain, we use the *Fast Fourier Transform* (FFT). In the frequency domain, the frequency axis is converted into MIDI numbers according to the following equation:

$$MIDI = 69 + 12 \log_2(f/440) \tag{1}$$

Taking into account that the fundamental frequency of the notes of the clarinet ranges from 146.83 Hz to 1975 Hz, the MIDI range of interest is 50 to 94.

However, it is necessary to remove redundant information from the FFT in order to simplify the signal spectrum. Thus, for a certain MIDI number n_{MIDI} the spectrum between $n_{MIDI} - 0.5$ and $n_{MIDI} + 0.5$ is considered and the maximum value of the spectrum in that interval is assigned to n_{MIDI}. Thus, the MIDI number spectrum (or MIDI simplified spectrum) of each note will have 45 samples. From this simplified spectrum, the pitch and the spectrum width around the pitch will be calculated. At this point, we have three time domain features (T_a, T_r, T_s and T_f) and two frequency domain features, F_p and F_w. F_p is the pitch of the played note, and F_w is the spectral width of the fundamental frequency defined and calculated as the number of significative samples around the fundamental frequency. We consider as significant samples those over 30% of the value of the fundamental frequency. Thus, the feature space is composed by six-dimensional vectors in the form (T_a, T_r, T_s, T_f, F_p, F_w). These six features may be discriminant enough to characterize the playing technique.

3.3 Playing technique characterization

The time and frequency domain features previously described can be used for characterizing different playing techniques from clarinet music samples. Thus, we will recognize four playing techniques as *normal (NO)*, *staccato (ST)*, *vibrato (VI)* and *trill (TL)*.

Normal playing technique presents a short attack time (T_a) (see Fig. 1). Figure 1, presents the audio waveform, the envelope and the simplified spectrum of an $A4$ played normal. This note corresponds to the middle range of the notes played with a violin. There not special characteristics if notes with higher or lower pitches are played. The shortness of the attack time, and the lengths of the sustain time (T_s) and the release time (T_r) (the time it takes the sound to die when the air pressure, that maintains the clarinetist, finishes) can be observed.

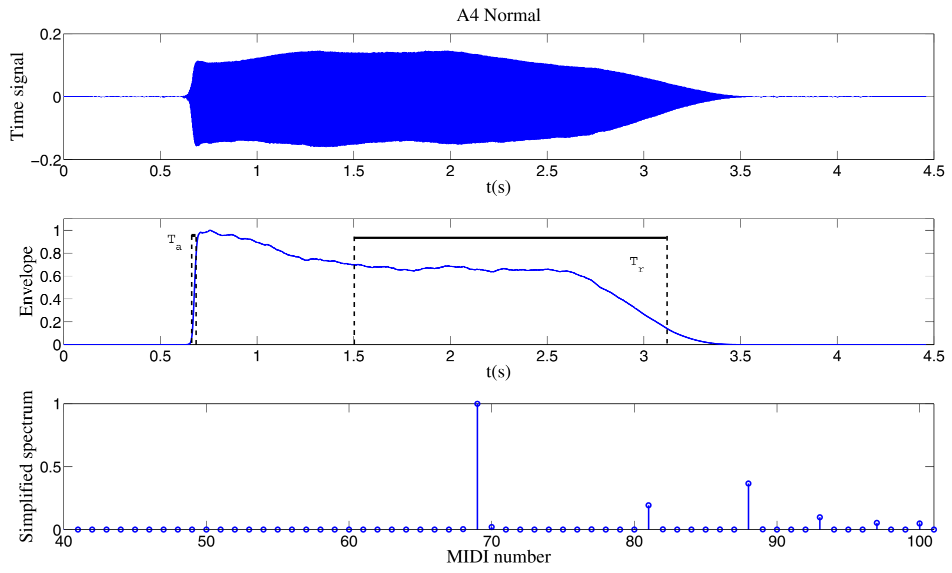

Fig. 1. Waveform, envelope and simplified spectrum of an A4 played normal.

In *staccato*, the duration of the note is reduced to a half, that is, the main characteristic of a note played in staccato will be its short duration. In this technique the clarinettist, produces an constant air pressure, but his tongue will be in the reed to avoid any vibration. When the clarinettist removes the tongue from the reed, a fast attack is get and just after put again his tongue in the reed, so the sustain time and the release time are reduced. So, in this technique, the attack, sustain and release times are going to be reduced. Figure 2, presents the audio waveform, the envelope and the simplified spectrum of an $G4$ played in staccato.

Vibrato stands for a slight fluctuation of the amplitude and the pitch of the notes. This technique can be an undesirable effect when it is produced unwittingly due to the nervousness of the clarinettist. Vibrato is a difficult playing technique and it requires certain level of expertise. In this technique, the clarinettist produces a periodic variation of the air pressure by means of contractions of the diaphragm or lip pressure over the reed. Consequently, when this technique is used, the envelope shows periodic oscillations. Figure 3, shows the envelopes of $E3$ and $D6$ played in normal mode and in vibrato and Figure 4, shows the simplified spectrum of $E3$ and $D6$ played in normal mode and in vibrato.

Trill is a quavering or vibratory sound, specially a rapid alternation of sung or played notes. This technique consists on changing between one note and the following one very quickly by moving fingers. So, it should be expected that the envelope and the spectrum present notable

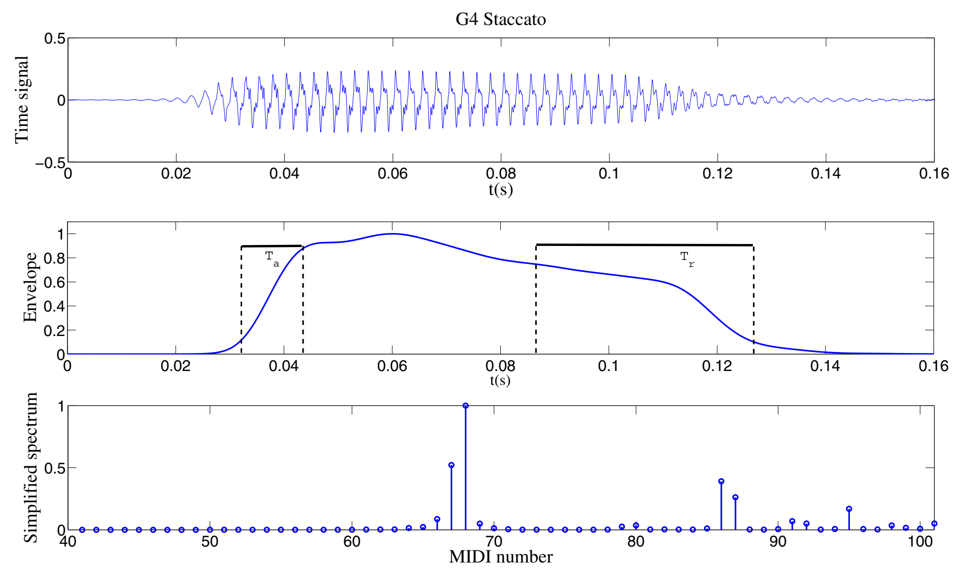

Fig. 2. Waveform, envelope and simplified spectrum of an *G4* played staccato.

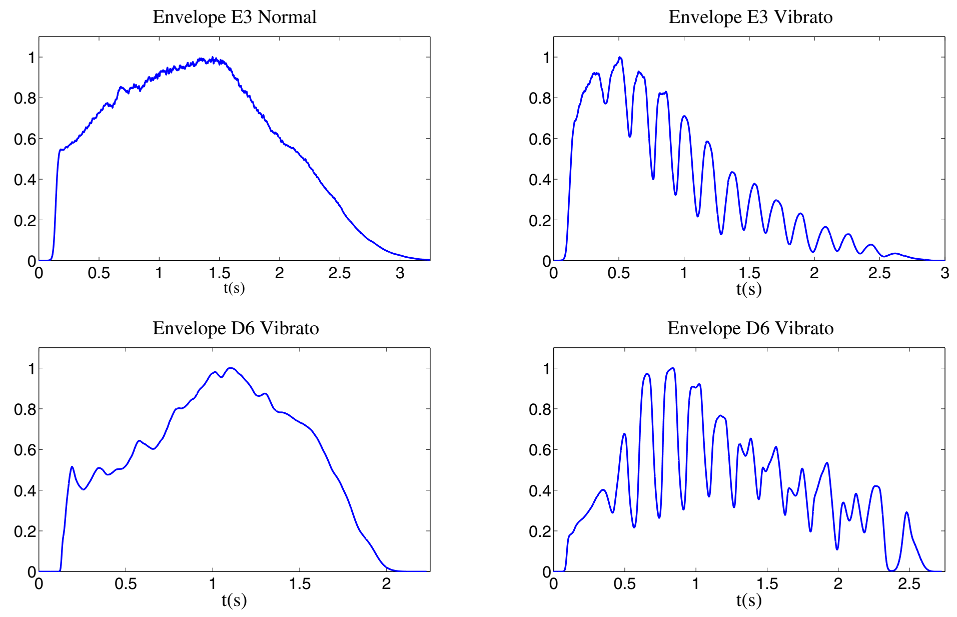

Fig. 3. Envelopes of *E3* and *D6* played in normal mode and in vibrato.

differences with respect to the ones found when the other playing techniques are employed. Figure 5, presents the time signal, the envelope and the simplified spectrum of an *G5* played with trill.

4. Classification with Self-Organizing Maps

The Self-Organizing Map (SOM) is one of the most commonly used artificial neural network models for unsupervised learning. This model, proposed by Kohonen Kohonen (2001), is a biologically inspired algorithm based on the search for the most economic representation of data and its relationships, as in the animal brain.

Sensory experience consists in capturing features from the surrounding world, and these features usually are multidimensional. For instance, the human visual system captures

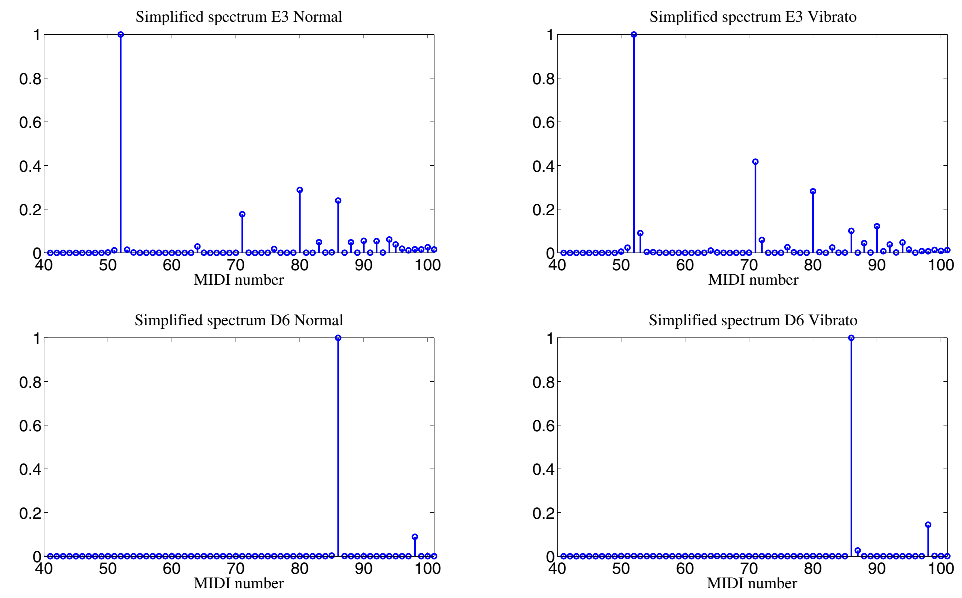

Fig. 4. Simplified spectrum of *E*3 and *D*6 played in normal mode and in vibrato.

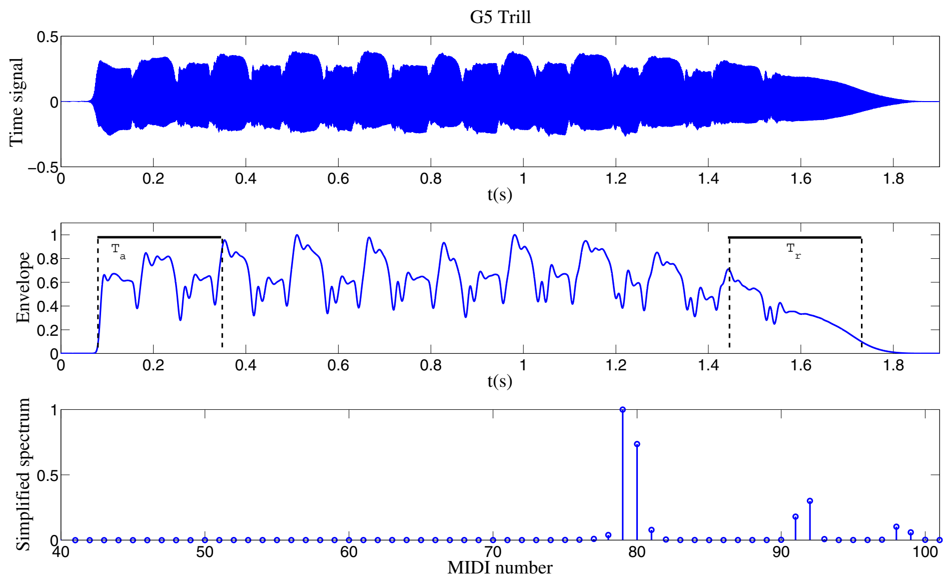

Fig. 5. Waveform, envelope and simplified spectrum of an *G*5 played with trill.

features from different objects such as colour, texture, size or shape, which will provide enough information to ensure further recognition of that object. The learning process in the human brain projects the extracted features onto the neural structures of the brain to creating different maps corresponding to different types of features. Moreover, the learning process stores the prototypes of the features in the maps created. These maps are continuously modified as the learning process progresses. Prototypes could be considered as the smallest feature set needed to be able to represent the sensory information acquired. In other words, prototypes represent generalizations of the learnt features, being possible to distinguish between different objects, and associate similar ones. Moreover, sensory pieces of information coming from different organs are topologically ordered in the brain cortex (topology preservation in brain mapping).

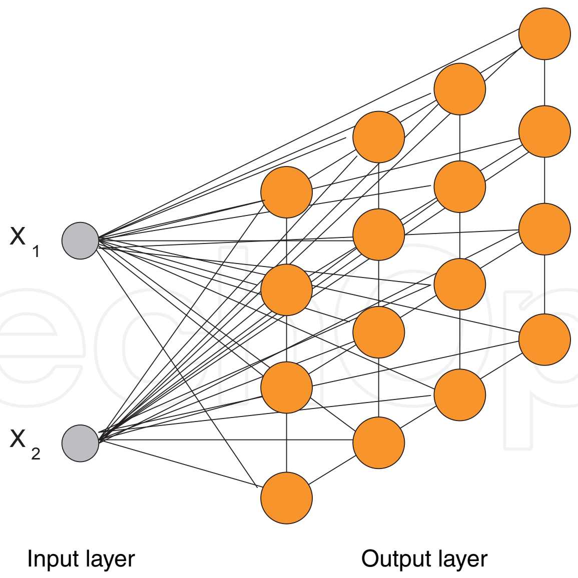

Fig. 6. Self-Organizing Map architecture.

Thus, as in the brain learning process, the main purpose of SOMs is to group the similar data instances close into a two or three dimensional lattice (output map), keeping apart the different ones. Moreover, as the difference between data instances increases, the distance in the output map also increases.

SOMs consist of a number or neurons, also called units, which are arranged following a previously determined 2D or 3D lattice, and each unit stores a multidimensional prototype. Thus, topology is preserved as in the human brain, which is a unique feature of the SOM. Figure 6 shows the SOM architecture in which the output layer is arranged in a 2D lattice.

Units on the SOM are self organized during the training phase. During this stage, the distance between any input vector and the weights associated to the units on the output map are calculated. Usually, the Euclidean distance is used as shown in Equation 2:

$$U_{\omega(t)} = \underset{i}{\operatorname{argmin}} \parallel x(t) - \omega_i(t) \parallel \tag{2}$$

where $x(t) \in X$, is the input vector at time t and $\omega_i(t)$ is the prototype vector associated to the unit i.

The unit closest to the input vector $U_\omega(t)$ is referred to as winning unit and the associated prototype is updated. To complete the adaptive learning process of the SOM, the prototypes of the units in the neighbourhood of the winning unit are also updated according to equation 3 Then, the unit nearest the input vector is referred to as winning unit and the associated weight is updated. Moreover, the weights of the units in the neighbourhood of the winning unit are also updated according to Equation 3:

$$\omega_i(t+1) = \omega_i(t) + \alpha(t) h_{U_i}(t) \big(x(t) - \omega_i(t)\big) \tag{3}$$

where $\alpha(t)$ is the exponential decay learning factor and $h_{U_i(t)}$ is the neighborhood function associated to the unit i. Both, the learning factor and the neighborhood function decay with time, thus the prototypes adaptation becomes slower as the neighborhood of the unit i contains less number of units. This is a competitive process in which the winning neuron each iteration is called Best Matching Unit (BMU).

The neighbour function defines the shape of the neighbourhood and it is usually a Gaussian function which shrinks at each iteration, as defined by the next equations:

$$h_{U_i}(t) = e^{-\frac{\|r_U - r_i\|^2}{2\sigma^2(t)}} \tag{4}$$

$$\sigma(t) = \sigma_0 e^{\frac{-t}{\tau_1}} \tag{5}$$

In Equation 4, r_i represents the position on the output space (2D or 3D) and $\| r_U - r_i \|$ is the distance between the winning unit and the i-neuron on the output space. On the other hand, $\sigma(t)$ controls the reduction of the Gaussian neighborhood at each iteration. $\sigma(t)$ usually takes the form of exponential decay function as in Equation 5.

Similarly, the learning factor in Equation 3, also diminishes in time. However, α may decay in a linear or exponential fashion. Therefore, during the training phase, the prototypes associated to each unit are computed at each iteration. At the same time, the position of the units in the output space changes according to the similarity among the actual prototypes.

As previously commented, topology preservation is a unique feature of the SOM related to the goodness of the clustering process performed during training. This way, the calculation of the quality of the output map, results essential to evaluate the overall process.

In order to perform a quantitative evaluation of the goodness of the SOM, two measures can be accomplished. The first one is the quantization error, which is a measure of the resolution of the map. This quantization error can be calculated by computing the average distance between all the BMUs and the input data vectors as shown in Equation 6:

$$qe_i = \sum_{x_j \in c:i} \| \omega_i - x_j \| \tag{6}$$

The second one measures the topographic error, i.e.: how the SOM preserves the topology. This error can be computed using Equation 7:

$$t_e = \frac{1}{N} \sum_{i=1}^{N} u(\overrightarrow{x_i}) \tag{7}$$

In this equation, N is the total number of input vectors. $u(\overrightarrow{x_i})$ is 1 if the first and the second BMU for the input vector $\overrightarrow{x_i}$ are adjacent units and 0 otherwise Arsuaga & Díaz (2005); Kohonen (2001). Then, the lower q_e and t_e, the better the SOM is adapted to the input patterns.

5. SOM clustering

In order to deal with fully unsupervised classification using SOMs, it is necessary to compute the clusters brought up during the training phase.

This clustering process consist on grouping the prototypes in different classes according to a similarity criterion. This similarity criterion depends on the clustering technique used. Thus, several clustering approaches which use the Euclidean distance as similarity criterion have been developed Murtagh (1995); Rossi & Villa (2009); Vesanto et al. (2000); Wu & Chow (2004). In this chapter, we deal with SOM clustering using two different approaches.

5.1 SOM clustering with K-means

k-means algorithm MacQueen (1967) is a well-known and widely used clustering method due to its performance and simplicity. *k-means* aims to create k partitions from n observations in such a way that each observation will belong to the cluster with the nearest mean to that observation. In other words, the algorithm computes the centroids of each class at each iteration, and computes the euclidean distance to each observation. Then, the nearest observations to each centroid are considered as belonging to the centroid class.

Thus, given a $n - dimensional$ data collection $(x_1 ... x_n)$, *k-means* creates $k, k < n$ classes, while minimizing the *mean squared error* of the euclidean distance between each data point and the corresponding cluster centroid, as shown in equation 8:

$$\underset{C}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x_j \in m_i} \| x_j - s_i \|^2 \tag{8}$$

where m_i represents the mean (centroid) of all the points belonging to the class i.

Although k-means constitutes a usual way to cluster the SOMs in an unsupervised manner, it presents two main drawbacks:

- The number of clusters to be found has to be determined in advance. In other words, it is necessary to know beforehand the value of k.
- k-means only uses the Euclidean distance between prototypes, and does not take into account the relationship between the output and the data topology to compute the clusters. This underutilizes the knowledge available at the output layer of the SOM given by data topology and data distribution Tasdemir & Merenyi (2009); Tasdemir et al. (2011).

Hence, to deal with fully unsupervised classification using SOMs, supervised clustering techniques have to be used at the output layer. In this sense, the CONN clustering method Tasdemir et al. (2011), which is based on SOM visualization of a weighted Delaunay graph, not only performs unsupervised clustering but also takes into account data distribution and topology.

5.2 CONN clustering

CONN clustering computes clusters on the SOM output layer using a new similarity metric for the prototypes Tasdemir & Merenyi (2009). This similarity measure is based on the receptive field of each unit, instead of using the Euclidean distance, unlike other clustering algorithms. Moreover, this technique does not need to know the number of clusters to be found, as in k-means. CONN clustering computes the number of clusters during the clustering process.

As described in Section 4, each SOM unit has an associated prototype ω_i. Each of these units is the centroid of its voronoi polyhedron containing the receptive field of unit i, as described in equation 9:

$$RF_i = \{x_k \in X : \| x_k - \omega_i \| \leq \| x_k - \omega_j \forall j \in S\} \tag{9}$$

where X is the data manifold and S is the set of units on the SOM layer.

Then, it is possible to define the connectivity strength between two prototypes ω_i and ω_j as data vectors for which ω_i and ω_j are the BMU and second BMU Tasdemir & Merenyi (2009);

Tasdemir et al. (2011):

$$RF_{ij} = \{x_k \in RF_i / \parallel x_k - \omega_j \parallel \leq \parallel x_k - \omega_l \parallel, \forall l \neq i\} \tag{10}$$

Then, the connectivity strength matrix $CONN(i,j)$ is created as $CONN(i,j) = |RF_{ij} - RFji|$ where each element indicates the connectivity between the prototypes ω_i and ω_j. In this way, if $CONN(i,j) = 0$, it indicates that ω_i and ω_j are not connected.

Thus, it is possible to determine not only the similar prototypes which are included in a cluster but also the number of clusters with a fully unsupervised clustering technique. In addition, this clustering technique exploits the relationship between the data topology and the SOM layer in a hierarchical agglomerative clustering method.

Figure 7a shows the labels assigned to each SOM unit when audio signal features are extracted from clarinet music, and Figure 7b shows the clustering of the output layer using k-means, computed using the *SOM toolbox* Vesanto et al. (2000). As commented above, it is necessary to know the number of clusters (k) beforehand to run the k-means algorithm. Thus, it is possible to compute the number of clusters that attain the best performance using the k-means algorithm, using a measure of the validity of the clustering.

There are several metrics to evaluate the validity of the clustering process, such as the Davies-Boulding index Davies & Bouldin (1979), the Generalized Dunn index (GDI) Bezdek & Pal (1998), PBM index Hassar & Bensaid (1999) or the silhouette width criterion Kaufman & Rosseauw (1990). These validity indexes provide lower or higher values as the clustering improves depending on the specific algorithm. Particularly, DBI provides lower values for better clustering results. This index is defined by the next equation:

$$DBI = \frac{1}{K} \sum_{k=1}^{K} \max_{i \neq j} \left(\frac{S_K Q_i + S_K Q_j}{S(Q_i, Q_j)} \right) \tag{11}$$

where K is the number of clusters, S_K is the average distance of all objects from the cluster to their cluster centre and $S(Q_i, Q_j)$ the distance between clusters centroids.

Then after several runs of the k-means algorithm the DBI has been computed and represented in Figure 7c. According to this Figure, $k = 4$ leads to the best clustering scheme since it attains the minimum DBI.

On the other hand, Figure 8 shows the clustering found using the CONN clustering algorithm described above.

6. SOM modelling

In Section 4, the SOM learning model was described. In that model, the unit corresponding to the prototype which is closest to the data instance is activated (BMU). This imposes with a binary response of each unit, since each unit is activated or deactivated but intermediate states are not considered.

A variant of the SOM consist on measuring the response of the map units instead of calculating the BMU as the unit which is closest to the input data. This is related to a probabilistic view of the output layer. In order to provide the SOM with this probabilistic behaviour, a Gaussian Mixture Model (GMM) is built over the output layer Kohonen (2001). Thus, the

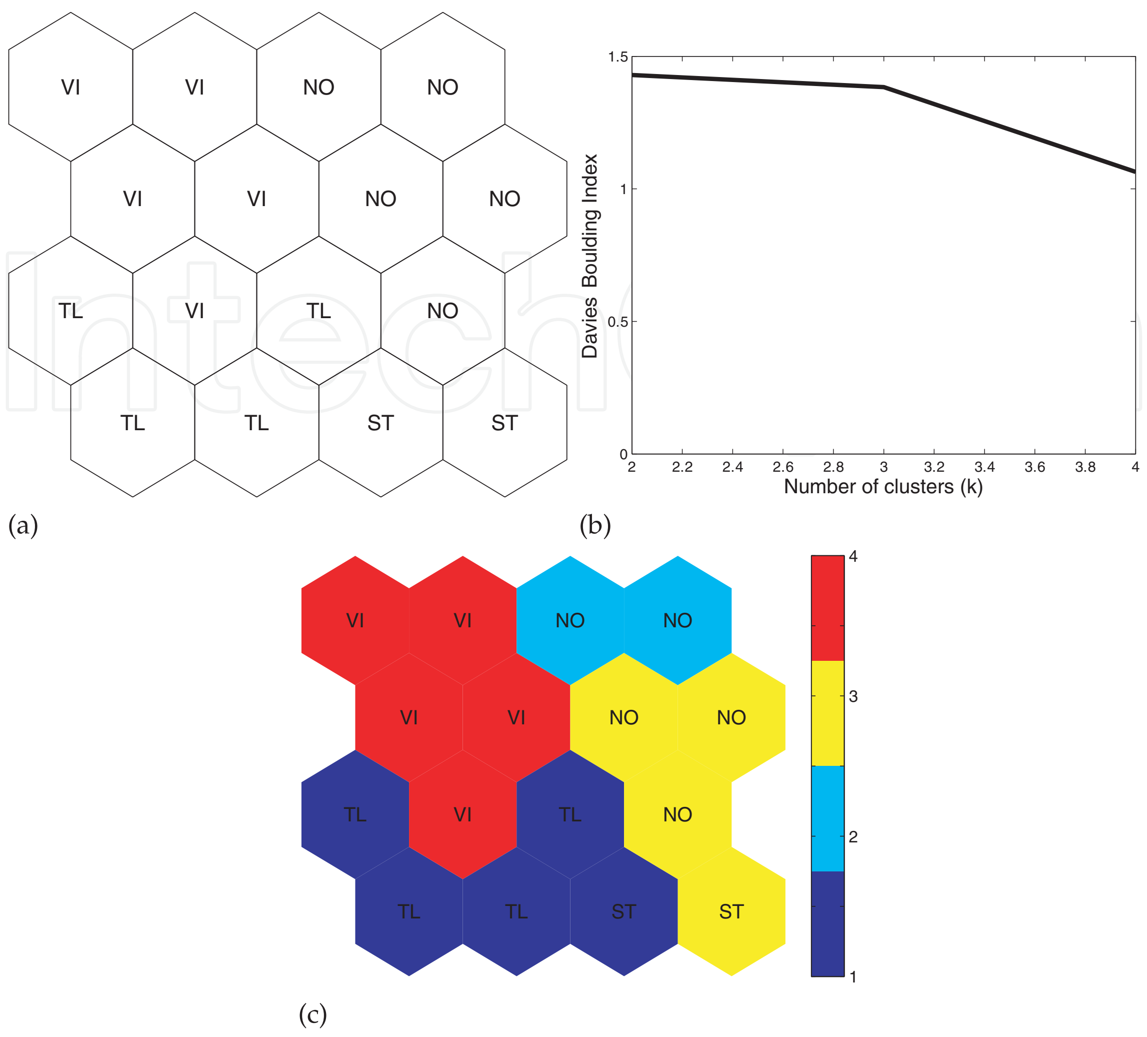

Fig. 7. (a) Unit labels, (b) Davies-Boulding index for different values of *k* (c) Clustering result for minimum DBI.

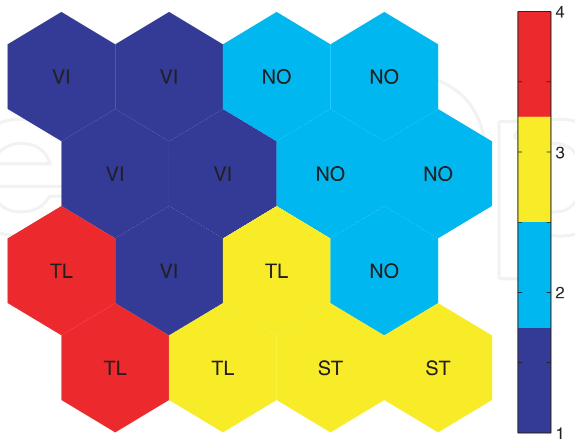

Fig. 8. SOM clustering using the CONN Tasdemir et al. (2011) algorithm.

BMU is determined computing not only the minimum distance from an input vector but also taking into account the likelihood of a unit of being the BMU. This way,the responses of the units surrounding the BMU can be taken into account.

Thus, the prior probability of each map unit i is computed in a similar way as in Alhoniemi et al. (1999), as shown in equation 12:

$$p(i) = \frac{\#\widetilde{X_i}}{\#\widetilde{X}} \tag{12}$$

where $\#\widetilde{X}$ is the total number of input vectors and $\#\widetilde{X_i}$ is the number of vectors whose closest prototype is ω_i. More specifically, $\#\widetilde{X_i}$ is the number of sample vectors found by equation 13:

$$\widetilde{X_i} = \{x \in V \ / \ \|x - m_i\| \leq \|x - m_k\| \ k = 1, ...N\} \tag{13}$$

Thus, $\#\widetilde{X_i}$ can be defined as the set of data samples whose first BMU is the unit i (Voronoi set of unit i).

The GMM is built according to the equation 14:

$$P(x_1, ..., x_n) = \sum_N p_i P_i(x_1, ..., x_n) \tag{14}$$

where the weights p_i for each Gaussian component correspond to the prior probabilities computed in equation 12. In 14, each individual Gaussian component P_i corresponds to the $n - dimensional$ weights associated to each unit (prototype vectors) Alhoniemi et al. (1999); Riveiro et al. (2008). The mean of each individual Gaussian component (kernel centre) is the weight vector of the corresponding unit itself, while the covariance matrix for the i-component is given by the dispersion of the data samples around the prototype i.

Once the GMM model has been built, the response of the unit i can be computed as the posterior probability by using the Bayes theorem.

$$p(\omega_k|x) = \frac{p(x|\omega_k)P(\omega_k)}{p(x)} \tag{15}$$

In equation 15, $p(\omega_k|x)$ represents the probability that a sample vector x belongs to class ω_k. $p(x|\omega_k)$ is the probability density function of the prototype ω_k computed from the GMM and $p(x)$ is a normalization constant. This way, this posterior probability can be used to classify new samples.

Figure 9 shows the mixing proportions of each component in the GMM that correspond to each unit on the SOM. Thus, peaks and valleys can be used to identify th units to be activated with largest probability.

In this way, SOM modelling provides a framework to include a probabilistic behaviour to the SOM, making it possible to modify the activation likelihood of each unit by means of the mixing proportions.

On the other hand, 6-dimensional SOM prototypes can be projected into a 2 or 3 dimensional space using PCA and storing the 2 or 3 principal components with the largest eigenvalues, respectively. Then, also the reduced prototypes can be modelled using a GMM. Figure 10a shows the clusters computed by the projected prototypes considering a 2-dimensional GMM. Similarly, Figure 10b shows the clusters computed using a 3-dimensional GMM.

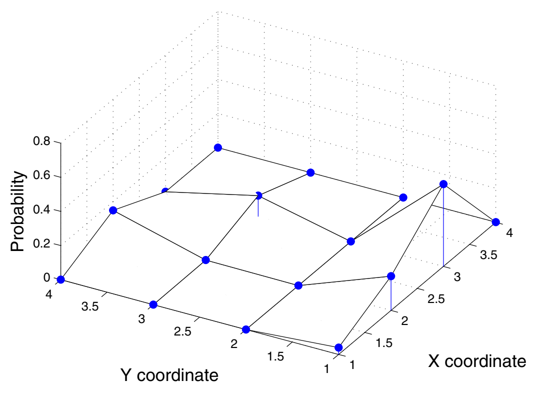

Fig. 9. Activation probability of each SOM unit.

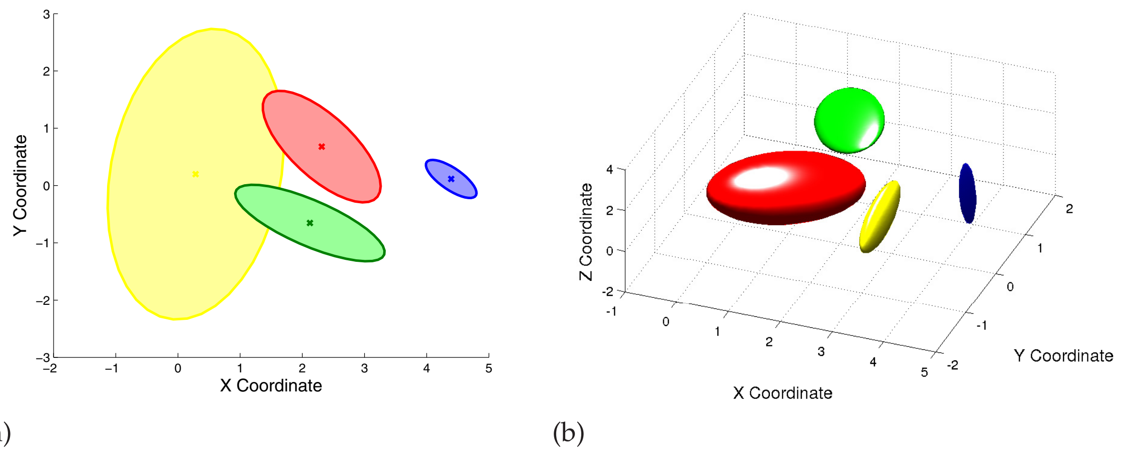

Fig. 10. GMM modelling of the projected prototyped into (a) 2D space and (b) 3D space.

6.1 SOM labeling

Once the map is trained, a label can be assigned to each unit in order to identify the receptive field of this unit. This labelling process can be addressed in two main ways:

1. Labelling with majority voting scheme Vesanto & Alhoniemi (2000); Vesanto et al. (2000). Each unit is labelled with the most frequent label on its 1-neighbourhood. In this case, the size of the neighbourhood for labelling is fixed to radius 1. This process is performed once after the SOM is trained and the assigned labels are used in further classification tasks.
2. Dynamic labelling Cruz et al. (2012); Ortiz, Górriz, Ramírez & Salas-Gonzalez (2011a); Ortiz, Gorriz, Ramirez & Salas-Gonzalez (2011b); Ortiz, Ortega, Diaz & Prieto (2011). As commented in Section 6, the response of the SOM units computed from the posterior probabilities are used to label those units which remain unlabelled during the training process. This way, when a new sample arrives, the BMU with the maximum *a posteriori* probability is selected. However, this BMU could be unlabelled if the map is big enough. In that case, the label of this unit is computed taking into account the response of the units in the neighbourhood of the BMU. Hence, the label assigned to the BMU will be the label of the unit in the neighbourhood $\mathcal{L}_p$ which provides the strongest response at the unlabelled BMU, as shown in equation 16:

$$\mathcal{L}_p = \mathcal{L}\{argmax_i\{p(\omega_i|x)\ \forall i\ \in \mathcal{B}_N\}\} \tag{16}$$

This leads to a dynamic method which labels the units according to the response strength of the neighbourhood units.

Figure 11a shows *a posteriori* probabilities associated to each unit in the 1-neighbourhood of the BMU, which is currently unlabelled. As indicated in the figure legend, darker color correspond to larger *a posteriori* activation probability. Then, the label of the unit in the neighbourhood with the largest activation probability is assigned to the BMU, as depicted in Figure 11b.

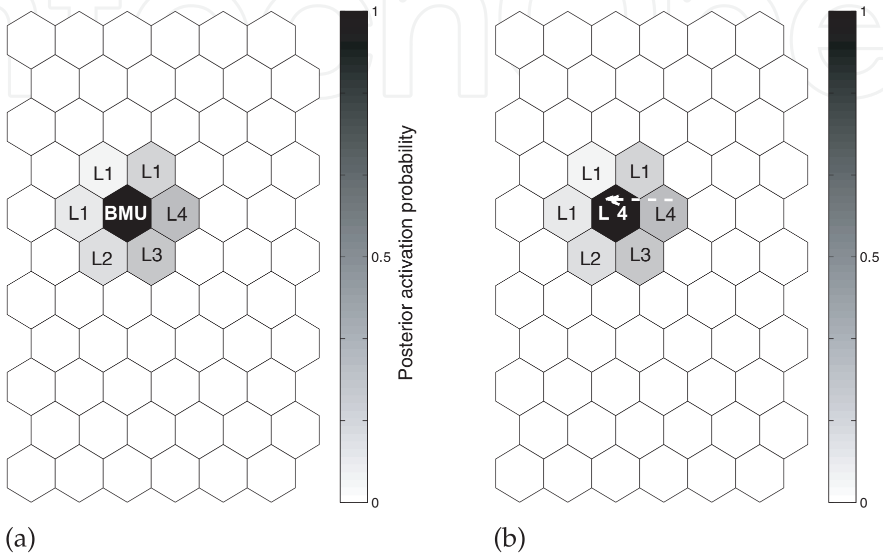
L1
L1
L1
BMU
L4
L2
L3
1
0.5
0
Posterior activation probability
(a)
L1
L1
L1
L 4
L4
L2
L3
1
0.5
0
Posterior activation probability
(b)

Fig. 11. Probabilistic labelling procedure Cruz et al. (2012).

7. SOM modelling and clustering using ICA

SOM modelling has been addressed by means of Gaussian Mixture Models. This assumes Gaussian distributions on SOM prototypes. An alternative for SOM modelling consist in using Independent Component Analysis (ICA) Mixture Models Ghahramani & Beal (2000); Lee et al. (2000); Tipping & Bishop (1999). This deals with an unsupervised classification of SOM prototypes by modelling them as a mixture of several independent classes. Each of these classes are described by linear combinations of independent non-Gaussian densities Lee et al. (2000). Thus, in the same way the previously model uses multivariate Gaussian to model the SOM prototypes, ICA mixture models can be used to estimate the probability of each data instance to be generated by a non-Gaussian structure. Let $X = \{x_1, ..., x_n\}$ be the data set generated by a mixture model. The likelihood of the data can be expressed in terms of the component densities in a similar way that in Equation 14, but in this case, each component has a non-gaussian density derived from the ICA model. Then, the data belonging to each class are described as

$$x_t = A_k s_k \tag{17}$$

where A_k is the mixing matrix in ICA and s_k is the source vector.

Equation 17 assumes that the individual sources are independent, and each class was generated from 17 using a different mixing matrix A_k. This way, it is possible to classify the input data and to compute the probability of each class for each data point. In other word, this allows calculating the probability of a data instance to be generated from an independent component. Nevertheless, as each SOM unit is considered as a kernel center, gaussian components are usually more suitable to provide a probabilistic measurement of the SOM units activation.

8. Growing Hierarchical Self-Organizing Map

The main drawback of SOMs is the size of the map, which has to be selected beforehand. In addition, the performance of the classification process depends on the size of the SOM. Additionally, the performance of SOMs with highly dimensional input data highly depends on the specific features and the calculation of the clusters borders may be not optimally defined. Taking into account these issues, the Growing Hierarchical Self-organizing Map (GHSOM) Dittenbach et al. (2000); Rauber et al. (2002) arises as a convenient variant of SOMs. GHSOMs dynamically grow to overcomes the limitations of the SOMs and to discover inherent hierarchies on the data.

GHSOM is a hierarchical and non-fixed structure developed to overcome the main limitations of classical SOM Rauber et al. (2002). GHSOM structure (shown in Figure 12 consists of multiple layers in which each layer is composed by several independent SOM maps. Hence, during the training process, the number of the SOM maps on each layer and the size of each of these SOMs is determined. This constitutes an adaptive growing process in horizontal and vertical ways. The growing process is controlled by two parameters that control the depth of the hierarchy and the breadth of each map. Therefore, these two parameters are the ones which have to be determined beforehand.

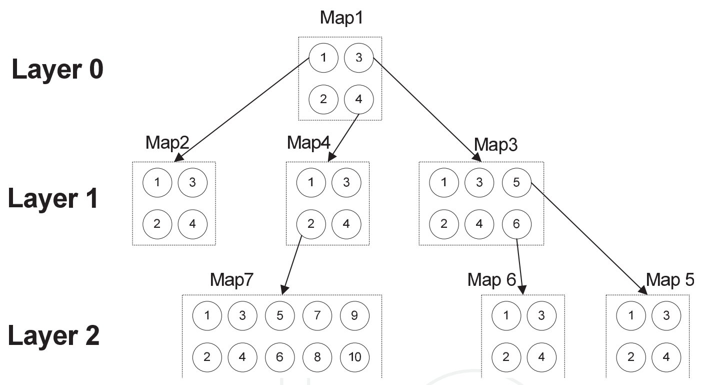

Fig. 12. GHSOM growing example.

In order to determine the limit of the growth of the GHSOM, the quantization error of each unit is calculated according to the next equation:

$$qe_i = \sum_{x_j \in c_i} \| \omega_i - x_j \| \tag{18}$$

where C_i is the set of input vectors mapped into the unit i, x_j is the j-th input vector belonging to C_i and is the weight associated to the unit i.

Initially, all the input vectors are considered to belong to C_0. This means that the whole available input data are used to compute the initial quantization error, qe_0. Then, the quantization errors qe_i for each neuron are calculated. Thus, if $qe_i < \tau_2 \times qe_0$, then the neuron i is expanded in a new map on the next level of the hierarchy. Each new map is trained as an independent SOM and the BMU calculation is performed as shown in Equation 3 by using the Euclidean distance metric. Once the new map is trained, the quantization error of each neuron on this map is computed as

$$q_i = \sum_{x_j \in C_i} \| \omega_i - x_j \| \tag{19}$$

(where q_i is the quantization error of the unit u on the upper layer). Then, the mean quantization error MQE_m of the new map is computed. If $MQE_m \leq \tau_1 q_i$ the map stops growing.

All the growing process is depicted in Figure 12. If τ_1 and τ_2 are selected in such a way that the GHSOM is sightly oversized, some of the units on the GHSOM maps may remain unlabelled after training.

Classification using GHSOM can be accomplished in two ways:

1. Using the clusters computed by GHSOM. These clusters correspond to the receptive field of each map on the lower level of the hierarchy. In this case, since labelling information has not been used, it is possible to distinguish different data instances because the BMU of these instances will be in the corresponding map. However, the identification of a specific playing technique is not possible. This is shown in Figure 13.
2. Using the labelled data for GHSOM training. In this case, although the clustering process is still competitive, the units will contain a label that identifies the playing technique represented by its receptive field.

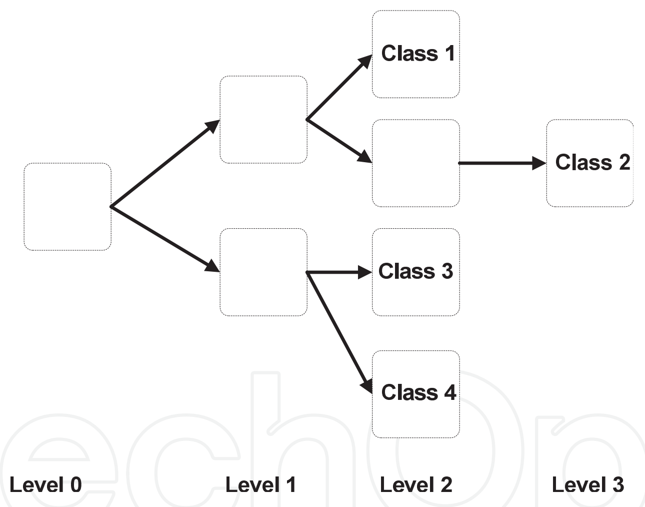

Fig. 13. GHSOM clustering.

Also, BMU calculation can be performed in GHSOMs in a similar way as in classical SOMs by simply following the hierarchy level until the deepest map is found.

8.1 BMU calculation on GHSOM

Once the GHSOM is trained, the BMU is computed for every data sample.

In the case of SOM, the BMU is calculated in the same way it was found during the training phase. Since several SOM layers have been created during the GHSOM training, we have to follow the whole SOM hierarchy in order to determine the winning unit and the map it belongs to.

To this end, an iterative algorithm, depicted in Figure 14, has been developed. In this figure, an example of BMU calculation on a three-level GHSOM is represented. Assume that the distances between an input pattern and the weight vectors of the level 0 map are calculated, then compute the minimum of these distances. As a result, the winning neuron on map 1 is found. Since other maps could be grown from this winning neuron, we have to check if the wining neuron is a parent unit. This test can be carried out making use of the parent vectors resulting from the GHSOM training process. If a new map arose from the wining neuron, the BMU on this map is calculated. This process is repeated until a BMU with no growing map is found. Thus, the BMU in the GHSOM is associated to a map in a level of the hierarchy.

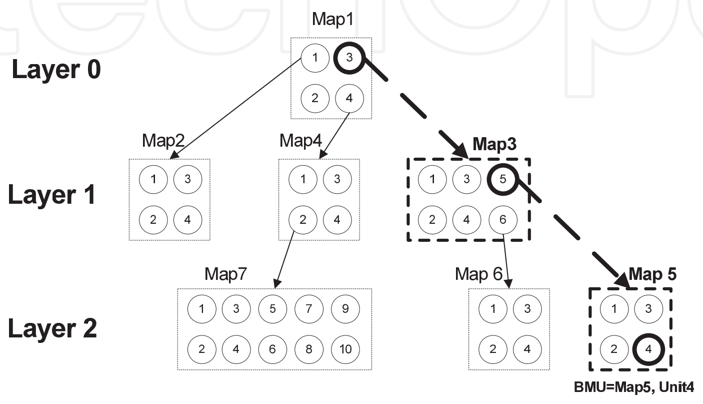

Fig. 14. GHSOM BMU calculation example.

At his point, a probability-based relabelling method can be applied using a 2D Gaussian kernel centered at each BMU Ortiz, Górriz, Ramírez & Salas-Gonzalez (2011a); Ortiz, Gorriz, Ramirez & Salas-Gonzalez (2011b); Ortiz, Ortega, Diaz & Prieto (2011). Thus, a mayority-voting scheme with the units inside the Gaussian kernel is used to relabel the unlabelled units, assigning the calculated label to the data samples as shown in Figure 15.

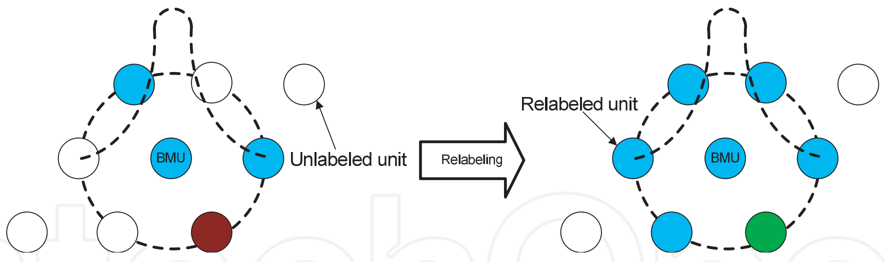

Fig. 15. GHSOM map relabeling method.

In Equation 20, the Gaussian kernel used to estimate the label for unlabelled units is shown:

$$L(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{20}$$

In this equation, σ determines the width of the Gaussian kernel. In other words, it defines the neighbourhood considered for the relabelling process. On the other hand, (x, y) is the position of the BMU in the SOM grid.

9. Concluding summary

In this chapter, we have presented the utilization of artificial intelligence techniques in the context of musical signal analysis.

We have described some features related to a specific classification problem and, then, we have focused on the utilization of neural models for the development of specific tasks in musical audio classification.

The classification task analysed in the context of the application of self-organizing maps and some of its variants which make use of the competitive learning paradigm to implement unsupervised classification techniques applied to audio signals.

As it has been shown in this chapter, these performance of these models can be improved by means of hybridizing with other clustering and dimension reduction techniques like Principal Component Analysis methodologies.

Also, the application of Gaussian Mixture Models provides a probabilistic behaviour of the maps instead of a binary activation of the neurons.

10. Acknowledgments

This work was supported by the Ministerio de Economía y Competitividad of the Spanish Government under Project No. TIN2010-21089-C03-02 and Project No. IPT-2011-0885-430000.

11. References

Alhoniemi, E., Himberg, J. & Vesanto, J. (1999). Probabilistic measures for responses of self-organizing map units alhoniemi, *Proceedings of the International ICSC Congress on Computational Intelligence Methods and Applications (CIMA'99)*.

Arsuaga, E. & Díaz, F. (2005). Topology preservation in som, *International Journal of Mathematical and Computer Sciences* 1(1): 19–22.

Barbancho, I., de la Bandera, C., Barbancho, A. M. & Tardon, L. J. (2009). Transcription and expressiveness detection system for violin music, *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing ICASSP 2009*, pp. 189–192.

Bezdek, J. C. & Pal, N. R. (1998). Some new indexes of cluster validity, 28(3): 301–315.

Cruz, R., Ortiz, A., Barbancho, A. & Barbancho, I. (2012). Unsupervised classification of audio signals, *7th International Conference on Hybrid Artificial Intelligence (HAIS2012)*.

Davies, D. L. & Bouldin, D. W. (1979). A cluster separation measure, (2): 224–227.

Dittenbach, M., Merkl, D. & Rauber, A. (2000). The growing hierarchical self-organizing map, *Proc. IEEE-INNS-ENNS Int Neural Networks IJCNN 2000 Joint Conf*, Vol. 6, pp. 15–19.

Farahani, G. & Ahadi, S. M. (2005). Robust features for noisy speech recognition based on filtering and spectral peaks in autocorrelation domain, *Proc. of the European Signal Processing Conference*, Antalya (Turkey).

Ghahramani, Z. & Beal, M. (2000). Variational inference for bayesian mixtures of factor analysers, *Advances in Neural Information Processing Systems* 12: 449–455.

Goto, M. (2004). Development of the rwc music database, *Proceedings of the 18th International Congress on Acoustics*.

Hassar, H. & Bensaid, A. (1999). Validation of fuzzy and crisp c-partitions, *Proc. NAFIPS Fuzzy Information Processing Society 18th Int. Conf. of the North American*, pp. 342–346.

Haykin, S. (1999). *Neural Networks: a comprehensive foundation*, 2nd edn, Prentice-Hall.

Holmes, W. J. & Huckvale, M. (1994). Why have HMMs been so successful for automatic speech recognition and how might they be improved?, *Speech, Hearing and Language, UCL Work in Progress*, Vol. 8, pp. 207–219.

Jenses, J. (1999). Envelope model of isolated musical sounds, *Proceedings of the 2nd COST G-6 workshop on digital audio effects, Trondheim (Norway).*

Juang, B. H. & Rabiner, L. R. (2005). Automatic speech recognition – a brief history of the technology, *in* K. Brown (ed.), *Encyclopedia of Language and Linguistics*, Elsevier.

Kaufman, L. & Rosseauw, P. (1990). *Finding groups in data*, Wiley, New York.

Kimura, S. (1999). Advances in speech recognition technologies, *Fujitsu Sci. Tech. J.* 35(2): 202–211.

Kohonen, T. (2001). *Self-Organizing Maps*, Springer.

Kohonen, T., Oja, E., Simula, O., Visa, A. & Kangas, J. (1996). Engineering applications of the self-organizing map, *Proc. of the IEEE*.

Lee, T., Lewicki, M. & Sejnowski, T. (2000). Ica mixture models for unsupervised classification of non-gaussian classes and automatic context switching in blind signal separation, *IEEE Transactions on Patt* 22(10): 1078–1089.

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations, *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press. pp. 281–297.*

Minematsu, N., Nishimura, T., Murakami, T. & Hirose, K. (2006). Speech recognition only with suprasegmental features - hearing speech as music, *Proc. of the International Conference on Speech Prosody*, Dresden (Germany).

Müller, M., Ellis, D. P. W., Klapuri, A. & Richard, G. (2011). Signal processing for music analysis, *IEEE Journal on Selected Topics in Signal Processing* 5(6): 1088–1110.

Murtagh, F. (1991). Multilayer perceptrons for classification and regression, *Neurocomputing* 2(5–6): 183–197.

Murtagh, F. (1995). Interpreting the kohonen self-organizing map using contiguity-constrained clustering, *Pattern Recognition Letters* 16(4): 399–408.

Ortiz, A., Górriz, J., Ramírez, J. & Salas-Gonzalez, D. (2011a). Mr brain image segmentation by growing hierarchical som and probability clustering, *Electronics Letters* 47(10): 585–586.

Ortiz, A., Gorriz, J., Ramirez, J. & Salas-Gonzalez, D. (2011b). Mri brain image segmentation with supervised som and probability-based clustering method, *Lecture Notes in Computer Science, LNCS-6686*, pp. 49–58.

Ortiz, A., Ortega, J., Diaz, A. & Prieto, A. (2011). Network intrusion prevention by using hierarchical self-organizing maps and probability-based labeling, *Proceedings of the 11th international conference on Artificial neural networks conference on Advances in computational intelligence*.

Panagiotakis, C. & Tziritas, G. (2005). A speech/music discriminator based on rms and zero-crossings, *IEEE Transaction on Multimedia* pp. 155–166.

Prasad, B. & Prasanna, S. R. M. (eds) (2008). *Speech, Audio, Image and Biomedical Signal Processing using Neural Networks*, Vol. 83 of *Studies in Computational Intelligence*, Springer.

Rauber, A., Merkl, D. & Dittenbach, M. (2002). The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data, 13(6): 1331–1341.

Riveiro, M., Johansson, F., Falkman, G. & Ziemke, T. (2008). Supporting maritime situation awareness using self organizing maps and gaussian mixture models, *Proceedings of the 2008 conference on Tenth Scandinavian Conference on Artificial Intelligence: SCAI 2008*.

Rossi, F. & Villa, N. (2009). Topologically ordered graph clustering via deterministic annealing, *Proceesdings of the 17th European Symposium on Artificial Neural Networks*.

Tardón, L. J., Sammartino, S. & Barbancho, I. (2010). Design of an efficient music-speech discriminator, *J. Acoustical Society of America* 127(1): 271–279.

Tasdemir, K. & Merenyi, E. (2009). Exploiting data topology in visualization and clustering of self-organizing maps, 20(4): 549–562.

Tasdemir, K., Milenov, P. & Tapsall, B. (2011). Topology-based hierarchical clustering of self-organizing maps, 22(3): 474–485.

Therrien, C. (1989). *Decision estimation and classification: an introduction to pattern recognition and related topics*, John Wiley & Sons, Inc.

Thornburg, H., Leistikow, R. & Berger, J. (2007). Melody extraction and musical onset detection via probabilistic models of framewise stft peak data, *IEEE Transactions on Audio, Speech, and Language Processing* 15(4): 1257–1272.

Tipping, M. & Bishop, C. (1999). Mixtures of probabilistic preservationincipal component analyzers, *Neural Computation* 11(2): 443–482.

Tzanetakis, G. & Cook, P. R. (2002). Musical genre classification of audio signals, *IEEE Transactions on Speech and Audio Processing* 10(5): 293–302.

Vesanto, J. & Alhoniemi, E. (2000). Clustering of the self-organizing map, 11(3): 586–600.

Vesanto, J., Himberg, J., Alhoniemi, E. & Parhankangas, J. (2000). Som toolbox, Helsinki University of Technology.
URL: *http://www.cis.hut.fi/somtoolbox/*

Wu, S. & Chow, T. (2004). Clustering of the self organizing map using a clustering validity index based on inter-cluster and intra-cluster density, *Pattern Recognition Letters* 37(2): 175–188.