UNIVERSIDAD NACIONAL DE COLOMBIA

Categorización de letras de canciones de un portal web usando agrupación

Fabio Leonardo Parra Anzola

Universidad Nacional de Colombia
Facultad de Ingeniería, Departamento de Ingeniería de Sistemas y Computación
Bogotá, Colombia
2013

UNIVERSIDAD NACIONAL DE COLOMBIA

Web lyrics categorization using clustering

Fabio Leonardo Parra Anzola

Universidad Nacional de Colombia
Facultad de Ingeniería, Departamento de Ingeniería de Sistemas y Computación
Bogotá, Colombia
2013

Web lyrics categorization using clustering

Fabio Leonardo Parra Anzola

Trabajo final presentado como requisito parcial para optar al título de:
Magister en Ingeniería de Sistemas y Computación

Director(a):
Ph.D. Elizabeth León Guzmán

Líneas de Profundización:
Machine Learning and Software Engineer
Grupo de Investigación:
MIDAS

Universidad Nacional de Colombia
Facultad de Ingeniería, Departamento de Ingeniería de Sistemas y Computación
Bogotá, Colombia
2013

Dedicatory

This work is dedicated to my wife Jenny

Acknowledgements

To Ph. D. Elizabeth León Guzmán
To members of MIDAS group for their valuable suggestions and constructive criticism.

Resumen

Algoritmos de clasificación y de agrupación han sido usados ampliamente en sistemas de recuperación de información musical (MIR) para organizar repositorios musicales en categorías o grupos relacionados, por ejemplo género, modo o tema, usando el sonido o sonido en combinación con la letra de la canción. Sin embargo, la investigación relacionada con agrupación usando solamente la letra de la canción es poca. El objetivo principal de este trabajo es definir un modelo no supervisado de minería de datos para la agrupación de letras de canciones recopiladas en un portal web, usando solamente características de la letra de la canción, con el fin de ofrecer mejores opciones de búsqueda a los usuarios del portal. El modelo propuesto primero identifíca el lenguage de las letras de canciones usando Naïve Bayes y n-grams (para el caso de este trabajo se identificaron 30.000 letras de canciones en Español y 30.000 en Ingles). Luego las letras son representadas en un modelo de espacio vectorial Bag Of Words (BOW), usando caracteristicas de Part Of Speech (POS) y transformando los datos al formato TF-IDF. Posteriormente, se estima el número apropiado de agrupaciones (K) y se usan algoritmos particionales y jerárquicos con el fin de obtener los grupos diferenciados de letras de canciones. Para evaluar los resultados de cada agrupación se usan medidas como el índice Davies Bouldin (DBI) y medidas internas y externas de similaridad de los grupos. Finalmente, los grupos se etiquetan usando palabras frecuentes y reglas de asociación identificadas en cada grupo. Los experimentos realizados muestran que la musica puede ser organizada en grupos relacionados como género, modo, sentimientos y temas, la cual puede ser etiquetada con técnicas no supervisadas usando solamente la información de la letra de la canción.

Palabras Clave: Recuperación de Información Musical, Agrupación de Páginas Web, Agrupación, Aprendizaje no Supervisado, Selección de Características, Minería de Datos, Minería de Texto Análisis de Letras de Canciones, Reglas de Asociación.

Abstract

Classification and clustering algorithms have been applied widely in Music Information Retrieval (MIR) to organize music repositories in categories or clusters, like genre, mood or topic, using sound or sound with lyrics. However, clustering related research using lyrics information only is not much. The main goal of this work is to define an unsupervised text mining model for grouping lyrics compiled in a website, using lyrics features only, in order to offer better search options to the website users. The proposal model first performs a language identification for lyrics using Naïve Bayes and n-grams (for this work 30.000 lyrics in Spanish and 30.000 in English were identified). Next lyrics are represented in a vector space model Bag Of Words (BOW), using Part Of Speech (POS) features and transforming data to TF-IDF format. Then, the appropriate number of clusters (K) is estimated and partitional and hierarchical methods are used to perform clustering. For evaluating the clustering results, some measures are used such as Davies Bouldin Index (DBI), intra similarity and inter similarity measures. At last, the final clusters are tagged using top words and association rules per group. Experiments show that music could be organized in related groups as genre, mood, sentiment and topic, and tagged with unsupervised techniques using only lyrics information.

Keywords: Music Information Retrieval, Clustering, Clustering, Unsupervised Learning, Feature Selection, Data Mining, Text Mining, Lyrics Analysis, Association Rules

Contents

List of Tables

List of Figures

1 Introduction

Internet has remained in growth for years. It is estimated that between 2011 and 2016, the average growth in consumer internet traffic will be 32% [13]. For taking advantage of this new traffic, it is necessary to organize textual information in websites, offering better search options to users and improving website user experience. Most of lyrics websites do not have tags in lyrics. However, clustering lyrics information could offer benefits to website users like new search options and recommendations based on previous lyrics visualizations. This could be a starting point for tag lyrics and then, with the website users, verify the relevance of those tags.

Areas of research in knowledge discovery and text mining have been commissioned to study this kind of problems, in order to extract an implicit, previously unknown and potentially useful information from data [6]. Making adjustments within the websites so users can handle this knowledge, will allow to offer better benefits in terms of accuracy and effort required.

In a standard web search engine, results are based on search words (query) used by users. In some cases, the results are not relevant. This is because it is possible that the query has different meanings according to the context. To solve this problem, some search engines use clustering algorithms (groupings of the results by proximity or similarity; in the case of text mining, clustering of documents can be by topic), like Carrot[1] or Yippy[2], in order to organize search results. In lyrics websites, clustering results could allow users to find more related songs according to their needs.

Song lyrics are documents with some particularities. They are organized in blocks of chorus and verses; they have rhymes and rhythm and may contain spelling mistakes because they are added by websites users. These characteristics have to be managed in preprocessing step to perform a good classification or clustering.

Most of the studies in MIR (Music Information Retrieval) that use lyrics have been conducted in classification tasks using audio and lyrics features. Although, there are some studies that use only lyrics information for classification tasks [11, 19].

[1] see: http://search.carrot2.org/stable/search
[2] see: http://search.yippy.com

In music there are not a standardized set of features for organizing repositories. MIR researches areas working on obtaining information from music and organizing music repositories automatically. In 2005 MIREX (Music Information Retrieval Evaluation eXchange) was created and it took place during the 6th ISMIR Conference. The goal of MIREX is to compare state-of-the-art algorithms and relevant systems for Music Information Retrieval. Since then, MIR researchers have developed different techniques and have made important contributions to music classification using sound [7]. In recent years, some MIR researches developed techniques using mixing approaches (sound + lyrics) in order to improve previous approaches that used only sound [16, 10, 17, 15, 22, 21].

MIREX has its own standardization and sound datasets that have been used for different music classification tasks like by genre and mood of the music. However, these works developed have not delved into unsupervised clustering techniques using only lyrics and taking into account a real environment of a lyrics website. This environment implies large datasets, limited computational resources, multi-language lyrics, the special language in lyrics and so on.

MIR has applications in music websites. These websites should evolve to give users effectiveness and speed in their searches. Many document websites have already organized their information; however, lyrics websites are relegated.

This work focuses on researching applicability and utility of clustering lyrics without any previous label or tags information using multi-language lyrics dataset compiled in a lyrics website, comparing results of clustering lyrics in Spanish and English, using Bag Of Words (BOW) features and using Part Of Speech (POS) features. In order to cluster these datasets, k-means and repeated bisections algorithms are applied evaluating results with Davies Bouldin Index (DBI), intra similarity and inter similarity measures. Finally, top words and association rules with FP-growth algorithm are extracted in every cluster to help to discover cluster tags.

1.1 Goal

The purpose of this work is to define an unsupervised text mining model for grouping lyrics of a web portal with multi-language content, in order to be able to offer to its users better search options. Specifically, this work concentrates on:

1. **Lyrics language detection.** Lyrics language was identified by Naïve Bayes using a cumulative frequency addition of n-gram technique.

2. **Lyrics preprocessing.** It applies lyrics stopwords removal and some clean steps like html extraction and some replacements with regular expressions in order to prevent mistakes in cluster lyrics information.

3. **Feature extraction.** It uses datasets with and without stemming in order to compare results of clustering. In this step TF-IDF weighting was used.

4. **Clustering techniques in a lyrics dataset.** Some different techniques are applied to the lyrics dataset to discover benefits in grouping results and compare different approaches in this area.

5. **Evaluation of clustering quality.** It evaluates different results with the help of internal and external measures of clustering quality. Additionally it takes into account the time of response and scalability.

6. **Lyrics annotation.** It discovers top words per cluster and association rules in every cluster with FP-Growth algorithm. Words and rules discovered help to tag lyrics clusters.

1.2 Research Process

For achieve the proposed goals it was necessary to apply the following steps:

1. **Find the dataset.** The dataset was obtained from a website[3] with around 300.000 lyrics in different languages.

2. **Review state of the art in lyrics clustering.** Researches related to text clustering and high dimensional, lyrics clustering, music information retrieval, topic detection in clusters and language detection in texts were the main groups that were necessary to analyse for performing this research. From them, it was necessary to review applicability to achieve the proposed goals.

3. **Build and select necessary tools.** For applying techniques and test results, it was necessary to find existing tools that could help in this process. The main tools that were found useful for this research was CLUTO [14] for clustering and freeling [26] for extracting lemma version of words. Other necessary tools for detecting language, pre-processing, additional clustering algorithms and validation measures among others were developed mainly in c# and java.

4. **Prepare the model.** It was necessary to apply the general process of text clustering for lyrics with its steps: pre-processing, clustering, evaluation and topic detection

[3] see: http://www.albumcancionyletra.com

several times in order to detect problems in the process and find solutions and options for them. These iterations allow us to select the appropriate model features satisfying the requirements defined in our goals. At the end, the last iteration was selected and it is the iteration presented in this document.

1.3 Main Contributions

The main contribution of this work is a model for grouping lyrics of a web portal with multi-language content. Additionally this work contributes in:

1. **Paper accepted in MLDM conference.** An article "Unsupervised Tagging of Spanish Lyrics Dataset Using Clustering" for the 9th International Conference on Machine Learning and Data Mining MLDM 2013 was accepted.

2. **Pre-processed lyrics datasets in English and Spanish.** Lyrics datasets contain 30.000 lyrics per language and have versions with lemmatization and without it.

3. **Model software prototipe for running and extracting clusters.** A program was developed in c# with all steps and algorithms used in this work.

1.4 Document Outline

This document is organized as follows:
Chapter 2 gives an state of the art of the techniques and concepts that are used in the model. This chapter contains the main task of MIR for supervised and unsupervised learning, a comparison of approaches for music classification and clustering, techniques and algorithms from text data mining that could be used in lyrics categorization process, some clustering quality evaluation methods, K estimation and topic detection approaches.

Chapter 3 develops unsupervised lyrics categorization process taking into account different clustering techniques and three main steps: pre-processing (language identification, POS processing, cleaning dataset, stopwords removal and TF-IDF weighting), clustering and evaluation (K estimation, K-means and Repeated Bisection, DBI index and Intra/Inter similarity index) and topic detection (frequent words and association rules).

Chapter 4 shows some conclusions and future work.

2 State of the Art

This chapter presents the background of techniques and concepts that are used in models for grouping and extracting knowledge from music repositories and text repositories. In lyrics case, text techniques are useful because lyrics are texts and could be worked with the same techniques.

2.1 Music Information Retrieval

In order to organize music in categories and obtain knowledge from music, MIR researches have applied different approaches using the info that music provides. There are approaches of supervised learning (classification) and unsupervised learning (clustering) that have shown important results for improving music repositories [31, 7, 15]. In order to apply these techniques, it is necessary first to make a feature extraction and after apply the classification or clustering approach for finally annotating music. Feature extraction addresses the problem of representing the examples that will be classified, in terms of feature vectors or pairwise similarities. For music, feature extraction could be by audio or by lyrics. Classification is a task in which objects are assigned to one of several predefined categories and clustering is the task of grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups.

2.1.1 Supervised Learning in Music: Classification

In supervised learning (classification) the objective is to learn a target function f that maps each attribute set of objects x to one of the predefined class categories y [34]. For music classification, the objective is to find f that maps songs to a one of the predefined labels or tags that could be genre or mood among others. Classification by genre attempts to identify music genre and is the most widely studied problem in MIR. Mood classification tries to divide music into different emotional categories like angry, sad or happy.

Music classification could be performed using sound, lyrics or both. However, classification based on sound is the most widely studied approach, although recently mixed and only lyrics approaches have been studied too [11, 19]. Both cases of music classification the labels or

categories are defined mainly by genre and mood.

MIR researches have made important progress in this area using sound and lyrics content of music [16, 10, 17, 22]. However, the classification could include more characteristics like artist, instrument (only for sound), style or similar songs [7]. For example artist identification involves tasks of classifying artist, singer and composer of music and the purpose of instrument recognition is to identify instruments that are played in an interval or segment of the song (made by sound).

2.1.2 Unsupervised Learning in Music: Clustering

Unsupervised learning approaches try to assign a set of objects into groups or clusters, so that the objects in the same cluster are more similar to each other than to those in other clusters. These approaches do not use labels or prior knowledge when training the model.

In music clustering, methods starts with datasets without labels and the objective is to find patterns and similarities between songs in order to make groups of songs with different content or style. For music, unsupervised approaches have been applied less than supervised. However, there are investigations that attempt to identify genres without prior information [32, 3] and there is another attempt that tries to organize lyrics by topic in an unsupervised way [15].

2.1.3 Audio Features

Audio signal is a representation of sound. It has frequencies in a range that human could hear. This audio signal has been used widely for classifying music, but it requires feature extraction in order to provide useful information for classification techniques. Fu et al [7] joins previous attempts to characterize audio features (See figure **2-1**). In these, audio features are divided into two levels: mid-level and low-level features. Low-level features are obtained directly from analysing spectral distribution of the signal [31], but this is not closely related to the properties perceived and appreciated by human listeners. This low level contains timbre and temporal features. Timbre features capture the tonal quality of sound that is related to different instrumentation, and temporal features capture the variation and evolution of timbre over time [7].

Mid-level features provide a closer relationship to the information that is perceived by people, and include rhythm, pitch and harmony.

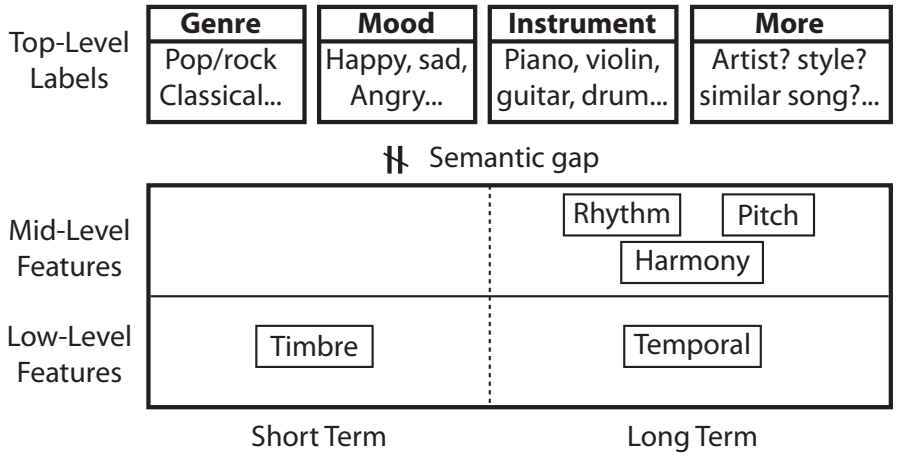

Figure 2-1: Characterization of audio features adapted from [7].

At the top level there are labels that are more related to how people understand and interpret music and these labels are the ones that methods try to discover or assign to music. Between mid-levels and low-levels to the top-level there is a semantic gap and the purpose of music classification is to obtain those labels inferring from low-level and mid-level features [7] in order to be assigned to songs.

In another perspective, mid and low level features could be classified in short-term features and long-term features. In short term features, feature is usually captured in frames with 10-100 ms. duration, whereas long-term features capture the long-term effect and interaction of the signal and is normally extracted from frames with longer durations [7].

2.1.4 Lyrics Features

Lyrics are documents that exhibit specific properties different from traditional text documents. For example lyrics could have rhyming verses, text stylistics particularities, several repetitions, incomplete phrases, spelling mistakes and lyrics are loaded by listeners. However, from lyrics could be extracted rhyme features, text stylistic features and text features.

Rhyme Features

A rhyme is a repetition of similar sounds in two or more words. This similarity could be lexical word endings or with identical or similar ending phonemes. Rhyme is common in songs and has been used in previous lyrics classification tasks [19].

Text Stylistic Features

In lyrics, this refers to special words (e.g., oh, ah, etc.), punctuations and word statistics (e.g., number of unique words, length of words, character frequencies, words per line, etc.). These features have been used in genre identification by Mayer [19].

Text Features

Lyrics could be analysed with classical text information retrieval methods and could be extracted a very rich set of textual features for using in the classification or clustering task. It possible to perform language identification, extract POS features, represent lyrics in Vector Space Model and convert to TF-IDF format. These text features are explained in section 2.3.

2.1.5 Genre and Mood as Categories

Genre and mood are the two mainly approaches for classifying music used by MIR researches. For example in MIREX investigations, they have their own standardization and sound datasets that have been used in different researches of classification and clustering and the main focus is to organize repositories by genre and mood. This section explains characteristics of these two main approaches for organizing music repositories.

Genre

Music genres are not fully defined. These are categories that have arisen through a complex interplay of cultures, artist, and market forces to organize music collections and characterize musicians [31]. The boundaries between genres still remain fuzzy, making their separation difficult. Pachet and Cazaly [25] have shown that there is no general agreement in genre taxonomy and it is not easy to build a general taxonomy.

Genre can also be classified in different ways [22], for example in the case of Christmas carols exist a natural classification based on semantic information, but they may have styles of singing like rock, classical, pop, etc.

However, MIR researches have addressed this problem using pre-defined datasets with genre categories in order to evaluate music classification techniques and compare different approaches on an equal basis [7].

Mood

The purpose of this approach is to classify music into different emotional categories. In MIR there are no standard mood categories and it is difficult to compare different mood classification researches because, in general, these researches use different mood categories. Russell's model [29] is the most popular base model of mood that has been used in MIR researches [10, 12, 16]. This is a dimensional model where emotions are positioned in a multidimensional space. There are two dimensions in it: pleasure and arousal. The original model place 28 emotions in the dimensional space (see figure **2-2**)

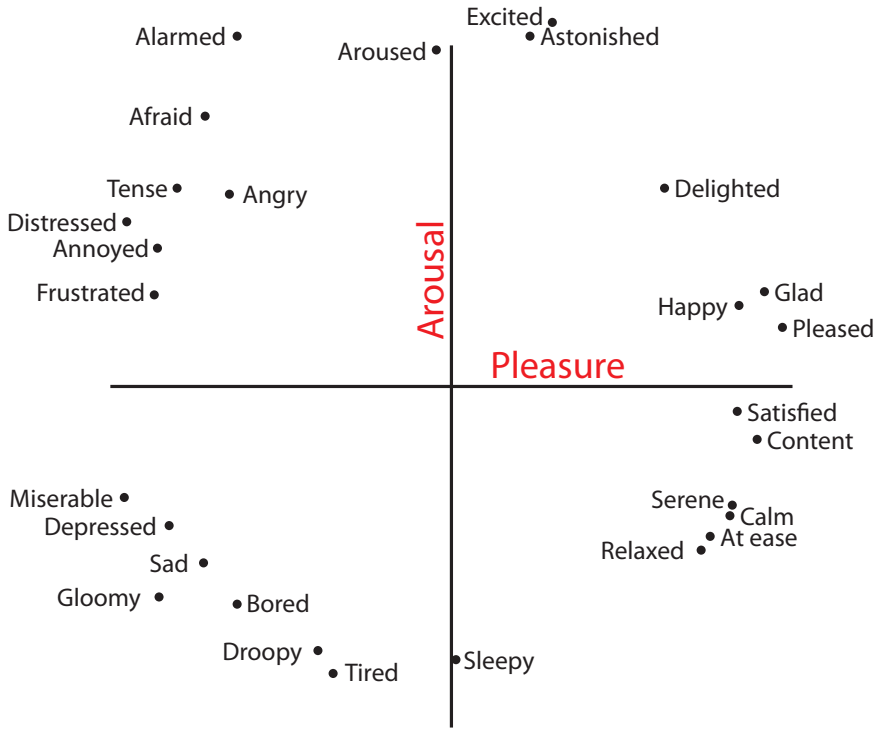

Figure 2-2: Russell's model of mood [29].

2.1.6 Music Categorization

Music categorization attempts to assign labels called tags to music that give some meaning to it. In a standard classification, each group has a single label only. For example, in music genre classification each song is assigned to a single genre class. But this is not necessarily the general case. In general, songs could be related to multiple categories. In music categorization, a song is described by many semantically meaningful tags or labels, which can be anything. For example for genre, mood, style, topic and so on [7].

For lyrics, music annotation could be performed with help of word frequency in clusters by extracting those words and finding labels or categories that summarize the cluster [15].

2.2 Comparison of Approaches for Music Classification

There are some investigations that have undergone classification using only the lyrics and others using the lyrics in combination with sound. In general, these methods have been focused on supervised learning (see table **2-1**).

Table 2-1: Comparative review articles on music classifications

Article	**Type**	**Dataset**	**Features**
Hu 2009 [11]	4 moods	981 lyrics	Words, synonyms, moods relationships. Lyricator as a tool for classification
Laurier 2008 [16]	4 moods	1000 songs (lyrics and sound)	Sound, vector space model, Lucene, latent semantic analysis, support vector machines
Hu 2009 [12]	18 moods	5845 songs (lyrics and sound)	Sound, words, rhymes, semantic and statistical information, spectral clustering
Mayer 2008 [19]	10 genres	397 lyrics	Words, rhymes, semantic and statistical information, support vector machines
Li 2006 [17]	4 groups (like genre)	570 songs (lyrics and sound)	Sound, vector space model, K-means
Kleedorfer 2008 [15]	31 genres	60000 songs (33863 with lyrics in 15 different languages)	Sound, Weighting term frequency, vector space model, non-negative matrix factorization
Neumayer 2007 [21]	54 genres	7554 songs (lyrics and sound)	Sound, vector space model, statistical information, self organizing maps

All these analyses have datasets with information of the class and only one of them uses a non-supervised scheme, but this has a very small dataset which already has classes [17].

Others focus on supervised schemes by selecting characteristics and different analysis techniques.

Four out of the six tested approaches applied learning methods taking into account sound and lyrics. The ones that focused on the mood conclude that the lyrics present a significant contribution regarding the effectiveness of the classification. For those that used the genre, a contribution was presented, but it was not so significant.

For the cases [15, 11, 19, 21], a complete mining process was taken into account: pre-processing steps, analysis of the characteristics, application of techniques and evaluation of results.

2.3 Document Clustering Techniques

In this section, an emphasis on document clustering is given. Lyrics of the songs are text and they are considered documents for clustering techniques. For this analysis, it is necessary to take into account language identification process, document representation and clustering algorithms as based on partitions and hierarchical.

2.3.1 Language Identification in Texts

One of the main steps for working with a multi-language dataset is the language identification. When websites and text repositories with multi-language content are organized it is necessary to perform language identification and lyrics repositories are not an exception. However, lyrics repositories have particular characteristics: misspelling errors, several repetitions and in some cases words with no sense that other repositories did not have.

N-gram frequency technique for language identification is ideally suited for text coming from noisy sources [5]. Human languages have some words which occur more frequently than others. The most common way to express this behaviour is with Zipf's Law: the nth most common word in a human language text occurs with a frequency inversely proportional to n [5]. This law has the implication that there is always a set of words which is more dominating than other words of the language in terms of frequency of usage. In N-gram technique this law could be extended to n-grams extracted from terms and for text from the same language it is possible to obtain similar n-gram frequency distribution. With this information of frequency distribution per language, language detection could work as a classification task. One technique is Naïve Bayes that uses conditional probabilities of observing features (n-grams) in a text to deduce a probability of a text to be written in a given language. Bayes Technique has shown good performance in language identification

using n-grams over others alternative classification methods [8].

2.3.2 Pre-Processing Task for Text Documents

In most of the text categorization methods, the representation of the documents is based on Vector Space Model (VSM) [30]. This method is widely used because of its conceptual simplicity and the appeal of the underlying metaphor of using spacial proximity for semantic proximity. In VSM, every document is a vector with words as a features and the text content is treated as a Bag of Words (BOW).

Bag-of-words (BOW) Features

Bag-of-words (BOW) are collections of unordered words. In these, each unique term is regarded a feature and lyrics are represented as feature vectors. In those vectors, each word has a value that represents importance of the word in the lyrics. This importance could be frequency of the word, normalized frequency, TF-IDF weight or a boolean value indicating presence or absence of the word in the lyrics. TF-IDF is the most widely used technique in text analysis and MIR.

Additionally, to select the set of words that will make up the BOW is an important task. Tasks like converting all words to the same case folding, removing stopwords and word-stemming could be performed in order to obtain better results.

TF-IDF: TF-IDF [30] consists in finding words with high repetition frequency but low frequency in documents. This method allows words with these characteristics to be representative of the categories which were being associated. TF-IDF is computed as:

$$tf \times idf(t,d) = tf(d) \times ln(N/df(t)) \tag{2-1}$$

Where $tf(d)$ is the number of times that term t appears in document d and $df(t)$ is the number of documents in the collection that term t occurs in.

Stopwords: There are words such as articles, prepositions, conjunctions and pronouns that are very frequent and do not provide additional information to content of documents. These words are called stopwords and it is very common to remove these words from documents in information retrieval and document clustering. Eliminating these words also allows a dimensional reduction of the vectors and reduces requiring processing.

Part-of-speech (POS) Features: Part-of-speech tagging is a grammatical tagging of words according to a linguistic category. For example nouns, verbs, pronouns, adjectives,

stem etc. POS tagging has been used in lyrics classification tasks with the help of POS tagging software [36]. Word stemming case is process for reducing inflected words to their stem, base or root form. Word stemming dimensionally reduces vectors because these use stem form of words that is shorter. This technique has shown mixed results (some with improvements, others without improvements) in performance of lyrics classification approaches [11, 19].

2.3.3 Partitional Clustering Algorithms

Partition clustering algorithm splits the data into K partition, where each partition represents a cluster. The partition is done based on certain objective function. Among the techniques based on partitions it was reviewed K-Means, K-Medoids and Fuzzy C-Means [35].

K-Means

K-Means [18] is one of the most popular categorization techniques that exist. This algorithm performs a grouping of documents in K number of clusters. This K must be given in advance by the user. In general the algorithm starts by randomly selecting K features as its centroid, and then assigns all documents to these clusters taking into account some distance measures in the vector space (i.e. the cosine distance) so that each document could be assigned to its closer centroid. Once you have all the documents in the cluster it recalculates the centroid and makes the process again to perform a new grouping. The main problem with this process is that you can converge to local minimum, since everything depends on the way in which initial clusters have been selected [35].

The main objective of this algorithm is to minimize the objective function:

$$arg_{\mathbf{S}} \min \sum_{i=1}^{k} \sum_{x_j \in S_i} \|x_j - \mu_j\|^2 \tag{2-2}$$

Where $\|x_j - \mu_j\|^2$ are the distances from the documents to their centroids. In general terms the K-means algorithm consists of the following steps:

1. Put K points in space, which represent classes or groups that you want to achieve. These are the centroids.

2. Assign each document to its centroid. Document is associated with the centroid that is closer.

3. When all documents have been assigned to a centroid, the centroids are recalculated.

4. Repeat steps (2 and 3) until the centroids do not change.

Every time you run the algorithm, it is possible that results are different, because the initial centroids are selected randomly.

K-Medoids

The K-Means algorithm is sensitive to outliers. This problem could cause that documents with very significant differences to others in the same group, affect the location of centroids. This might cause poor quality clustering results.

To improve this issue, K-medoids proposes the idea of using more central point or document or the document which is the most representative of the group, instead of using the average point between the documents of the cluster [35].

This algorithm requires an initial amount of clusters K number, like in K-means. The main difference with K-means is that K-medoids tries to minimize the sum of distances between the documents in a cluster to their most representative point or document.

This algorithm consists of the following steps:

1. Randomly select K documents as the most representative documents. Each of these corresponds to the Medoid from each of clusters.
2. Assign each document to its closer medoid.
3. A random document different to the medoid is selected and profit calculation is done assuming that this new document is the medoid. If profit is obtained (the distance between documents to its medoid is minimum) a medoid changes for this new one.
4. Steps (2 and 3) are repeated until the centroids do not moved any more.

Fuzzy C-Means

Normally the categorization algorithms generate partitions. In these partitions, each of their items only belongs to one of the clusters. Fuzzy C-Means allows these partnerships to introduce a concept called membership function [35]. This algorithm is intended to minimize the objective function:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^m \|x_i - C_j\|^2, 1 \leq m < \infty \tag{2-3}$$

Where m is a real number greater than 1, U_{ij} is the degree of membership of x_i in the cluster C_j. This algorithm iteratively works where in each iteration will update the u_{ij} (see equation (2-4)) and the C_j (see equation (2-5)).

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - C_j\|}{\|x_i - C_k\|} \right)} \tag{2-4}$$

$$C_j = \frac{\sum_{i=1}^{N} u_{ij}^m x_i}{\sum_{i=1}^{N} u_{ij}^m} \tag{2-5}$$

This algorithm consists of the following steps:

1. Initialize the matrix $U = [u_{ij}]$ matrix. This is initialized randomly and is called $U^{(0)}$.
2. At k-step: calculate vectors $C^{(k)}$ with $U^{(k)}$.
3. Update $U^{(k)}$, $U^{(k+1)}$
4. If $\|U^{(k+1)} - U^{(k)}\| < \epsilon$, then finish the process. Else, return to the step 2.

ϵ controls the rate of change from a model to another. The process stops if the change is not sufficiently representative. At the end, U is a matrix with values between 0 and 1 and contains grades of membership of each of the items with their respective clusters.

2.3.4 Hierarchical Clustering Algorithms

Hierarchical clustering algorithms produce a sequence of nested partitions. At each stage, the algorithm either merges two clusters (agglomerative approach) or splits a cluster in two (divisive approach). The result of the clustering can be displayed in a tree-like structure, called a dendrogram, with one cluster at the top containing all the documents of the collection and many clusters at the bottom with one document each one. If the appropriate level of the dendrogram is chosen it is possible to get a K amount of clusters [23].

Agglomerative approach

This is better known as bottom-up model, which takes each document as a cluster and from these performs unions of more similar pairs. This requires a definition of cluster similarity or distance. At the end it is possible to obtain the amount of desired clusters [33].

The process is as follows:

1. Assign each document to a single cluster.
2. Compute the similarity between all pairs of clusters, e.g., calculate a similarity matrix whose ij_th entry gives the similarity between the i^th and j^th clusters.
3. Merge the two most similar (closest) clusters.
4. Update the similarity matrix with the similarity. between the new cluster and the original clusters.
5. Repeat steps 2 and 3 until only one cluster remains.

Divisive approach

The divisive approach is contrary to the agglomerative approach. This model starts with all objects grouped in a single cluster and each of the iterations is making partitions up to the amount of desired cluster.

One of the popular divisive algorithms is called a Bisecting K-Means [33]. The process for this is as follows:

1. Assign all documents to a single cluster.
2. Pick a cluster to split.
3. Perform the split using K-means to obtain 2 clusters.
4. Repeat step 3 for N times and take the split that produces the clustering with the highest overall similarity.
5. Repeat steps 2, 3 and 4 until the desired number of clusters is reached.

Another algorithm that use this approach is Repeated Bisection (RB) that is implemented in CLUTO software [14].

2.3.5 K Detection

To detect the amount of groups K, measures of cluster quality, used for cluster validation, could be used for testing several clusterings with different K and evaluate which one performs better. For example measures as Entropy, Mean Square Error, Davies-Bouldin and Jaccard. These measures are explained in section 2.4.

However, there is another approach that has been used specifically for finding the appropiate K, and it has shown important results [28]. The main objective is to calculate distortions in the data and select the K value that minimize distortions. In this approach, the following evaluation function $f(K)$ is defined:

$$f(K) = \begin{cases} 1 & \text{if } K = 1 \\ \frac{S_K}{\alpha_K S_{K-1}} & \text{if } S_{K-1} \neq 0, \forall K > 1 \\ 1 & \text{if } S_{K-1} = 0, \forall K > 1 \end{cases} \tag{2-6}$$

$$S_K = \sum_{i=1}^{K} \sum_{x \in C_i} \|x - m_k\|^2 \tag{2-7}$$

$$\alpha_K = \begin{cases} 1 - \frac{3}{4N_d} & \text{if } K = 2 \text{ and } N_d > 1 \\ \alpha_{K-1} + \frac{1-\alpha_{K-1}}{6} & \text{if } K > 2 \text{ and } N_d > 1 \end{cases} \tag{2-8}$$

Where S_K (see (2-7)) is the sum of the cluster distortions when the number of clusters is K, N_d is the number of data set attributes or dimensions and α_K (see (2-8)) is a weight factor.

In this function, $f(K)$ results with less values have more probability of be an optimal K value, but require evaluation with several K, which could be expensive in time.

2.4 Cluster Quality Validation

One of the main purposes in clusters is to find the best grouping in such a way that the objects in clusters are as close as possible, but cluster centres are as separate as possible.

Since research mainly focuses on unsupervised models, some of the measures to evaluate the quality of the results are Entropy, Mean Square Error (MSE), Davis-Bouldin Vadility Index (DBI) and Jaccard Index.

2.4.1 Entropy

Entropy is a probabilistic approach to measure the quality of a cluster [27]. It allows us to evaluate the probability distribution of class labels in each cluster. The probability of class i in cluster j can be estimated by the proportion of occurrences of class label i in cluster j. The formal definition is:

$$Entropy(C_j) = -\sum_{j=1}^{k} Pr_i\left(C_j\right) \log_2 Pr_i\left(C_j\right) \tag{2-9}$$

Where $Pr_i\left(C_j\right)$ is the probability that C_j object belongs to the i class. The total entropy is calculated using:

$$Entropy_{total}(C_j) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} * entropy(C_i) \tag{2-10}$$

Where D_i is the size of the cluster i and D is the total number of objects in the dataset. With this measure lower value means better categorization.

2.4.2 Mean Square Error (MSE)

Partition based techniques attempt to minimize the existing variance within clusters. This technique computes the variance between clusters so as good partition based clustering algorithms should minimize the MSE [27]. MSE is calculed using:

$$\text{MSE} = \frac{1}{N} \sum_{j=1}^{k} \sum_{x \in C_j} \|x - m_k\|^2 \tag{2-11}$$

Where K is the number of clusters, m_k is the mean of cluster C_j and N is the total number of objects.

2.4.3 Davis-Bouldin Vadility Index (DBI)

This is an internal evaluation scheme, where the validation of how well the clustering has been done is made using quantities and features inherent to the dataset. To measure DBI, first it is necessary to define the spread between clusters C_i and C_j [27] (See equation (2-12))

$$R_{i,j} = \frac{e_i + e_j}{D_{ij}} \tag{2-12}$$

Where e_i and e_j are the average dispersion of clusters C_i and C_j, and D_{ij} is the Euclidean distance between C_i and C_j. e values are calculated with the following formula:

$$e_i = \frac{1}{N_i} \sum_{x \in C_i} \|x - m_k\|^2 \tag{2-13}$$

Where m_k is the center of the cluster C_k.

Now it is possible to obtain R_k terms for each cluster C_k with the following formula:

$$R_k = max_{i \neq j} R_{i,j} \tag{2-14}$$

Finally the DB index is defined as:

$$DB(k) = \frac{1}{K} \sum_{k=1}^{K} R_k \tag{2-15}$$

This measure takes into account the dispersion of the cluster and the distance between centres of each cluster. This means that separate and compact clusters are preferred.

2.4.4 Jaccard Index

This measurement could be used to find similarities between two different techniques for categorization. For this the two techniques should have the same amount of K clusters. The way in which it is defined is the following [4]:

Given a pair of clusterings x_1^c and x_2^c, a is the number of objects that belong to the same cluster in x_1^c and x_2^c, b is the number of objects that belong to x_1^c but not to x_2^c and c is the number of objects that belong to x_2^c but not to x_1^c. Taking into account these definitions, Jaccard index between x_1^c and x_2^c is calculated with:

$$J(x_1^c, x_2^c) = \frac{a}{a+b+c} \tag{2-16}$$

The result is a number between 0 and 1. Greater values mean more similarity between categories that were evaluated.

2.5 Topic Detection

Topic detection and tagging is the last step in a clustering process. After identifying groups it is necessary to describe these groups with words or short phrases in order to be shown to website users and to be useful in the website.

Anaya et al [2] attempts to discover and describe topics with the help of terms and their co-occurrences, disregarding stopwords, and thus showing effectiveness. In this case, association rules could help to discover the rules of co-occurrence that could bring important information to an expert to help in the process of tagging the clusters.

2.5.1 Association Rules Definition

The problem of find association rules is defined as follows [1]: Let $I = \{i_1, i_2, ..., i_m\}$ being a set of literals called items. Let D a set of transactions, where each transaction T is a set of items such as $t \subseteq I$ and each transaction t in D has a unique transaction ID. In this case $TcontainsX$, a set of some items in I, if $X \subseteq T$. A rule is defined as an implication of the form $X \implies Y$, where $X \subseteq I$, $Y \subseteq I$, and $X \cap Y = \emptyset$. The rule $X \implies Y$ holds in the transaction set D with confidence c if $c\%$ of transactions in D that contain X also contain Y. The rule $X \implies Y$ has support s in the transaction set D if $s\%$ of transactions in D

contain $X \cup Y$.

2.5.2 Apriori Algorithm

Apriori Algorithm [1] is a classical approach to find frequent itemsets in a database. This Algorithm uses a "bottom up" approach, where frequent subsets are extended one item at a time.

In the first step, it counts the support of individual items and determines which of them are equal or pass some support threshold. In each subsequent step, it starts with a seed set of itemsets found. It uses this seed set for generating new potentially large itemsets, called candidate itemsets, and counts the actual support for these candidate itemsets during the steps over the data. At the end of the process, it determines which of the candidate itemsets are equal or pass support threshold, and it becomes the seed for the next step. This process continues until no new large itemsets are found.

2.5.3 Fp-Growth Algorithm

Fp-Growth Algorithm [9] is another approach to find frequent itemsets. Fp-growth returns the same itemsets that apriori does.

This approach has two main steps. The first step builds a compact data structure called the FP-tree and the second step extracts frequent itemsets directly from the FP-tree. This approach works faster than Apriori [9].

2.6 Summary

In this chapter the background of techniques and concepts that are used in models for grouping and extracting knowledge from music repositories and text repositories was presented. MIR methods was reviewed taking into account sound and lyrics. Document clustering techniques and evaluation methods that help in the process of cluster lyrics information was investigated and the main steps in text clustering process of a multi-language dataset was explained.

3 Lyrics Categorization Model

In this chapter, a model of lyrics categorization based on clustering is presented. This model uses pre-processing steps of text data mining for representing the songs, techniques from data mining that help in the process of clustering and topic detection techniques to complete the steps of the process. A standard process in text mining problems presents a scheme of steps [24], which may be appropriate for the issue of clustering lyrics information. The steps are: pre-processing, clustering, evaluation and topic detection. In our model some sub-steps were added like language identification, cleaning html and extract POS features in preprocessing step, estimating the amount of groups and evaluating different validation measures for identifying the clustering algorithm that performs better in the clustering and evaluation step and extracting top words and association rules in the topic detection step (see figure **3-1**).

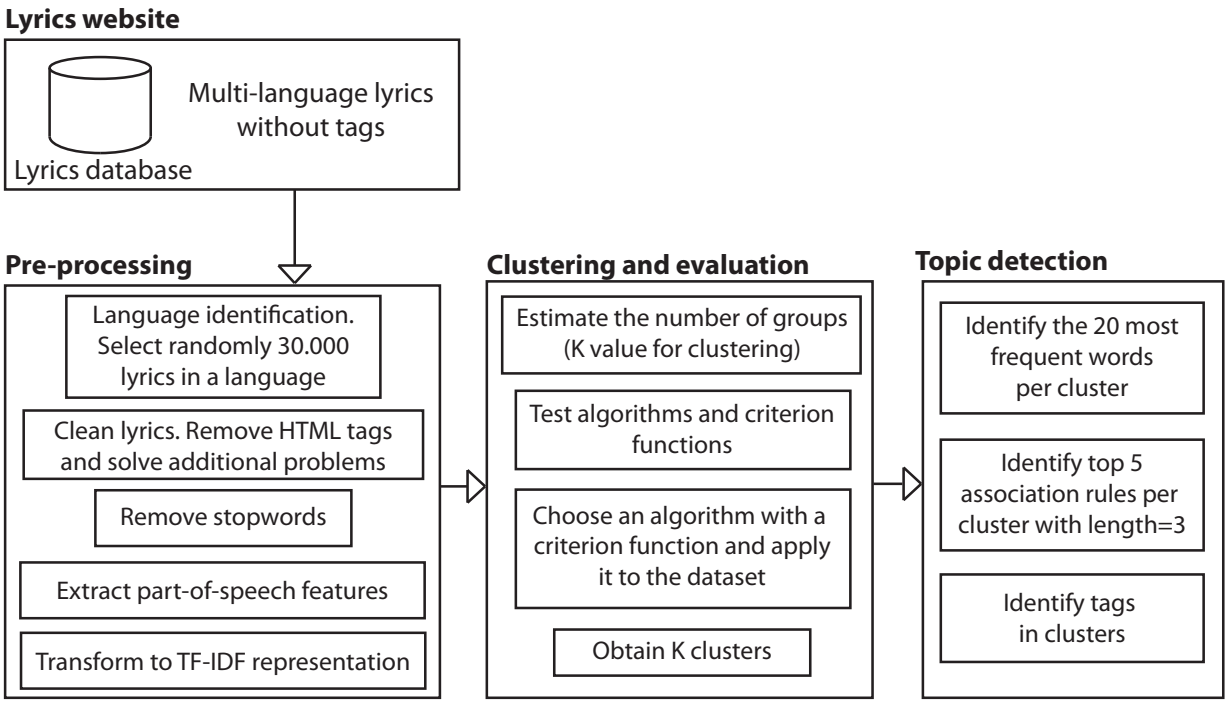

Figure 3-1: Lyrics clustering and tagging process.

3.1 Dataset

The dataset selected for the work was taken from a lyrics website with multi-language lyrics[1] with around 280.000 lyrics. This website has a search engine and offer to its users different sections in the left part of the results page. Lyrics were loaded by website users around the world. With clustering data would be possible to add a new search options to improve user experience. This original dataset did not have any information for guess labels or tags.

3.2 Pre-Processing

Some researches have found problems when pre-processing lyrics [11, 19]. For example, they have faced problems of many repetitions in the case of the chorus, spelling errors, phrases such as "repeat chorus x3", attributions of authorship, etc. To fix this, in cases of small data sets, they have made corrections one by one, but in case of large datasets, they have used regular expressions that help cleaning data. In our case of lyrics clustering, the following steps of the pre-processing for lyrics are used: language identification, cleaning dataset, stopwords removal and feature selection using stemming version of datasets and TF-IDF representation.

3.2.1 Language Identification

Lyrics are a noisy source and n-grams are well suited for detecting language in this case. With Naïve Bayes it is possible to obtain a good language probability [8]. That is the combination that was used for detect language. To do that, a library in java was used [20]. From the original dataset 30.000 lyrics with probability of Spanish over 99% and 30.000 lyrics with probability of English over 99% were selected randomly. Both groups with lyrics length over 500 characters.

3.2.2 Cleaning Dataset

As the dataset is in HTML, it is necessary to remove all HTML tags. In the lyrics, some additional problems with accents, special characters, spelling and encoding were identified. For cleaning this dataset and extracting the text, some regular expressions, replacements, spelling corrections and encoding standardization were applied in both datasets, English and Spanish.

[1] see: http://www.albumcancionyletra.com

3.2.3 Removing Stopwords

In order to reduce dimensionality, it is common to remove stopwords and select relevant terms. For stopwords removal, a list of standard Spanish and English stopwords was started with, adding words like "ooh", "chorus", "bis" and other words that were found with many repetitions in lyrics datasets and that did not provide additional information in clustering process. Words with two or less characters and words that are not in at least in some amount of lyrics were eliminated for preventing spelling mistakes.

3.2.4 Feature Selection

Both clean datasets, in English and Spanish, were divided into two: one with lemmatization using freeling software [26], and the other with the original texts. All four datasets were represented in TF-IDF format.

TF-IDF Weighting.

TF-IDF in this lyrics dataset was computed as:

$$tf \times idf(t,d) = tf(d) \times ln(N/df(t)) \tag{3-1}$$

Where $tf(d)$ is the number of times that term t appears in lyrics d and $df(t)$ is the number of lyrics in the collection that term t occurs in.

Finally, for clustering process Spanish datasets remain with 24.409 different terms (dimensionality) and English datasets with 21.286 different terms.

3.3 Clustering and Evaluation

Previous results in document clustering have shown that hierarchical algorithms like partitional clustering algorithms are well-suited for clustering large document datasets due to their relative low computational requirements [37]. For this work K-Means and Repeated Bisection algoritms were used and to achieve comparable results, the same validation measures were applied.

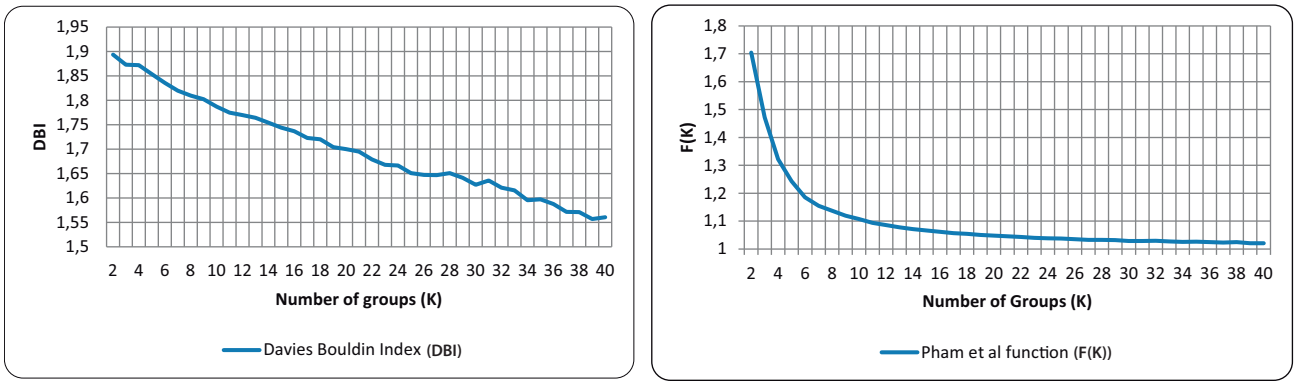

Figure 3-2: K estimation for Spanish without stemming dataset.

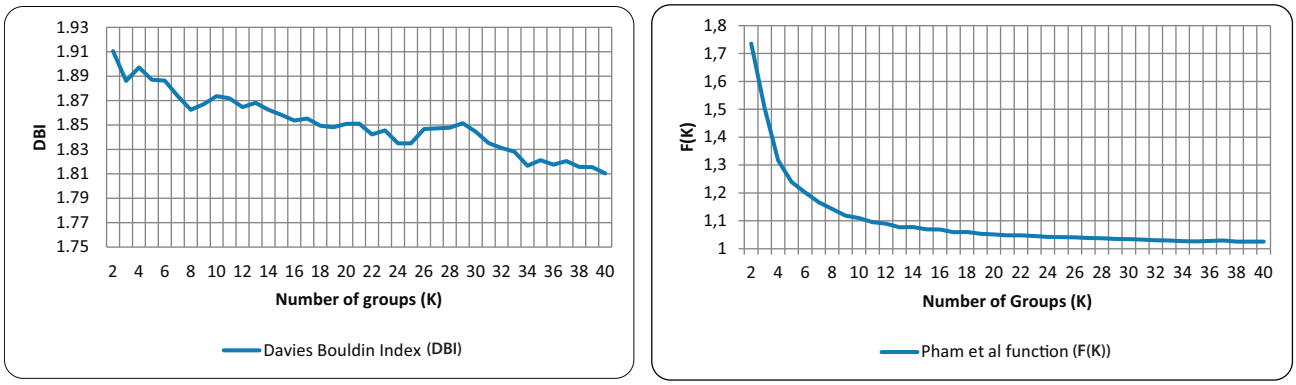

Figure 3-3: K estimation for English without stemming dataset.

K-means is the most popular partitional algorithm and has been used widely to compare different results of clustering. For this algorithm, a standard implementation was developed. For Repeated Bisection, CLUTO software[2] with its two implementations RB and RBR was used.

3.3.1 K Estimation

For detecting the amount of clusters different values of K were tested. With small values of K, results are clusters with mixtures of multiple topics and increasing K, validation measures perform better [15] but there is not an optimal K value (see figures **3-2**, **3-3**). For an approximation of K, Pham et al technique [28] was used and the K value was chosen where the graph starts stabilization (around K=20 for both languages).

[2]see: http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview

Criterion Function	Optimazition Function	
$\mathcal{I}_1$	maximize $\sum_{i=1}^{k} \frac{1}{n_i} \left(\sum_{v,u \in S_i} \text{sim}(v,u) \right)$	(1)
$\mathcal{I}_2$	maximize $\sum_{i=1}^{k} \sqrt{\sum_{v,u \in S_i} \text{sim}(v,u)}$	(2)
$\mathcal{E}_1$	minimize $\sum_{i=1}^{k} n_i \frac{\sum_{v \in S_i, u \in S} \text{sim}(v,u)}{\sqrt{\sum_{v,u \in S_i} \text{sim}(v,u)}}$	(3)
$\mathcal{G}_1$	minimize $\sum_{i=1}^{k} \frac{\sum_{v \in S_i, u \in S} \text{sim}(v,u)}{\sum_{v,u \in S_i} \text{sim}(v,u)}$	(4)
$\mathcal{G}'_1$	minimize $\sum_{i=1}^{k} n_i^2 \frac{\sum_{v \in S_i, u \in S} \text{sim}(v,u)}{\sum_{v,u \in S_i} \text{sim}(v,u)}$	(5)

Figure 3-4: Clustering criterion functions from CLUTO.

3.3.2 Evaluation of Clustering Algorithms

For choosing the best algorithm, three different clustering algorithms were evaluated: K-Means with random initialization, Repeated Bisections (RB) and Repeated Bisections with Refinements (RBR).

With the detected K, repeated bisections algorithms were applied and compared with the standard K-means results. CLUTO has two implementations of repeated bisections: RB and RBR. The main difference is that RBR applies an additional global optimization that RB does not apply. Additionally, in RB and RBR, it is possible to apply some optimization functions [14]. These functions optimize clustering solutions based on some criteria. So, given a particular clustering criterion function C, the clustering problem is to compute a k-way clustering solution so that the value of C is optimized [37]. Our study involves a total of five different criteria function that are shown in figure **3-4**.

K-means changes results in every execution because centroids were selected randomly. To obtain this result, K-means algorithm was applied 36 times and mean and standard deviation of the results were calculated.

Results with original dataset in Spanish using DBI are in table **3-1** and English results with DBI are in table **3-3**. Results using Inter similarity/Intra similarity, weighted by amount of objects in clusters, for Spanish are in table **3-2** and for English are in table **3-4**.

Table 3-1: Spanish results with K = 20 using DBI (less is better)

	Max-I1	Max-I2	Min-E1	Min-G1	Min-G1'
RB	1,8387	1,8855	1,9169	1,8844	1,9088
RBR	1,8648	1,8850	1,8869	1,8583	1,8860
RB (POS)	1,9190	1,9176	1,9221	1,9232	1,9202
RBR (POS)	1,9177	1,8885	1,8864	1,9045	1,8799

K-Means: 1,9257 +/- 0.007; **K-Means (POS)**: 1,9258 +/- 0.012

Table 3-2: Spanish results with K = 20 using Inter similarity/Intra similarity (more is better)

	Max-I1	Max-I2	Min-E1	Min-G1	Min-G1'
RB	1,1024	1,1010	1,0881	1,0929	1,0910
RBR	1,1101	1,1085	1,0971	1,1091	1,0997
RB (POS)	1,1130	1,1142	1,1033	1,1044	1,1063
RBR (POS)	1,1249	**1,1253**	1,1177	1,1198	1,1194

K-Means: 1,1024 +/- 0.002; **K-Means (POS)**: 1,1218 +/- 0.003

Table 3-3: English results with K = 20 using DBI (less is better)

	Max-I1	Max-I2	Min-E1	Min-G1	Min-G1'
RB	2,1568	1,9020	1,9226	2,6443	1,9019
RBR	2,2022	1,8873	1,8797	2,7912	1,8821
RB (POS)	2,0987	1,8842	1,9090	1,9975	1,8948
RBR (POS)	2,2592	1,8737	1,8692	1,9208	1,8610

K-Means: 1,9178 +/- 0.013; **K-Means (POS)**: 1,9035 +/- 0.029

Table 3-4: English results with K = 20 using Inter similarity/Intra similarity (more is better)

	Max-I1	Max-I2	Min-E1	Min-G1	Min-G1'
RB	1,1117	1,1102	1,0996	1,1065	1,1015
RBR	1,1241	1,1212	1,1114	1,1233	1,1130
RB (POS)	1,1207	1,1249	1,1110	1,1154	1,1135
RBR (POS)	1,1366	**1,1396**	1,1266	1,1329	1,1279

K-Means: 1,1167 +/- 0.002; **K-Means (POS)**: 1,1365 +/- 0.005

RBR was the algorithm selected for clustering lyrics in both cases. This algorithm presents better results in most cases. POS dataset performs better in the majority of cases and I_2 was the criterion function that offers better results in RBR (POS) execution.

K-means performance was useful, but this algorithm creates groups with similar amount of objects and for some of them it is difficult to identify tags. Repeated bisections algorithms creates several groups with different length, and some of them were well separated and easy to tag.

3.4 Topic Detection

The set of most frequent words in each cluster could help in the process of making an approximation to the topic [2]. This set of words provides information that is useful to identify cluster's tags. In this case, the 20 most frequently words and the top 5 association rules per language were identified. For discovering association rules the FP-Growth algorithm was used with min length = 3 indicating support value for every rule (see table **3-5** for spanish dataset and table **3-6** for english dataset).

With all steps completed it is possible to show the model used for cluster and tag the multi-language lyrics dataset (see figure **3-5**).

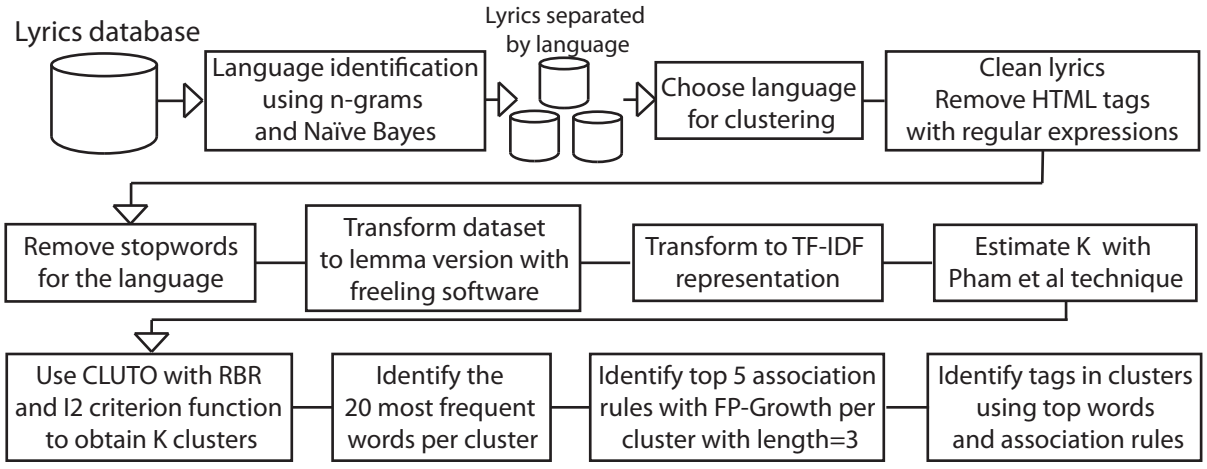

Figure 3-5: Lyrics clustering and tagging model.

Table 3-5: Spanish final clusters using RBR(POS) and I2 as criterion function

Cluster Id	**Top Words in TF-IDF**	**Top Itemsets**	**Tags**
Cluster 1 (Size: 731, Inter:0,2688, Intra:0,1039)	enamorar, amor, estar, corazón, querer, haber, decir, mujer, amar, tener, vida, poder, solo, ese, saber, ver, niño, dar, tanto, amigo.	enamorar,amor,querer (0.438) enamorar,querer,estar (0.409) enamorar,amor,estar (0.379) enamorar,querer,tener (0.372) enamorar,amor,corazon (0.360)	In love, Love, Heart
Cluster 2 (Size: 750, Inter:0,2629, Intra:0,1077)	adiós, doler, amor, decir, dolor, corazón, haber, amar, llorar, nuestro, aunque, tanto, querer, perder, saber, poder, alma, quedar, vida, dejar.	adios,amor,decir (0.375) adios,amor,haber (0.356) adios,amor,querer (0.332) adios,decir,haber (0.309) adios,amor,poder (0.304)	Good bye, Love, Pain
Cluster 3 (Size: 756, Inter:0,2437, Intra:0,0830)	vos, sos, tic, tac, estar, tenes, querer, co, queres, poder, corazón, solo, amor, ver, dar, saber, tener, siempre, pasar, ese.	vos,estar,querer (0.283) vos,estar,poder (0.279) vos,estar,tener (0.254) vos,querer,poder (0.238) vos,poder,tener (0.221)	Not identified
Cluster 4 (Size: 1259, Inter:0,2548, Intra:0,1143)	amar, amor, querer, decir, nadie, vida, corazón, saber, amarar, solo, haber, tanto, poder, tener, mujer, dar, dejar, ese, estar, siempre.	amar,amor,querer (0.459) amar,amor,haber (0.372) amar,amor,tener (0.357) amar,querer,haber (0.351) amar,amor,poder (0.345)	Love
Cluster 5 (Size: 852, Inter:0,2489, Intra:0,1114)	contigo, conmigo, querer, estar, amor, tener, vivir, solo, vida, decir, poder, ver, dar, haber, siempre, sentar, noche, saber, amar, quedar.	contigo,querer,estar (0.488) contigo,querer,amor (0.435) contigo,querer,tener (0.408) contigo,querer,poder (0.393) contigo,estar,amor (0.393)	Love, Declaration of love
Cluster 6 (Size: 1215, Inter:0,2447, Intra:0,1138)	olvidar, poder, amor, recordar, querer, amar, dejar, decir, haber, aunque, pensar, tanto, vida, corazón, saber, volver, recuerdo, estar, seguir, vivir.	olvidar,querer,amor (0.403) olvidar,querer,poder (0.390) olvidar,amor,poder (0.355) olvidar,querer,haber (0.347) olvidar,querer,estar (0.328)	Love, Obscurity, Memories

Cluster Id	Top Words in TF-IDF	Top Itemsets	Tags
Cluster 7 (Size: 1049, Inter:0,2395, Intra:0,1041)	llorar, sufrir, amor, querer, pena, corazón, dolor, haber, solo, dejar, decir, saber, ver, lagrimar, estar, vida, amar, olvidar, pensar, morir.	llorar,querer,amor (0.365) llorar,querer,haber (0.350) llorar,amor,haber (0.333) llorar,querer,estar (0.285) llorar,querer,ver (0.284)	Pain, Tears, Love
Cluster 8 (Size: 977, Inter:0,2199, Intra:0,0626)	bailar, mover, ritmo, gozar, pa, menear, cumbia, baile, fiesta, gustar, pegao, cuerpo, colita, cheque, sua, poner, cintura, querer, ver, ese.	bailar,querer,ver (0.288) bailar,querer,tener (0.275) bailar,querer,dar (0.247) bailar,querer,ese (0.233) bailar,tener,ver (0.230)	Dance
Cluster 9 (Size: 1121, Inter:0,2369, Intra:0,1118)	volver, querer, ver, amor, poder, haber, estar, saber, dejar, vida, decir, vivir, esperar, perder, sentir, día, tener, solo, pensar, pasar.	volver,querer,poder (0.354) volver,querer,haber (0.344) volver,querer,estar (0.331) volver,querer,ver (0.329) volver,querer,tener (0.320)	Love, Reconcilia-tion
Cluster 10 (Size: 706, Inter:0,2194, Intra:0,0817)	loco, gustar, na, querer, estar, ese, decir, tener, dar, volver, amor, zancudo, haber, saber, ver, nene, mirar, solo, corazón, dejar.	querer,gustar,tener (0.215) querer,loco,estar (0.201) querer,tener,haber (0.200) querer,tener,loco (0.194) querer,tener,decir (0.191)	Not identified
Cluster 11 (Size: 2098, Inter:0,2046, Intra:0,1102)	amor, corazón, querer, morir, vida, dar, solo, dolor, haber, poder, tener, vivir, saber, decir, sentar, estar, tanto, amar, nuestro, siempre.	amor,querer,corazon (0.380) amor,querer,haber (0.343) amor,querer,tener (0.331) amor,corazon,haber (0.326) amor,querer,poder (0.323)	Love, Heart
Cluster 12 (Size: 947, Inter:0,1835, Intra:0,0724)	dios, señor, usted, za, jesús, santo, cristo, gloria, don, gracia, haber, dar, poder, alabar, bendecir, vida, amor, padre, hijo, estar.	dios,haber,estar (0.189) dios,haber,tener (0.187) dios,haber,poder (0.183) dios,haber,querer (0.182) haber,querer,tener (0.172)	God, Religion

Cluster Id	**Top Words in TF-IDF**	**Top Itemsets**	**Tags**
Cluster 13 (Size: 1130, Inter:0,1773, Intra:0,0815)	cantar, canción, luna, flor, maría, querer, noche, corazón, amor, estrella, canto, haber, sol, vida, voz, bello, dar, ese, tener, mirar.	cantar,querer,haber (0.197) cantar,querer,amor (0.184) cantar,querer,cancion (0.183) cantar,querer,tener (0.176) cantar,cancion,haber (0.168)	Sing
Cluster 14 (Size: 1552, Inter:0,1673, Intra:0,0660)	pa, mami, dar, gato, poner, eh, hey, ma, vamo, tra, tener, papi, nene, querer, ese, meter, ver, tirar, duro, gustar.	tener,querer,dar (0.385) tener,querer,ver (0.348) tener,querer,decir (0.350) tener,querer,ese (0.335) tener,querer,haber (0.303)	Not identified
Cluster 15 (Size: 892, Inter:0,1680, Intra:0,0679)	rap, mierda, os, puta, bla, rock, hop, hip, estilo, rima, perro, tener, ese, pa, mc, culo, roll, haber, dar, joder.	tener,haber,estar (0.352) tener,haber,ver (0.343) tener,haber,dar (0.339) tener,estar,dar (0.332) tener,haber,querer (0.330)	Rap
Cluster 16 (Size: 1974, Inter:0,1789, Intra:0,0953)	beso, cuerpo, besar, piel, noche, labio, boca, querer, mujer, amor, ah, dar, fuego, ojo, sentir, solo, tener, mio, ese, sentar.	querer,amor,tener (0.216) querer,amor,beso (0.209) querer,amor,haber (0.188) querer,amor,estar (0.188) querer,amor,dar (0.187)	Love, Kiss
Cluster 17 (Size: 2987, Inter:0,1817, Intra:0,1048)	decir, poder, querer, saber, pensar, haber, algo, nadie, estar, entender, solo, igual, ver, hablar, mejor, dejar, tener, importar, cambiar, mal.	querer,decir,poder (0.304) querer,decir,haber (0.291) querer,decir,estar (0.282) querer,decir,tener (0.281) querer,poder,haber (0.279)	Not identified
Cluster 18 (Size: 3252, Inter:0,1799, Intra:0,1066)	solo, estar, sueño, vida, quedar, poder, vivir, haber, encontrar, día, siempre, perder, esperar, tanto, ver, seguir, querer, sentar, pensar, junto.	estar,poder,querer (0.232) haber,estar,poder (0.229) haber,estar,querer (0.227) estar,poder,solo (0.222) haber,estar,solo (0.220)	Loneliness

Cluster Id	Top Words in TF-IDF	Top Itemsets		Tags
Cluster 19 (Size: 2248, Inter:0,1654, Intra:0,0837)	volar, sol, mar, luz, cielo, azul, estrella, viento, noche, agua, ver, tierra, ojo, brillar, poder, amor, luna, haber, sueño, querer.	haber,ver,poder querer,ver,poder haber,querer,ver haber,querer,poder haber,querer,amor	(0.118) (0.115) (0.112) (0.110) (0.109)	Firmament, Sky
Cluster 20 (Size: 3504, Inter:0,1323, Intra:0,0725)	haber, amigo, tener, bueno, gente, malo, ese, niño, calle, matar, dar, hombre, venir, estar, mucho, decir, pasar, hijo, madre, viejo.	haber,tener,querer haber,tener,estar haber,tener,ver haber,tener,decir haber,tener,poder	(0.219) (0.210) (0.203) (0.203) (0.197)	Not identified

Table 3-6: English final clusters using RBR(POS) and I2 as criterion function

Cluster Id	Top Words in TF-IDF	Top Itemsets		Tags
Cluster 1 (Size:492, Inter:0,3011, Intra:0,0683)	rock, roll, baby, beat, party, say, metal, music, high, body, ready, everybody, beethoven, move, night, want, way, stop, dance, boom.	rock,just,say rock,roll,just rock,just,make rock,roll,say rock,say,make	(0.195) (0.193) (0.185) (0.168) (0.166)	Rock and Roll, Dance, Party
Cluster 2 (Size:812, Inter:0,2654, Intra:0,0939)	tonight, alright, night, feel, make, baby, turnaround, light, cause, take, need, just, theres, eye, say, life, want, hold, see, infinity.	tonight,just,see tonight,just,make tonight,just,take tonight,just,feel tonight,just,night	(0.214) (0.203) (0.188) (0.185) (0.182)	Night
Cluster 3 (Size:745, Inter:0,2529, Intra:0,0748)	dance, shake, music, move, body, party, floor, drop, everybody, night, whine, baby, dj, feel, rhythm, people, just, chance, beat, disco.	dance,just,feel dance,just,see dance,just,say dance,just,take dance,just,night	(0.201) (0.197) (0.189) (0.182) (0.181)	Dance, Party

Cluster Id	Top Words in TF-IDF	Top Itemsets		Tags
Cluster 4 (Size:1537, Inter:0,2591, Intra:0,0938)	baby, girl, crazy, just, make, want, tell, give, need, say, love, way, feel, babe, see, take, think, good, never, little.	baby,just,make baby,just,see baby,just,say baby,just,way baby,just,want	(0.288) (0.261) (0.258) (0.237) (0.235)	Girl, Loneliness, Love
Cluster 5 (Size:1130, Inter:0,2411, Intra:0,0890)	girl, baby, sexy, want, just, say, make, world, cause, good, little, need, tell, see, way, bad, crazy, look, thing, feel.	girl,just,say girl,just,make girl,just,see girl,baby,make girl,baby,sexy	(0.354) (0.334) (0.316) (0.304) (0.283)	Girl, Woman
Cluster 6 (Size:813, Inter:0,2413, Intra:0,0947)	believe, beautiful, see, something, never, love, say, tell, otherside, make, feel, way, need, leave, lie, find, dream, day, want, still.	believe,see,just believe,just,make believe,see,make believe,see,say believe,just,say	(0.210) (0.188) (0.181) (0.176) (0.169)	Dream, Love
Cluster 7 (Size:770, Inter:0,2399, Intra:0,0929)	fall, stand, apart, never, sky, just, heart, lose, see, feel, world, break, way, away, say, cause, look, take, find, thing.	fall,just,see fall,just,say fall,see,never fall,just,never fall,never,take	(0.178) (0.163) (0.153) (0.146) (0.146)	Broken Hearted
Cluster 8 (Size:696, Inter:0,2322, Intra:0,0774)	rain, sing, happy, christmas, knock, cloud, ring, tiki, sun, bell, day, stormy, smile, umbrella, softly, sunny, refrain, merry, heart, make.	rain,see,say rain,see,just rain,see,day sing,just,life rain,heart,thing	(0.099) (0.095) (0.093) (0.093) (0.089)	Christmas Carols
Cluster 9 (Size:1070, Inter:0,2312, Intra:0,0736)	nigga, fuck, shit, bitch, niggaz, niggas, wit, ass, gon, fuckin, hoe, verse, hit, money, cause, pop, huh, rap, see, make.	shit,see,just shit,see,make see,just,make shit,just,make shit,see,fuck	(0.358) (0.349) (0.346) (0.342) (0.341)	Racism
Cluster 10 (Size:1151, Inter:0,2207, Intra:0,0965)	away, run, far, love, take, stay, day, walk, never, say, just, hide, feel, fade, way, throw, make, give, leave, try.	away,just,take away,just,say away,just,make away,take,love away,just,see	(0.224) (0.217) (0.192) (0.192) (0.180)	Leave, Love

Cluster Id	Top Words in TF-IDF	Top Itemsets		Tags
Cluster 11 (Size:1127, Inter:0,2162, Intra:0,0961)	nothing, theres, everything, leave, say, take, else, change, never, just, noone, way, lose, something, see, feel, give, matter, cause, find.	nothing,theres,see nothing,just,say nothing,just,see nothing,theres,just nothing,theres,say	(0.190) (0.187) (0.184) (0.182) (0.179)	Not identified
Cluster 12 (Size:1529, Inter:0,2110, Intra:0,0939)	want, need, give, touch, just, feel, really, say, tell, honey, many, baby, free, make, way, cause, never, somebody, show, everything.	want,need,just want,just,see want,just,make want,just,say want,need,see	(0.219) (0.193) (0.192) (0.192) (0.182)	Not identified
Cluster 13 (Size:1239, Inter:0,2084, Intra:0,0915)	wait, home, miss, long, lonely, alone, day, road, way, every, night, just, take, find, see, feel, leave, never, still, walk.	wait,just,see wait,just,take wait,just,say wait,just,make wait,just,day	(0.116) (0.103) (0.101) (0.099) (0.096)	Loneliness
Cluster 14 (Size:878, Inter:0,1972, Intra:0,0690)	lord, god, jesus, thank, king, thy, praise, jah, holy, christ, bobby, glory, hallelujah, save, pray, thou, sing, heaven, faith, give.	god,see,just god,see,say god,just,say god,see,many god,see,world	(0.098) (0.086) (0.086) (0.084) (0.083)	God, Religion
Cluster 15 (Size:1588, Inter:0,2067, Intra:0,0989)	heart, forever, cry, break, goodbye, remember, tear, always, together, never, say, kiss, apart, hold, feel, eye, give, last, make, every.	heart,just,say heart,say,never heart,just,break heart,just,never heart,just,make	(0.159) (0.144) (0.142) (0.141) (0.140)	Not identified

Cluster Id	**Top Words in TF-IDF**	**Top Itemsets**	**Tags**
Cluster 16 (Size:1398, Inter:0,1968, Intra:0,0942)	live, life, alive, anymore, enough, world, die, change, give, just, feel, never, many, way, take, good, day, make, want, see.	live,life,just (0.215) live,life,see (0.193) live,life,make (0.177) live,life,never (0.177) live,life,way (0.175)	Not identified
Cluster 17 (Size:2749, Inter:0,1638, Intra:0,0694)	death, blood, their, die, fire, hell, fight, dead, burn, kill, evil, soul, war, fear, world, life, pain, eye, lie, power.	life,see,die (0.062) life,see,eye (0.069) life,see,take (0.054) life,see,their (0.051) life,see,death (0.052)	Death
Cluster 18 (Size:4275, Inter:0,1789, Intra:0,0993)	say, never, think, try, tell, just, thing, feel, good, make, way, many, find, see, cause, look, take, bad, something, lie.	just,say,make (0.184) just,say,see (0.178) just,say,think (0.167) just,say,way (0.159) just,say,tell (0.157)	Not identified
Cluster 19 (Size:2738, Inter:0,1675, Intra:0,0832)	dream, light, night, fly, sun, sky, shine, day, wind, eye, moon, star, blue, high, world, feel, sea, dark, see, rise.	see,night,light (0.084) night,light,just (0.076) see,night,eye (0.072) night,light,eye (0.072) see,night,day (0.071)	Future, Dreams
Cluster 20 (Size:3263, Inter:0,1278, Intra:0,0651)	big, little, mama, everybody, round, money, woman, say, make, just, work, whoa, around, new, play, city, take, town, good, cause.	just,say,make (0.152) just,say,see (0.145) just,say,take (0.143) just,make,take (0.134) just,make,see (0.128)	Not identified

These results with top words and association rules help in the process of tagging but the tags depends on the person that performs tagging process. In this case, using 20 clusters was possible to identify tags for 15 of them in Spanish and 14 for English dataset.

3.5 Summary

This chapter presented a scheme of steps that could be used for cluster lyrics information. These steps allow to discover tags in music using lyrics information only. Algorithms and methods were tested with big datasets and show that is possible to cluster and tag some of the discovered clusters.

4 Conclusions and Future Work

4.1 Conclusions

Music categorization is an actual trend with a lot research in recent years. Those researches have focused their efforts on classifying datasets mainly composed by sound, but there are mixed cases in which they use lyrics too.

Investigations have been focused primarily on two branches: genre and mood of music. In the part of the genre, lyrics do not provide so much information, whereas in the case of mood, the lyrics present larger contributions. In this work it was found that more groups like feelings and specific topics, not related with mood or genre, could be discovered too using lyrics information.

The feature selection stands as one of the main points in the whole mining process. Dataset using POS reduces dimensionality and improves results in all cases.

Internal measures like Davies Bouldin Index (DBI) estimate clustering quality using averages of worst case scenarios. In the case of lyrics, some clusters are compact and well separated, but others are not. Inter similarity / Intra similarity offer better results for choosing the best option between different algorithms.

Repeated bisections with optimization (RBR) was the algorithm that offers better results in most cases. Additionally, with this algorithm it is possible to bisect clusters in which it is not possible to identify tags.

Our model takes into account all steps in lyrics clustering process, to finally discover topics and assign tags in some clusters. This model shows how it is possible to apply text mining techniques in a problem from the real world with a large amount of data.

Detecting the appropriate number of groups k is not an easy task. This happens because groups could be divided into several different topic groups (like genre, artist, mood, sentiment, topic, etc) with different boundaries. In our results some tags were genre others - moods and topics. This model helps in the process of tagging lyrics but has not given a complete set of tags for music.

The experiment results show how our analysis gives initial results in music tagging. Using 20 clusters allow us to identify tags for 15 of them in Spanish and 14 for English using top words and association rules.

4.2 Future Work

This work could be a starting point for improving automatic tagging process in music using only lyrics information. In our case tag annotation process was made with top words and association rules that help an expert in the process of labelling every group. There are some other techniques that summarize clusters and could help in this process. On the other hand, tags could be related to different categories like genre, mood or topic. Tagging process could be optimized to detect only one of those categories.

Clustering results could be improved by attempting to find sub-clusters in those groups without tags. In those cases, it could be possible that groups need more splitting for performing tag annotation.

Amount of K in these problems is not easy to identify. There are some clustering techniques that could help in find K improving the results.

In the pre-processing step datasets with only some POS information like names, verbs could be tested. This could offer different results in clustering process for performing additional tag annotation.

Additional clustering techniques in depth like fuzzy techniques and test datasets could be applied in other languages too.

In this real environment of a website, the affinity of the tags in lyrics with the songs could be tested. This could be possible by assigning those tags to lyrics in the website and asking users to validate the tags per song.

Bibliography

[1] Agrawal, Rakesh ; Srikant, Ramakrishnan: Fast Algorithms for Mining Association Rules in Large Databases. En: *Proceedings of the 20th International Conference on Very Large Data Bases*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1994 (VLDB '94). – ISBN 1–55860–153–8, p. 487–499

[2] Anaya-Sánchez, Henry ; Pons-Porrata, Aurora ; Berlanga-Llavori, Rafael: A document clustering algorithm for discovering and describing topics. En: *Pattern Recogn. Lett.* 31 (2010), April, Nr. 6, p. 502–510. – ISSN 0167–8655

[3] Barreira, L. ; Cavaco, S. ; Da Silva, J.F.: Unsupervised music genre classification with a model-based approach. En: *Lecture Notes in Computer Science* 7026 LNAI (2011), p. 268–281. – ISBN 9783642247682

[4] Bekkerman, Ron ; Scholz, Martin ; Viswanathan, Krishnamurthy: Improving clustering stability with combinatorial MRFs. En: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA, 2009 (KDD '09). – ISBN 978–1–60558–495–9, p. 99–108

[5] Cavnar, William B. ; Trenkle, John M.: N-Gram-Based Text Categorization. En: *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, 1994, p. 161–175

[6] syan Chen, Ming ; Hun, Jiawei ; Yu, Philip S. ; J, Ibm T. ; Ctr, Watson R.: Data Mining: An Overview from Database Perspective. En: *IEEE Transactions on Knowledge and Data Engineering* 8 (1996), p. 866–883

[7] Fu, Zhouyu ; Lu, Guojun ; Ting, Kai M. ; Zhang, Dengsheng: A Survey of Audio-Based Music Classification and Annotation. En: *Multimedia, IEEE Transactions on* 13 (2011), april, Nr. 2, p. 303 –319. – ISSN 1520–9210

[8] Gottron, Thomas ; Lipka, Nedim: A comparison of language identification approaches on short, query-style texts. En: *Proceedings of the 32nd European conference on Advances in Information Retrieval*. Berlin, Heidelberg : Springer-Verlag, 2010 (ECIR'2010). – ISBN 3–642–12274–4, 978–3–642–12274–3, p. 611–614

[9] Han, Jiawei ; Pei, Jian ; Yin, Yiwen: Mining frequent patterns without candidate generation. En: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. New York, NY, USA : ACM, 2000 (SIGMOD '00). – ISBN 1–58113–217–4, p. 1–12

[10] Hu, Xiao ; Downie, J. S.: Improving mood classification in music digital libraries by combining lyrics and audio. En: *Proceedings of the 10th annual joint conference on Digital libraries*. New York, NY, USA, 2010 (JCDL '10). – ISBN 978–1–4503–0085–8, p. 159–168

[11] Hu, Xiao ; Downie, J. S. ; Ehmann, Andreas F.: Lyric Text Mining in Music Mood Classification. En: *10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, 2009, p. 411–416

[12] Hu, Yajie ; Chen, Xiaoou ; Yang, Deshun: Lyric-based Song Emotion Detection with Affective Lexicon and Fuzzy Clustering Method. En: *Proceedings of ISMIR 2009*, 2009, p. 123–128

[13] Inc., Cisco: Cisco Visual Networking Index: Forecast and Methodology, 2011-2016 / Cisco. 2012. – Informe de Investigación

[14] Karypis, George: CLUTO A Clustering Toolkit / Dept. of Computer Science, University of Minnesota. 2003 (02-017). – Informe de Investigación. Available at http://www.cs.umn.edu˜cluto

[15] Kleedorfer, Florian ; Knees, Peter ; Pohle, Tim: Oh Oh Oh Whoah! Towards Automatic Topic Detection In Song Lyrics. En: Bello, Juan P. (Ed.) ; Chew, Elaine (Ed.) ; Turnbull, Douglas (Ed.): *ISMIR*, 2008. – ISBN 978–0–615–24849–3, p. 287–292

[16] Laurier, Cyril ; Grivolla, Jens ; Herrera, Perfecto: Multimodal Music Mood Classification Using Audio and Lyrics. En: *Proceedings of the 2008 Seventh International Conference on Machine Learning and Applications*. Washington, DC, USA : IEEE Computer Society, 2008 (ICMLA '08). – ISBN 978–0–7695–3495–4, p. 688–693

[17] Li, Tao ; Ogihara, Mitsunori ; Zhu, Shenghuo: Integrating Features from Different Sources for Music Information Retrieval. En: *Data Mining, IEEE International Conference on* 0 (2006), p. 372–381. – ISSN 1550–4786

[18] MacQueen, J. B.: Some Methods for Classification and Analysis of MultiVariate Observations. En: Cam, L. M. L. (Ed.) ; Neyman, J. (Ed.): *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability* Vol. 1, University of California Press, 1967, p. 281–297

[19] Mayer, Rudolf ; Neumayer, Robert ; Rauber, Andreas: Rhyme and style features for musical genre classification by song lyrics. En: *In Proceedings of the 9th International Conference on Music Information Retrieval*, 2008

[20] Nakatani, Shuyo: Language Detection Library for Java / Cybozu Labs, Inc. 2011. – Informe de Investigación

[21] Neumayer, Robert ; Rauber, Andreas: Integration of text and audio features for genre classification in music information retrieval. En: *in Proceedings of the 29th European Conference on Information Retrieval (ECIR 2007)*, 2007, p. 724–727

[22] Neumayer, Robert ; Rauber, Andreas: Multi-modal music information retrieval - visualisation and evaluation of clusterings by both audio and lyrics. En: *In Proceedings of the 8th Conference Recherche d'Information AssistÃ©e par Ordinateur (RIAO 2007)*, ACM, 2007

[23] Oikonomakou, Nora ; Vazirgiannis, Michalis: A Review of Web Document Clustering Approaches. En: Maimon, Oded (Ed.) ; Rokach, Lior (Ed.): *The Data Mining and Knowledge Discovery Handbook*. Springer, 2005. – ISBN 0–387–24435–2, p. 921–943

[24] Ozgur, Arzucan: *Supervised and Unsupervised Machine Learning Techniques For Text Document Categorization*. Istanbul, Turkey, Department of Computer Engineering, Bogazici University, Tesis de Grado, 2002

[25] Pachet, Franois ; Cazaly, Daniel: A Taxonomy of Musical Genres. En: *In Proc. Content-Based Multimedia Information Access (RIAO 2000)*, 2000

[26] Padró, Lluís ; Stanilovsky, Evgeny: FreeLing 3.0: Towards Wider Multilinguality. En: *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*. Istanbul, Turkey, May 2012

[27] Patel, Dipak ; Zaveri, Mukesh: A Review on Web Pages Clustering Techniques. En: Wyld, David C. (Ed.) ; Wozniak, Michal (Ed.) ; Chaki, Nabendu (Ed.) ; Meghanathan, Natarajan (Ed.) ; Nagamalai, Dhinaharan (Ed.): *Trends in Network and Communications* Vol. 197. Springer Berlin Heidelberg, 2011. – ISBN 978–3–642–22543–7, p. 700–710

[28] Pham, S S; Nguyen C D.: Selection of K in K -means clustering. En: *Proceedings of the Institution of Mechanical Engineers* Vol. 219, 2005, p. 103

[29] Russell, J. A.: A circumplex model of affect. En: *Journal of Personality and Social Psychology* 39 (1980), p. 1161–1178

[30] Salton, G. ; Buckley, C.: Term-weighting approaches in automatic text retrieval. En: *Information Processing and Management* 24 (1988), Nr. 5, p. 513–523. – cited By (since 1996) 1952. – ISSN 03064573

[31] Scaringella, N. ; Zoia, G. ; Mlynek, D.: Automatic genre classification of music content: a survey. En: *Signal Processing Magazine, IEEE* 23 (2006), march, Nr. 2, p. 133 –141. – ISSN 1053–5888

[32] Shao, Xi ; Xu, Changsheng ; Kankanhalli, M.S.: Unsupervised classification of music genre using hidden Markov model. En: *Multimedia and Expo, 2004. ICME '04. 2004 IEEE International Conference on* Vol. 3, 2004, p. 2023 – 2026 Vol.3

[33] Steinbach, M. ; Karypis, G. ; Kumar, V. *A comparison of document clustering techniques*. 2000

[34] Tan, Pang-Ning ; Steinbach, Michael ; Kumar, Vipin: *Introduction to Data Mining*. 1. Addison Wesley, Mai 2005. – ISBN 0321321367

[35] Velmurugan, T. ; Santhanam, T.: A Survey of partition based clustering algorithms in data mining: An experimental approach. En: *Information Technology Journal* 10 (2011), Nr. 3, p. 478–484. – ISSN 18125638

[36] Ying, Teh C. ; Doraisamy, S. ; Abdullah, L.N.: Genre and mood classification using lyric features. En: *Information Retrieval Knowledge Management (CAMP), 2012 International Conference on*, 2012, p. 260 –263

[37] Zhao, Ying ; Karypis, George: Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering. En: *Mach. Learn.* 55 (2004), Juni, Nr. 3, p. 311–331. – ISSN 0885–6125