AN EXHAUSTIVE ANALYSIS OF AUTOMATIC GENRE CLASSIFICATION

Bruno Afonso Nº.53732

Instituto Superior Técnico
Av. Rovisco Pais, 1049-001 Lisboa, Portugal
E-mail: bruno_afonso@msn.com

ABSTRACT

The increase of the musical databases on the Internet and multimedia systems have brought a great demand for music information retrieval (MIR) applications, and especially automatic analysis of musical databases. A classifier system that allows to make a genre classification, considering a symbolic database into hierarchically organized taxonomies is developed in this work. It is also described the process of applying supervised Machine Learning (ML) to music genre classification. The classification system begins with the feature extraction from the Musical Instrumental Digital Interface (MIDI) files, in which it was applied then a feature selection algorithm, known by Sequential Floating Forward Selection - Least Squares. Finally, the classifiers used in the classification system were Support Vector Machines (SVM) and K-Nearest Neighbour (KNN).

A similarity metric known by Normal Compression Distance (NCD) and based on Kolmogorov Distance was also tested. NCD was used in the KNN classifier, in which new approaches were realized, mainly in the discretization procedure, and in the conversion of the feature values into a string. As the results obtained with NCD, by considering the KNN classifier, were higher than the results obtained with the Euclidean distance, it was decided to use the NCD in the KNN considered in the classifier system.

The classifier ensemble proved to be an interesting approach, achieving results that can be compared to the best music genre classification methods reported in the literature.

Index Terms — Musical Genre, Automatic Genre Classification, Classifier Ensemble, Hierarchical taxonomy, Pre-processing, Support Vector Machines.

1. INTRODUCTION

Musical genre has a particular importance in the field of music information retrieval, because it is used by retailers, librarians, musicologists and listeners in general, as an important means of organizing music. The need for an effective automatic way of classifying music is increasing, since the manual genre classifications are a slow and inaccurate methodology. Nowadays, the music industry is powerful with genres being used as a key way of defining and targeting different markets because the purpose of such organization is to enable users to navigate to pieces of music they already know and like, but also to give them advice and recommendations (“If you like this, you might also like...”). Fabri [1] defined genre like a set of musical events whose course is governed by a definite set of socially accepted rules. This definition is essential to understand the main classification paradigm used: Supervised Learning. It is a machine learning technique for learning a function (set of rules) from the training data. The training data is formed by a set of samples that are considered as prototypical or a model of the categories considered. The process of applying supervised Machine Learning (ML) to a real word problem has three common steps: feature extraction, training of the classifiers and the classification.

The key to the success of whole process is the choice of features. It is important to use an algorithm that allows to extract the higher number of possible categories of features. The algorithms used were jSymbolic[2] and jAudio[3]. Unfortunately, the use of too many features can overload the classifiers, which can justify the need of a selection feature algorithm. The classifiers used are Support Vector Machines (SVM) and K-Nearest Neighbor (KNN). Recently, [4, 5] have suggested the using of an approximation to Kolmogorov distance between two musical pieces as a mean to compute clusters of music. They first process the MIDI representation of a music piece to turn it into a string from a finite alphabet. Then, they compute the distance between two music pieces using normalized compression distance (NCD). NCD uses the compressed length of a string as an approximation to its Kolmogorov complexity. Although the Kolmogorov complexity of a string is not computable, the compressed length approximation seemed to have good results in other works [6, 7, 8]. The novel aspects of this paper are as follows: SFFS-LS was the selection algorithm chosen. In fact, it was never used in genre classification. SVM has been used in works considering audio databases [9, 10]. Reasonable results have
been obtained in these approaches. It is expected that SVM may have an interesting performance when a symbolic dataset is used. Regarding the classifier KNN, the information extracted from MIDI files, and tested with NCD, has been based on the essence of the melody [6, 7, 8], which is indeed very limited information. In this work, other important categories of features were also used, like instrumental. Actually, in [12], instrumentation features had an important weight of 42% among all the available features present in jSymbolic. This demonstrates the relevance of this category.

The structure of this paper is organized as follows: Section 2 reviews recent research in automatic genre classification. Section 3 provides an overview of the features while the key to the success of the genre classification. Section 4 discusses the feature selection and classification techniques considered and explain how these techniques were used to perform hierarchical classifications. Section 5 presents the experiments realized to test the effectiveness of the system, and the results. Finally, Section 6 provides some final conclusions.

2. TECHNICAL BACKGROUND

The main classification paradigms are Supervised Learning (SL) and Unsupervised Learning (UL): in UL, the systems cluster the data that they are fed based on similarities that they perceive themselves rather than model categories. SL attempts to formulate their own classification rules by using machine learning techniques to train model examples. SL is based on how humans classify genres when those genres are already known, and consequently there is an important knowledge and training. Otherwise, the UL is based on how humans classify music when there is no previous knowledge. To sum up, SL seems to be the most appropriate classification paradigm. Therefore, all the works described in this section are based in SL. A summary of existing musical genre classification systems based on Supervised Learning is presented below.

2.1. Automatic Genre Classification considering Symbolic Databases

There is a number of interesting studies on automatic genre classification, considering symbolic and audio databases. In [12], Cory McKay presented two well known types of classifiers, which were used to classify symbolic data: feedforward neural networks (NN) and k-nearest neighbor (KNN). The database was formed by 225 MIDI files, 25 MIDI files per genre. An accuracy of 98% was obtained with three root genres (Classic, Jazz and Rock) and an accuracy of 90% with nine leaf genres (Baroque, Modern Classic, Romantic, Punk, Rap, Country, Bebop, Jazz Soul and Swing). The same author, in a further work [13] used a music classifier and meta-learner named Autonomous Classification Engine (ACE). ACE is a meta-learning software package for selecting, optimizing and applying machine learning algorithms. An accuracy of 66.4% was obtained with ACE, when applied to a taxonomy formed by ten genres.

P. León and J. Iñesta [14] algorithm uses a combination of k-nearest neighbors and Bayesian classifiers trained with different sets of statistical descriptors extracted from melody tracks that are in the MIDI files. With a hierarchical classification, an accuracy of 50.67% was achieved from a database with nine genres.

2.1.1. A similarity metric based on compression

There is no guarantee that NCD will be useful for genre classification, but previously works with Kolmogorov distance and NCD had quite reasonable results. In [7], it was demonstrated that this method based in NCD is not restricted to a specific application area. Actually, NCD is applicable to a wide variety of applications like genomics, languages, astronomy or music. In the music area there has been done some researches with interesting results. In [15], 771 MIDI files falling into 4 categories were considered; the two genres considered were western classical music, composed by Beethoven (302 files) and Haydn (261 files), and the two remaining genres were Chinese music (80 files) and Jazz (128 files). The dataset considered is poor if the main goal is to test NCD with a realistic taxonomy. However, if the main goal is to test the performance of NCD based on a LZ78 compressor, the dataset can be considered interesting. In that work an accuracy of 92.4% was achieved, considering a KNN classifier with the K parameter equal to 1. In [6], realistic classification taxonomy was used with a genre classification based on NCD. The dataset used was the same dataset considered by Cory McKay in [16], a 3-root and 9-leaf genre dataset of 225 MIDI files. However, the results were lower than the results obtained in the work described in [16]. The best results achieved by the authors were obtained by using the KNN classifier with K parameter equal to 10, the 3-root genre achieved 75% accuracy and a 9-leaf genre achieved 42% accuracy.

2.2. Automatic Genre Classification considering Audio Databases

Tzanetakis and Cook [17] considered for each genre a Gaussian classifier, which was trained using a dataset of 50 audio samples (each 30 seconds long). Six genres were considered: Classic, Country, Disco, Hip Hop, Jazz and Rock. An accuracy of 62.2% was obtained. Doraisamy, Golzari, Norowi, Sulaiman and Udzir [18] analyzed a database formed by traditional Malay music. They used Marsyas Software [19] to extract audio features and verified the performance of different feature selection and classification methods. It was concluded the addition of selection algorithms improves the final accuracies.
results showed that Multilayer neural network with back propagation (MLP) and Support Vector Machine (SMO) were the classifiers with the higher accuracy. Other interesting conclusion was the improvement of the accuracy in 1%, which is related to the using of selection algorithms. Cataltepe, Yaslan and Sonmez [8] used Linear Discriminant Classifiers (LDC) to classify a taxonomy formed by nine genres. The content-based audio features were obtained using Marsyas Software. One important particularity of this work is the conversion of the MIDI files in WAV files. The higher classification accuracies obtained were 86% for 3-root genre, and an accuracy of 63% for 9-leaf genre.

R. Neumayer [20] used Netlab Toolbox of Aston University [21]. Classification is done through multi layer perceptrons.

3. FEATURE EXTRACTION

After collecting the dataset, the second step in MIR is to extract features from music samples. Features serve as the input to pattern recognition systems and are the basis upon which classifications are made. Even an optimal classifier cannot correctly classify music if it is not provided with features that properly segment recordings into the required taxonomy. There are some available methods that allow the extraction of features.

Music data is generally stored digitally as audio format or symbolic format. Musical Instrumental Digital Interface (MIDI) is the most used symbolic format. On the other side, Waveform audio (WAV) and MPEG-1 Audio Layer 3 (MP3), are the most common audio formats. Most of the research in MIR used audio data instead of symbolic data.

This is a logical choice because music is mostly consumed by humans in audio format, so a classifier system that can make a more accurate classification from an audio database can be logically considered more interesting and practical. However, audio data represents sound signals by encoding analog waves as digital samples, which is considered a “low-level” representation of music. On the other hand, symbolic data is considered a “high-level” representation, because it stores musical events and parameters themselves rather than actual waves [2]. This means that it is much easier to extract high-level features from symbolic data than from audio data. High-level features contain information that consists in meaningful musical abstractions to musically trained individuals. This seems to be a huge advantage of symbolic data. However, MIDI files lose some useful information that could be important for genre classification. The quality of singing voice, lyrics, phrasing and expression can be eliminated in MIDI format, features that are often used by humans to make their own genre classification.

It is also possible to extract high-level features from audio format, but it is not a trivial task and a much wider range of high-level features can be extracted from symbolic data. Although low-level features do not provide information that seems to be musically intuitive, there is no doubt that they can have significant discriminating power when processed by computers.

There are some available algorithms to extract features from MIDI files like midicsv [22] and jSymbolic[2]. With jSymbolic is possible to extract a set of 111 high-level features, in which 101 are one-dimensional (those that consist of a single value) and the other 10 multi-dimensional (those that consist on an array of values). Each of these features can belong to one of these categories: Pitch statistics, Instrumentation, Texture, Dynamics, Melody and Chords.

When WAV files are considered, instead of MIDI files, one possible method to extract features is jAudio[3], that was developed by the same authors of jSymbolic. This software package allows the extraction of a quite large set of features that can be grouped in timbral, rhythmic and melody categories.

4. CLASSIFICATION METHODOLOGY AND FEATURE SELECTION

4.1. Feature Selection Algorithm

The feature selection involves the problem of selecting a minimal subset of M features from the original set of N features (M ≤ N). The features that will be chosen by a selection algorithm are a subset of all available features set that is the most appropriate in the context of particular taxonomies and classification problems. After selecting features the classification process can be done with lower computer requirements. When the main goal is to achieve an effective classification, it is therefore fundamental to have a relevant number of available features, and then use a feature selection method. If these selection techniques were not done, it would necessarily mean more training time and classifier overload. This is known by “curse of dimensionality”, which means the need for additional training samples grows exponentially with the dimension of the feature space. So the challenge is to pick features out from a wide available group, hence it is necessary to take a reasonable number of features from MIDI files and choose some of them to be the input of the classifier. Indeed the application of selection techniques has a relevant impact in the final results. In this work SFS (Sequential Forward Selection) was considered as a good approach to the feature selection algorithm. The evaluation criterion of Sequential Forward Selection consists on dependency measures. They are also known as correlation measures or similarity measures. It measures the ability to predict the value of one variable from the value of another. In feature selection for classification, it was looked for how strongly a feature is associated with the class. A certain feature is preferred to another feature if the association between this feature and a
certain class is higher than the association between another feature and that class. This technique starts with the empty set and adding features one by one until some maximum number of features were selected. It first considers that all possible features subsets are formed only by one feature, and choose the best one based on evaluation criteria. Next, from all possible features subsets that are formed by all combinations between the previous chosen feature and one more feature, the subset with higher result based on evaluation criteria is the new selected subset. This algorithm is a cycle where the next step always evaluates all possible subsets that are formed by all features selected before plus one of the available features that have not already been selected. However, SFS has a considerable disadvantage that is no way to remove or restore a feature that has already been added to the chosen subset of features. This problem is known by nesting, and for classification systems is a really limitation because one feature can be relevant when it is considered at the beginning of the selection process but in a later stage of the process it can no longer be one of the best features with improved performance. Fortunately, there is an alternative version that belongs to a class of SFS named SFFS (Sequential floating forward selection) that contains two steps: First, the feature leading to the largest improvement is included, and then the scheme goes back in the search and can exchange some previously selected features by the new ones if they improve the selection according to evaluation criteria. SFFS algorithm is an interesting selection technique that can be used in Musical Information Retrieval (MIR). More information about Sequential Forward Selection and its versions are available in [23].

4.2. Classification Algorithms

The most widely used classifiers can be classified as parametric or nonparametric classifiers. Nonparametric classifiers (e.g., nearest-neighbor and kernel-based classifiers) are more flexible and free from parametric model assumptions. But, the statistical instability of these classifiers may lead to poor performance when we have small numbers of training sample observations. On the other hand, parametric classifiers work well when the model assumptions obtained through the previous knowledge are valid. Violation in one or more of these parametric model assumptions often leads to a poor classifier. Nonparametric methods do not use any parametric structure of population densities. Therefore, even when one has some additional information about population densities, that important information is not used to modify the nonparametric classification rule.

In MIR there is no prior knowledge about feature’s population, this knowledge is forced to rely on information gleaned from training samples. Therefore, it can be concluded that nonparametric classifiers are more appropriate. The nonparametric classifiers considered in this work are going to be described next.

4.2.1. K-Nearest Neighbor

K-Nearest Neighbor is a classifier where each sample is classified based on majority of K-Nearest Neighbor category. The purpose of this algorithm is to classify a new object based on attributes and training samples. In a first step, training data is used to store the coordinates of each training sample in the multi-dimensional feature space. Then, when testing data is used, each test sample is classified by examining its surroundings in feature space and finding the labels of the nearest k training points. Different distance metrics can be used, nevertheless Euclidean and Manhattan distance are normally used. KNN classifiers do not use any model to fit, they only work based on memory. The choice of k is important; this can be considered a disadvantage because there is no a sophisticated method to get an optimal k. The KNN main advantage is related to the need for a reduced training time, but it is not capable of making complex relationships between features, and all features are considered as a whole by KNN classifiers.

4.2.1. Support Vector Machines

A Support Vector Machines (SVM) performs classification by constructing an N-dimensional hyperplane that optimally separates the data into two categories. SVM models are closely related to neural networks. Using a kernel function, SVM’s are an alternative training method for polynomial, radial basis function and multi-layer perceptron classifiers in which the weights of the network are found by solving a quadratic programming problem with linear constraints, rather than by solving a non-convex, unconstrained minimization problem as in standard neural network training. In the parlance of SVM literature, a predictor variable is called an attribute, and a transformed attribute that is used to define the hyperplane is called a feature. The task of choosing the most suitable representation is known as feature selection. A set of features that describes one case (i.e., a row of predictor values) is called a vector. So the goal of SVM modeling is to find the optimal hyperplane that separates clusters of vector in such a way that cases with one category of the target variable on one side of the plane and cases with the other category on the other side of the plane. The vectors near the hyperplane are the support vectors. Unfortunately, in the majority of applications samples are not well separated, which means that it is not possible to divide two groups with a straight line, flat plane or N-dimensional hyperplane. One might think that in this case it would be necessary a nonlinear line to divide the two categories. Rather than fitting nonlinear curves to the data, SVM handles this by using a Kernel function to map the data
into a different space where a hyperplane can be used to do the separation.

4.3. Normal Compression Distance

Recently was presented a similarity metric based on compression as a promising way to make a distinction between data from different domains and grouped it with a certain similarity. This method does not use subject-specific features or background knowledge about the data. The logic behind this method is really simple: two objects are deemed close if it would be possible compress one given the information in the other. Using a similarity metric based on compressibility allows estimating how much information is shared between two pieces of music. However, the really interesting idea is if two objects are close according to some effective similarity, then they are also close according to the normalized information distance. The normalized information distance (NID) is a metric and takes values in [0,1]. Many distances are absolute, but when it is desired to express similarity, then it is more interesting to use relative ones. For example, if two strings of length $10^3$ bits differ by $10^2$ bits, then those strings are relatively more similar than two strings of $10^2$ bits that have that distance and consequently are 100% different.

$$NID(x,y) = \frac{\max\{K(x|y),K(y|x)\}}{\max\{K(x),K(y)\}}$$

In NID the compression is based on Kolmogorov Complexity, which means the information of an object is defined as the length of the shortest binary computer program that describes the object, so if the object is defined by $x$ for example, this means that the Kolmogorov complexity of $x$ is defined by $K(x)$, which is the length of the ultimate compressed version of $x$. However, Kolmogorov complexity is non-computable, in other words there is no program that has as input a certain object $x$, and would have as output the $K(x)$. Actually, most of mathematical application and theorems are not directly applicable in real-world but still are interesting concepts that could be approximated and used in real-world applications. It is then necessary to make an approximation that is to replace the Kolmogorov complexity by the length of the compressed object using real-world normal compressors. Considering the previous example, the length of compressed $x$ is $C(x)$ instead of $K(x)$. The resulting of this NID approximation is known by the Normalized Compression Distance (NCD).

$$NCD(x,y) = \frac{C(xy) - C(x)}{C(y)}$$

4.4. Classifier Ensemble

4.4.1. Implementation of the classifier system

The classification system was built considering the hierarchical taxonomy used. In fact, hierarchical taxonomy is similar to the way how humans organize genres. By resorting to hierarchical classification the desired was to achieve better classification performance than with the so-called “flat” systems, where all categories are put at the same level without any arrangement. Hierarchical classification has some important advantages in the classification process. There are two feature types: one-dimensional and multi-dimensional. These features are extracted from recordings and they are the input of the classification system. In one-dimensional features a feature selection algorithm can be used, where M of N available features (M<N) are selected. This allows reducing the computational requirements and processing time, and would contribute to make an efficient classification. The selection algorithm used was SFFS-LS. SFFS was combined with LS Bound measure[24]. This is a measure that is derived from leave-one-out procedure of Least Squares Support Vector Machines (LS-SVM). The strength of the LS Bound measure is that it provides feature subsets leading to more accurate classification results than the filter method while the computational complexity is at the level of the filter method. This is the only difference between one-dimensional and multi-dimensional features in the classification system process. The classification system is formed by nonparametric classifiers: Support Vector Machine and K-Nearest Neighbor. Both classifiers were used from Matlab Toolbox. In the first step a training database to train the classifiers is used. In KNN, K value is set to the square root of the number of training samples used in each genre. Regarding the SVM, it only allows to classify between two different categories. Therefore, the “one against many” option was used, in which each category is split out and all of the other categories are merged. This option was considered the best approach because “one against one” option is computationally more complex and requires more processing time. However, there is a relevant drawback related with “one against many” approach: in the taxonomies used a recording can only belong to one and just one genre. An undesirable situation in which SVM cannot classify a recording as belonging to a genre or classify a recording as belonging to more than one genre can happen. If it happen when a certain feature category is considered, the recording will be labeled as “Unknown”. For each recording there are a set of labels that were obtained using SVM for each of the available features arrays considered. So, each recording has N labels assigned, in which N is the number of features arrays used. This number is always the result of the number of multi-dimensional features plus one that is related with one-dimensional
features array. These N labels will be used in majority voting process which is represented by the Equation 3.

\[ R_j = \sum_{i=1}^{N} D_i \times P_i \text{ where } D_i = \begin{cases} 1, & \text{if label is not "Unknown"} \\ 0, & \text{if label is "Unknown"} \end{cases} \]
\[ P_i = \begin{cases} 1, & \text{if multi-dimensional features} \\ 0, & \text{if one-dimensional features} \end{cases} \] (3)

The “Unknown” labels are not considered in this weighting process, which means that they do not count. KNN will only classify recordings labeled like “Not Labeled” even after majority voting process. Actually, it is extremely important to explain the difference between an “Unknown” recording and a “Not Labeled” recording: A recording is “Not Labeled” when even after the majority voting process it continues “Unknown”. It is important to explain the situations in which a recording is “Not Labeled” or in other words when KNN is called: (1) If all the labels obtained with the different features sets were classified as “Unknown”. (2) If there is more than one label with the same higher Rj.

So, it can be said that SVM is the main classifier and KNN is an auxiliary classifier that works together with SVM. SVM allows to model more complex logical analysis in relationships between features, but even with these complex analyses a recording cannot be classified, KNN is used in this situation. This is a new solution to overcome the undesirable situation in which SVM cannot classify the recordings due to “one against all” approach.

Next, it is presented how majority voting process works: If a certain recording j is labeled by SVM as “Unknown” Dj is 0. Otherwise Dj is 1 which allows associating the recording to the Pj parameter that is used to assign a different level of preponderance to one-dimensional and multi-dimensional features. After doing this procedure for all the N features sets considered there is a Rj for each genre that belongs to the hierarchical layer. The recording j belongs to the genre with the higher Rj value.

This majority voting process is then applied to both types of features: one-dimensional and multi-dimensional, in which is associated a high (Pj = 3) parameter to one-dimensional features set and a lower (Pj = 1) parameter for each of multi-dimensional features. One-dimensional features are considered like an array of features, but they are not directly related with each other. In fact, one-dimensional features can be assigning to different categories like Instrumentation, Musical Texture, Rhythm, Pitch Statistics, Melody and Chords. So, it makes sense assign a higher weight to one-dimensional features array, for the reason that this array is formed by a certain number of one-dimensional features which provides large and varying information about the recording. Unlike each of multi-dimensional arrays is formed by just one musical category.

4.4.2. Kolmogorov Distance in the classifier system

Regarding the feature extraction, jSymbolic was used. All the pre-processors developed in the previous works [6, 7, 8] used a feature extraction algorithm similar to midicsv, in which the core information of the MIDI files is present but without any kind of handling, in other words it can be said that the information is in the raw form. Consequently, the information needs to be handled which means that the pre-processor will be responsible for that. This can explain why the pre-processors used in previous works are quite complex to implement. By using jSymbolic, all the features from MIDI files are already organized by categories and handled which necessarily means a less complex pre-processor implementation. This can be an important advantage of this new approach. After extracting the features, it is necessary to normalize them, this normalization is essential to ensure the same range of values in all features. Then, an algorithm that converts the feature values into a string has to be developed because the input of a classifier should be a sequence of characters. Actually, it is only necessary to make a discretization procedure, which allows to constrain a continuous set of values (features) to a discrete set that is one of the intervals in which features values are divided. For each interval it is assigned a symbol which necessarily means that a feature value that belongs to a certain interval will be represented by the symbol that was assigned to the interval. Two different approaches were considered to convert feature values into strings: The first one considers that each symbol is represented by a cararacter that will be the character assigned to the interval and that will be placed in the string. This means that a codebook was used. In the second one, each of the N = 2^M intervals was enumerated and converted to an M-bit word. Each M-bit word was assigned to the feature values that belong to the respective interval. Therefore, two different pre-processors were tested, in which the main difference is the conversion of the feature values into a string.

Kolmogorov complexity is non-computable. Therefore, it is necessary to make an approximation which is to replace the Kolmogorov complexity by the length of the compressed object using real-world normal compressors. The compressors considered in this work were: bzip2, gzip and LZW12. The matrix obtained from the equation (2) is the similarity matrix that will be one of the inputs of the classification algorithm used; in this case the classification algorithm used is KNN. In this equation, C(x) is the length of the compressed file x in which are present the data strings obtained through the binary words or the codebook used in the conversion of the feature values into data string. On the other hand, C(xy) is the length of the compressed file that is obtained with the concatenation of both data strings from file x and file y, in which each of the files is assigned to a recording. The parameters described above were obtained by using the normal compression algorithm. It is important
to mention that the input of the file is the output of the pre-processing step that was described before. In other words, the strings obtained through the pre-processing algorithm are the input of the file that is then compressed. KNN is the classifier used, in which k parameter was set as the square root of the number of samples per genre in the training set. As the aim is to test the viability of the Kolmogorov distance, a classifier like KNN was considered a reasonable approach, for the reason that is easy to implement and it has a negligible training time.

5. THE EXPERIMENT

5.1. Results obtained with the Classifier Ensemble Formed by SVM and KNN

Two different taxonomies were analyzed, the first one is formed by 3-root genres and 9-leaf genres and will be named by Taxonomy A (T – A), while the other one is formed by 5 root-genre and 10 leaf-genres and will be defined as Taxonomy B (T – B). However, in audio dataset only Taxonomy B (T – B) was used. Each genre is formed by 25 recordings. These taxonomies were used in previous works [8, 12, 13], which allows to compare with the results obtained in these works.

In the experiments a 5-fold cross-validating was considered, which means that each experiment was performed five times with randomly chosen training and testing sets. All the results reported for each experiment in this chapter are averages of the results obtained in the 5-fold cross-validating. The database was divided in two different groups: training and testing. This division was done by an algorithm provided by Matlab Toolbox that returns randomly generated indices for a K-fold cross-validation of N observations. It was decided to use 60% of recordings for training and 40% for testing. This distribution is quite optimistic. Actually, a higher percentage of recordings for training is always used because it would assure a better learning process by classifiers, in [16], for example, the same database was used but 80% of the recordings were used for training and 20% for testing. This decision was conscious and well considered, a smaller training set means a lower time processing, if the classifier system was able to learn with a smaller training set this is an interesting way to test it.

The results using hierarchical classification were quite interesting, as can be seen in Table 1.

The SVM algorithm has some input parameters that are related with the Kernel function which allows to specify the Kernel function used to map the training data into a Kernel space. The best accuracies were all obtained for Linear Kernel function. For the training algorithm parameter, quadratic programming (QP), sequential minimal optimization (SMO) and least-squares (LS) methods were tested. The higher accuracies were obtained with QP.

Table 1 – Results obtained with the classifier ensemble

<table>
<thead>
<tr>
<th></th>
<th>Training</th>
<th>Success Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root Genres (T - A)</td>
<td>97.78%</td>
<td></td>
</tr>
<tr>
<td>Root Genres (T - B)</td>
<td>91.00%</td>
<td></td>
</tr>
<tr>
<td>Leaf Genres (T - A)</td>
<td>85.56%</td>
<td></td>
</tr>
<tr>
<td>Leaf Genres (T - B)</td>
<td>72.00%</td>
<td></td>
</tr>
</tbody>
</table>

Another important aspect is related with the confusion matrix (Figure 1, 2). Actually, the confusion matrix is an important visualization tool used in supervised learning, in which each row of the matrix represents the instances in a predicted class, while each column represents the instances in an actual class. In fact, in genre classification the confusion matrix provides more important information that is related with the subjectivity existent in genres.

When an audio database is considered, the features that are extracted from it are typically low-level. Although these features do not provide intuitively musical information, audio is clearly useful because it is the essential way in which music is consumed. In this experiment jAudio was considered instead of jSymbolic and only the T – B was tested.

The results obtained (Table 2) are far from expected, which means this classification system is not appropriate when an audio database is considered. This can be related with the low-level features extracted from audio, in which it becomes
harder for the classifier systems to learn how to identify the genres present in the database.

Table 2 – Results obtained with the classifier ensemble considering an audio dataset

<table>
<thead>
<tr>
<th>Training</th>
<th>Success Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root Genres (T - B)</td>
<td>72,00%</td>
</tr>
<tr>
<td>Leaf Genres (T - B)</td>
<td>43,00%</td>
</tr>
</tbody>
</table>

In fact, the classifier system has a reasonable performance in the root layer, only the Blues has a poor accuracy which can be explained with its origin and the subjectivity that is inherent to this genre.

5.2. Results obtained with the KNN considering the Kolmogorov Distance

Considering a hierarchical taxonomy, it is possible to verify the Kolmogorov distance behavior when a limited number of very distinct categories is used, and also when a certain numbers of categories, that include overlap and potentially objectively irrational distinctions is also considered. If in the root level the classification rates obtained are good the same does not happen in the leaf level. In root level, the best accuracy obtained 86,67%. The best accuracy obtained in the leaf level was 57,78%. These accuracies were obtained considering the pre-processor A1. It is an interesting result, because pre-processor A1 is based on two new approaches: the features were obtained through jSymbolic; and the conversion of the feature values into a string is done considering the codebook.

In fact, it is possible to conclude that pre-processor A1 has a better performance when compared with the pre-processor implemented in [8]. There is an improvement of 11,67% in the root accuracy and an improvement of 11,78% in the success accuracy.

Actually, besides the better accuracies obtained, pre-processor A1 has another important advantage: it is easier to obtain important information (features) from the recordings using appropriate software, which is already programmed to perform it. Therefore, pre-processor A1 does not need to be as complex as the pre-processors developed in previous works. This advantage also allows to use a higher diversity of feature categories like pitch statistics, instrumentation, texture, dynamics, melody and chords.

To demonstrate the importance of the instrumental features, the pre-processor A1 was considered again, but this time without instrumental features. The results are demonstrated in Table 3.

To understand and correctly analyze the pre-processor performance, more specifically the NCD performance is essential to have a comparator term.

Therefore, the Euclidean distance was also tested with KNN and the results can be compared. In T – A, NCD had a higher performance than Euclidean distance. In the root accuracy, there was an improvement of 8,89%, and in success rate an improvement of 18,89%. In T – B both performances were almost similar, but Euclidean distance had a better performance of 2% in root accuracy and 7% in success accuracy. Even that, NCD had quite an interesting performance.

One might think, and this thought is surely valid, that it would be an interesting approach to use NCD instead of Euclidean distance in the classifier system described in 4.1.1. Therefore, in the KNN used in the classifier system, the NCD was considered with the hope that the success accuracy obtained would increase. The results obtained are represented in Table 4.

Actually, the success rate in both taxonomies increased. This improvement means that NCD applied to an appropriate pre-processor can be used in a classifier system. To sum up, NCD can be considered an important tool to use in MIR applications.

The impact of KNN in the classifier ensemble is minor than expected, which is related with the configuration of the classifier system, explained in 4.1.1. If KNN had a more significant weight in the classifier system, the impact of the NCD in the classifier system would be surely more influent. Even so, the impact is positive in both taxonomies.

6. CONCLUSIONS AND FUTURE WORK

In this work, two different approaches were considered with the aim of classifying a symbolic database per genre: In the first one a classifier ensemble formed by SVM and KNN was used, and in the second one a similarity metric based on compression was also tested with KNN. The genre classification task, due to its complexity and subjectivity is a very interesting test to the classifiers. It is true that the majority of music is consumed through audio formats, but it is also true that is more difficult to extract valuable information from audio. Therefore, it has been given more

Table 3 - Impact of the instrumental features in pre-processor A1

<table>
<thead>
<tr>
<th>Pre-processor A1</th>
<th>Success Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>With Instrumental Features</td>
<td>57,78%</td>
</tr>
<tr>
<td>Without Instrumental</td>
<td>50,00%</td>
</tr>
</tbody>
</table>

Table 4 - Performance of the classifier ensemble considering the Euclidean distance and NCD in KNN

<table>
<thead>
<tr>
<th>Success Rate</th>
<th>T – A</th>
<th>T – B</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM + KNN (Euclidean distance)</td>
<td>85,56%</td>
<td>72,00%</td>
</tr>
<tr>
<td>SVM + KNN (NCD)</td>
<td>86,67%</td>
<td>74,00%</td>
</tr>
</tbody>
</table>
weight to the symbolic databases which can be explained with the high-level features that are possible to extract from MIDI files. Actually, symbolic format has an important role in genre classification and it is being used in some investigation works in which the classification utility of combining features extracted from separate audio, symbolic and cultural sources of musical information has been tested. Symbolic formats can thus, at the very least, be a powerful representational tool in automatic music classification. Considering the symbolic databases formed by MIDI files it was possible to obtain quite good results in the two different taxonomies used. The best results obtained with the classifier system developed in this work were 85.56% in success rate of T – A and 72% in success rate of T – B.

In fact, by comparing the results obtained with the classifier system developed in this thesis with the accuracies obtained in previous works, it can be concluded that they are very motivating. Mainly, because a smaller training set is considered and consequently a higher testing set had to be used. The training time consumed was another important positive aspect of the classifier ensemble. The higher training time consumed by this classifier ensemble was 6.82 minutes in T – A and 8.81 minutes in T – B which are, in fact, extraordinary values. Then, it can be concluded that SVM is an appropriate classifier with interesting results when a symbolic dataset is considered. The optimization of some parameters used in SVM can improve the results. For example, SVM allows experimenting or developing other kernel functions that could contribute to improve the results obtained. Another new approach used in this work was the selection algorithm: SFFS-LS. In fact, by using SFFS-LS there is a considerable improvement in the success accuracies: 5.56% in T – A and 1.00% in T – B.

Also in this work, a similarity metric based on compression was studied: NCD. The performance of this similarity metric has been tested in other works, nevertheless the preprocessors used are quite complex. In this work, some simpler preprocessors were developed, in which new categories of features never used before were considered with NCD. The results were promising. In fact, the results obtained with KNN classifier considering the pre-processor developed are higher than the results obtained when KNN was used considering the Euclidean distance. The best results obtained were a root rate of 86.67% and a success rate of 57.78% in the T – A. T – B was also tested with NCD, which was an interesting test since NCD was never used before in such large taxonomy. The results obtained were 59.00% in a root rate and a leaf rate of 36.00%. In a first analysis, these results seem to be lower, but by comparing with the results obtained when the Euclidean distance was considered in KNN, it is possible to conclude that NCD has quite an interesting performance. Due to the better performance of NCD, it was also used in the classifier ensemble, instead of Euclidean distance. In fact, the classifier system performance increased. It is possible to conclude that NCD is a viable alternative to the Euclidean distance when a KNN classifier is used.

With the elaboration of this thesis it was also conclude that the taxonomy chosen has a significant impact in the final accuracies. This is, in fact, an important aspect. Therefore, it was decided by the author of this work considering two different taxonomies, which allows to verify the behaviour of the classification system developed in two different situations, in which different genres were considered.

Regardless of the encouraging results obtained, the solution developed still leaves room for improvements:

- **Weighting Process** – The use of a more complex weighting process could improve the system performance. Mainly when an audio database is considered, in which 37 different multi-dimensional features are extracted by jAudio, it would be an important upgrade in the classifier ensemble. The lower accuracies obtained with the audio database through the classifier developed may be related with the non-use of a complex weighting process.
- **Larger Taxonomy** – The use of a larger taxonomy, with new root and leaf genres that are meaningful to general listeners could be added. Some of these genres could be Dance, Country and Latin, for example. It would be an interesting way to test the classifier performance when new genres are considered.
- **Combining Features** - Combining features extracted from separate audio, symbolic and cultural sources of musical information has given good results. The classifier system may be used for the three different sources.
- **Kernel Function** – There are other available Kernel function that can be used in the SVM of the classifier system. Few of the Kernel function are based on NCD, in which the pre-processors developed in this work may be used.
- **Multi Genre Approach** - Each recording should be permitted to have more than one genre label, and labels should be weighted.

### 7. ACKNOWLEDGEMENTS

I would like to express my gratitude to Cory McKay for generously sharing his data sets.

### 8. REFERENCES


