

2011-07-21

Filtering Social Tags for Songs based on Lyrics using Clustering Methods

Rahul Chawla

University of Miami, rahulchawla273@gmail.com

Follow this and additional works at: https://scholarlyrepository.miami.edu/oa_theses

Recommended Citation

Chawla, Rahul, "Filtering Social Tags for Songs based on Lyrics using Clustering Methods" (2011). *Open Access Theses*. 272.
https://scholarlyrepository.miami.edu/oa_theses/272

This Open access is brought to you for free and open access by the Electronic Theses and Dissertations at Scholarly Repository. It has been accepted for inclusion in Open Access Theses by an authorized administrator of Scholarly Repository. For more information, please contact repository.library@miami.edu.

UNIVERSITY OF MIAMI

FILTERING SOCIAL TAGS FOR SONGS BASED ON LYRICS USING
CLUSTERING METHODS

By

Rahul Chawla

A THESIS

Submitted to the Faculty
of the University of Miami
in partial fulfillment of the requirements for
the degree of Master of Science

Coral Gables, Florida

August 2011

©2011
Rahul Chawla
All Rights Reserved

UNIVERSITY OF MIAMI

A thesis submitted in partial fulfillment of
the requirements for the degree of
Master of Science

FILTERING SOCIAL TAGS FOR SONGS BASED ON LYRICS USING
CLUSTERING METHODS

Rahul Chawla

Approved:

Mitsunori Ogihara, Ph.D.
Professor of Computer Science

Terri A. Scandura, Ph.D.
Dean of the Graduate School

Burton Rosenberg, Ph.D.
Associate Professor of Computer Science

Dimitris Papamichail, Ph.D.
Assistant Professor of Computer
Science

CHAWLA, RAHUL

Filtering Social Tags for Songs based on
Lyrics using Clustering Methods

(M.S., Computer Science)
(August 2011)

Abstract of a thesis at the University of Miami.

Thesis supervised by Professor Mitsunori Ogihara
No. of pages in text. (57)

In the field of Music Data Mining, Mood and Topic information has been considered as a high level metadata. The extraction of mood and topic information is difficult but is regarded as very valuable. The immense growth of Web 2.0 resulted in Social Tags being a direct interaction with users (humans) and their feedback through tags can help in classification and retrieval of music. One of the major shortcomings of the approaches that have been employed so far is the improper filtering of social tags. This thesis delves into the topic of information extraction from songs' tags and lyrics. The main focus is on removing all erroneous and unwanted tags with help of other features. The hierarchical clustering method is applied to create clusters of tags. The clusters are based on semantic information any given pair of tags share. The lyrics features are utilized by employing CLOPE clustering method to form lyrics clusters, and Naïve Bayes method to compute probability values that aid in classification process. The outputs from classification are finally used to estimate the accuracy of a tag belonging to the song. The results obtained from the experiments all point towards the success of the method proposed and can be utilized by other research projects in the similar field.

Acknowledgement

I would like to thank Dr. Mitsunori Ogihara, my advisor for giving me the opportunity to work under him. With his constant guidance and motivation, I am able to graduate with a master's degree in Computer Science. It was a great learning experience to be under his wing. I would also like to thank my other thesis committee members, Dr. Burton Rosenberg and Dr. Dimitris Papamichail for their valuable feedback.

I would like to thank the Department of Computer Science for providing me with funding and necessary resources for the Graduate studies.

I want to sincerely thank my colleague and lab partner Yajie Hu for helping me with his knowledge and experience in the field.

I want to thank all my friends, colleagues and faculty for their support and help.

Table of Contents

List of Figures	v
List of Tables	vi
Chapter 1	Introduction	1
Chapter 2	Rapid Miner	10
Chapter 3	Related Work	15
Chapter 4	Proposed Method.....	28
Chapter 5	Experiments & Results	46
Chapter 6	Conclusions	53
BIBLIOGRAPHY		54

List of Figures

1.1	Complete framework of the method proposed.....	7
2.1	The welcome screen for RapidMiner.....	12
2.2	Design Perspective in RapidMiner.....	13
3.1	Hevner’s adjective model.....	20
3.2	Schlosberg’s facial expression categories.....	20
3.3	Russell’s model with two dimensions.....	22
3.4	Thayer’s Model having four mood clusters.....	22
4.1	WordNet::Similarity Perl script for Vector Algorithm.....	32
4.2	Applying hierarchical clustering to create tag clusters.....	33
4.3	Cluster structure and optimum threshold setting.....	34
4.4	Preprocessing done in RapidMiner.....	38
4.5	Feature selection and CLOPE clustering in RapidMiner.....	42
5.1	Collection of lyrics from HTML parsing.....	46
5.2	Collection of tags from Last.fm.....	47
5.3	Tags collected are stored in a text file.....	47
5.4	WordNet::Similarity Perl script output	48
5.5	Output of the CLOPE clustering method on RapidMiner.....	50

List of Tables

5.1: The clusters of social tags.....	49
5.2: Distribution of tags among lyric clusters.....	51
5.2: Experimental results of tag identification	52

Chapter 1

Introduction

1.1 Background and Motivation

In the recent few decades, we have seen a drastic transformation in the field of Music data handling and storage. Instead of having multiple tapes and cassettes, it is now possible to store the complete collection of several artists on a single handheld device. With this digital revolution, both in hardware devices and online availability of music, the challenge that appeared was how to access and mine this data in a way that is both convenient and takes human logic into account. Data Mining is a field to process data using sophisticated data search capabilities and statistical algorithms to discover patterns and correlations in large databases. Music Data Mining is a branch of data mining that focuses on extracting valuable information from the music related data and utilizing that information to build systems to cater different user needs such as retrieval and classification of music.

Early research in the field of music information retrieval was aimed towards finding similarity among pieces of music using their acoustic features. The example of such works can be seen in (Logan & Solomon, 2001), (Cooper & Foote, 2002) etc. The initial work was done using classical instrumental music; the work was then extended to other types of music. Audio data mining has been effective since, but still lacks in using vocal audio and transcribing it into words. Hence, when it came to mood and topic related information the research kept going for additional features.

The metadata for a song could be the genre, artist name, release year, composer etc. To organize and retrieve songs more effectively, music psychology is involved. The focus of research turns to extraction of high level features, for using them as metadata. The task is challenging because of the fact that most high level features have an implicit presence in the music data. The examples of high level features of a song are emotion and mood expressed by the song and topic of the song. The psychological models such as (Hevner, 1936) and (Schlosberg, 1952) were used as reference to see the range of emotions and moods. Chapter 3 discusses more of these models along with low dimensional graphical representations of emotion. In the early phases of research, the high level information extraction results were not satisfactory because of the lack of proper resources and techniques. The lyrical meaning became too complex for the machine to read and make sense. The semantic distance made it difficult to have a definite research question and therefore the goal was unclear.

With the emergence of Web 2.0, which gained a lot of popularity, there was an immeasurable amount of user participation. On social platforms general users started to share their opinions, views and many other information in form of keywords, phrases and sometimes plain text. The knowledge is now easy to understand by the machine. The downside to the Web 2.0 revolution is the high amount of noise that came with it. The user submitted data related to song information is very useful but it comes with information that is either not required or is incorrect. The motivation for this thesis comes from the researchers who would like to use the online resources but want the resources to be clean and narrowed down to the required set.

In this thesis, we delve into the mood and topic extraction aspect of music data mining. The work done in this aspect of field has been fairly recent. For all types of music resources, three major features are used in the field of music information extraction:

1. Audio features
2. Lyrics features
3. Social tags

As discussed earlier, the audio features have always been used predominantly. The acoustic features have been applied for music style recognition, genre categorization, similarity and emotion detection (Li & Ogihara, 2004). The drawbacks of using audio features include lack of accuracy for large datasets and restricted application. Lyrics features were used to aid the audio features and later used independently to compare the results. The field of text mining has evolved over the years. Hence, lyrics are now considered a great source to extract music information. The majority of lyrics lack proper structure and grammar, and several words in the lyrics do not match with standard dictionaries.

With the dawn of new millennium, Web 2.0 allowed social tags to be an intricate feature in music data mining research. The drawback of Web 2.0 was the inevitable *noise*, causing the results to not reach the desired mark. As discussed in chapter 3, various attempts were made to use hybrid of two of the three features, but yet results were good for certain categories and poor for others.

The music data mining has two major goals, Music Information Retrieval and Music Classification. Mood and Topic have both been recognized as important metadata for the retrieval and classification aspect. Music Information Retrieval has become more demanding because datasets are larger and user queries are getting more and more specific. Similar for classification, as predefined categories are not consistent, we need to come up with dynamic classes that can help other applications and ultimately the user.

For both retrieval and classification if we want to use high level features, then user involvement is necessary. Since the resource is offered by humans, the high level features are more clear and understandable for users to retrieve music. In the thesis, we focus on filtering tags by removing not only the noise present in the tags but also narrowing down the collection to the tags that impart the mood and topic information. This is done by using clustering algorithms and with the help of lyrics features. There has been work done in this field but there is dearth of state-of-the-art work in the field.

The Web 2.0 website used in this thesis is Last.fm, which is a popular and contains a large number of tags and is regularly visited by a large number of users. The tags that a user submits are based on different perspectives, e.g. genre, album, artist name, mood, release year, topic etc. The noise such as misspelled tags, abbreviations, personal subjective opinions of the user towards the song, numerical tags are also inherent in the tags submitted by the user. The tags that contradict the mood and topic also exist, for instance **happy** and **sad** may occur in the same song.

Now even though both tags may have been validly labeled, by using additional resources we can estimate probability values of the tags that represent the song.

As evident by early experimentation, the task is challenging to build a method lyrics features were incorporated to bolster the algorithm. The lyrics will provide an extra reference for tags, so that if any noisy tag was not removed by standard procedure, the lyrics feature will catch it and make the result more accurate. Lyrics are chosen because of their close relation to the mood and topic of a song and it is an easily available resource. For our task we used www.azlyrics.com, a well-known website that provides a large amount of lyrics that can be accessed through artist/album/song name.

1.2 Framework

The study showed that a method needs to be proposed that can identify true social tags required for the classification. The framework is then prepared which will be explained in this subsection along with the help of a figure. After observing the tags in Last.fm, the first step was clear, i.e., identifying all the tags that have spelling mistakes, abbreviations etc. The users when submitting tags are not always careful and the mistakes in tags are inevitable. To recognize and fix the errors we needed a dictionary, for which we chose WordNet, a popular lexical database used by researchers in field of data mining, semantic web etc. The version 3.0 of WordNet consists of 155,287 words organized in 117,659 synsets and 206,941 word-sense pairs. WordNet was able to provide possible correct candidates for an erroneous tag. To select the correct one we used Metaphone, a phonetic algorithm that picked up the common mistakes made due to similar sounding words. For

instance words containing sounds made by *ck* and *k*, *ph* and *f* etc. are often confused. The inverse transformation of Metaphone algorithm can solve the issue of similarly pronounced words.

Now that the errors made by user regarding the spelling of the tags and using abbreviations are taken care of using WordNet and string transformations. We move on to the next step which is to recognize and remove tags that are not related to mood and topic at all. The tags like song's genre information, artist/album names, year released etc. would be handled. Also opinion tags that describe what an individual user may think about the song are needed to be recognized and subsequently removed. This part can be tricky and in chapter 4 we discuss the techniques we employed to remove tags as accurately as possible.

The next challenge of ascertaining whether the tags of a song really belong to the mood and topic of the song was handled as follows. The concept applied was that, knowledge of the general class of song's mood and topic information would help in ascertaining whether or not the tags belonged to that song. For instance, a general class could represent happiness, then tags like **cheerful**, **delight**, **joy** etc. tags would be automatically considered acceptable. The extra tags just add details to an already established general mood and topic. The aforementioned categorization is possible if we have general categories for possible moods and topics. Using the semantic knowledge provided by tags we automatically generate categories rather than predefined set of categories. The module used to evaluate the similarity of tags we used WordNet::Similarity. The cluster generation is done by employing hierarchical clustering method; the clusters are seen as general category.

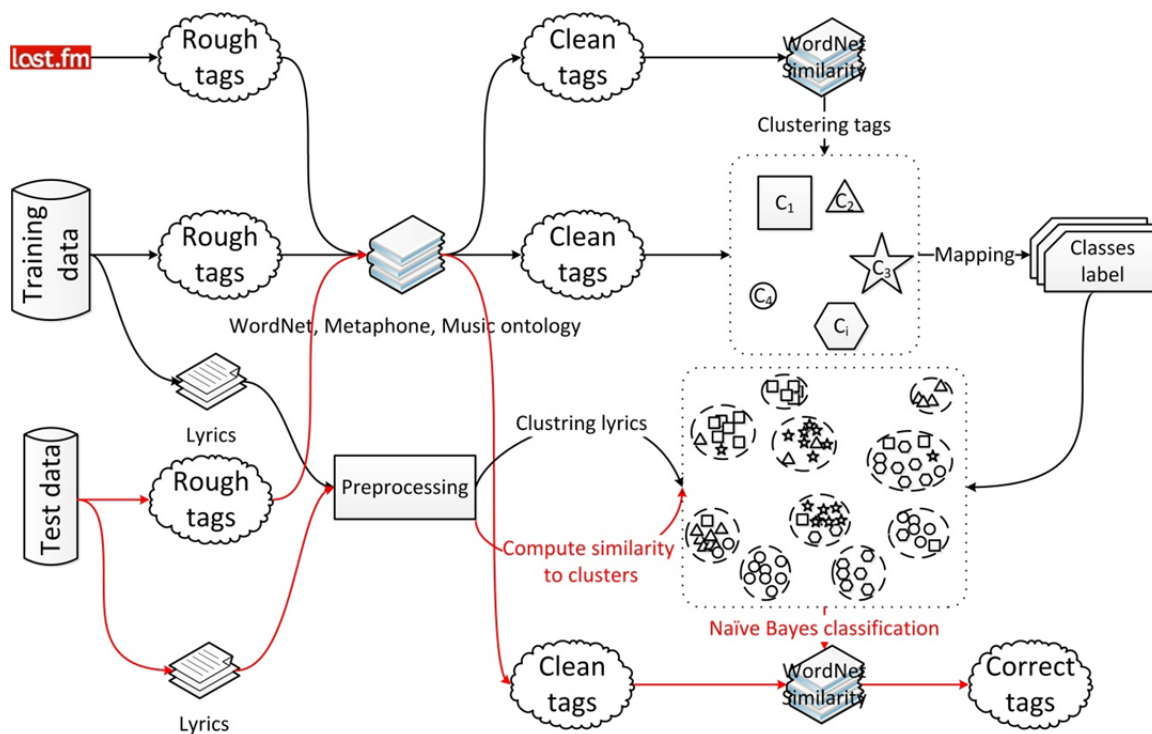


Figure 1.1: Complete framework of the method proposed.

The next part as visualized in the framework diagram is classifying the songs to the corresponding class. The lyrics features were utilized for solving the issue. The literature review suggested using CLOPE (Yang et al., 2002) clustering method and Probabilistic classifier (Lewis, 1998) would give better classification results. We were able to compute probability values like probability of a class given a song.

Also last part of our method was to obtain probability values of a class given the lyric and the similarity between the tag and other words in the class. This finally gave us a probability value of a tag being actually correct. The threshold value is set accordingly to get the best results. The framework shown in Figure 1.1 shows how the data transition takes place from training and test data through different steps mentioned above.

The next chapter will introduce the tools that were used to represent the data, apply and visualize the clustering algorithms and to perform results evaluation. Chapter 3 is a thorough literature review of the field. Chapter 4 discusses the proposed method along with the algorithm and probability equations to explain the framework discussed in Figure 1.1. Chapter 5 focuses on experiment and results data obtained by following the proposed method and using the data. Finally chapter 6 concludes the thesis with summarizing the goals achieved and possible applications for the method.

1.3 Contribution

This thesis proposes a methodology to improve the performance of Music Information Retrieval systems and Music Classification systems. The purpose is to introduce a novel approach for the systems that currently exist and systems that are under development to employ and achieve higher accuracy in the handling of online resources. The user-interaction through web applications in the form of social tags, comments, lists etc. provides high level information. This high level information can be considered as a vital resource for future work of Music Information Retrieval systems. The improper filtering of tags and lack of supporting features often leads to the poor performance of systems based on tags. The method proposed here analyzes, organizes and retrieves data with faster, scalable and effective techniques. The filtering process of tags developed in this thesis employs lyrics feature and various data mining techniques. The focus has been towards developing methods that can be used in an automated retrieval system thereby, avoiding any manual annotation.

The results suggest a high recall and reliable precision and hence, encourage other retrieval systems to employ the proposed method. To conclude this chapter we can confidently claim that, we have a robust, scalable and efficient method that has improved results than the contemporary methods employed for recognition and filtering of tags.

Chapter 2

RapidMiner

In the field of data mining, when you have a large amount of data, it is difficult to analyze data sources considering the increasing complexity of the data. The commercial approach to this issue is resorting to costly software that can provide and predict data behavior and hence enable the user to make rightfully measured decisions. For research purposes we have to find a software that is Open Source and highly effective in its performance. After some research, *RapidMiner* was found to be the best available tool that meets the requirements of this thesis.

In brief, RapidMiner was developed in 2001 under the direction of Prof. Dr. Katharina Morik at University of Dortmund. The project was known as YALE (Yet Another Learning Environment) in its early phases. The software has been progressively improving since, and now it consists of more than 500 operators. The list of operations that can be performed using RapidMiner includes data processing, text mining, web mining, time series analysis, opinion analysis etc. RapidMiner has several advantages over contemporary data mining software. RapidMiner not only contains all the functions and methods available in any contemporary software but also has a better user interface and better visualization of data and processes.

RapidMiner 5.0, the latest version of the series, is the one we used for the thesis. The important attributes that RapidMiner encompasses and due to which we selected the tool for our method are as follows:

- The tool has a great user interface, as it is developed using Java code. The tool allows flexibility in terms of process generation from operators as the representation is done by “operator trees” and internally it is represented by XML to check for any format adjustment requirement.
- Along with the outstanding modeling techniques, a lot of preprocessing work can be performed using RapidMiner. This was not possible in any other open source data mining software e.g. Weka. The preprocessing phase is a time consuming part of data mining work and RapidMiner enabled with us to save the time and hence get faster results.
- Another impressive feature that has been included in the recent version of the software is the optimization of memory usage. This attribute allows the software to perform large number of data analysis and transformations with faster speed and minimum load on the memory.
- Another important feature is the data connectivity; the sources nowadays range from SQL servers to Microsoft Access to Microsoft Excel and so on. RapidMiner shows capability of connecting to all different data sources and give effective results.

The Rapid-I community is online for all the updates, tutorials as well as for downloading extensions and plug-ins for the software. The following figures show some screenshots and we can see the different functionalities. In Figure 2.1, we can see the welcome screen with the menu bar. The Perspectives buttons give us the option to switch to design view, workplace view and back to welcome screen. The Actions include “New” to start a new process, “Open Recent” to provide a list of recently accessed processes,

“Open” to open the repository browser and “Open Template” to give us options of template analysis processes, “Online Tutorial” takes you directly to the tutorial page, it is highly recommended for beginners.

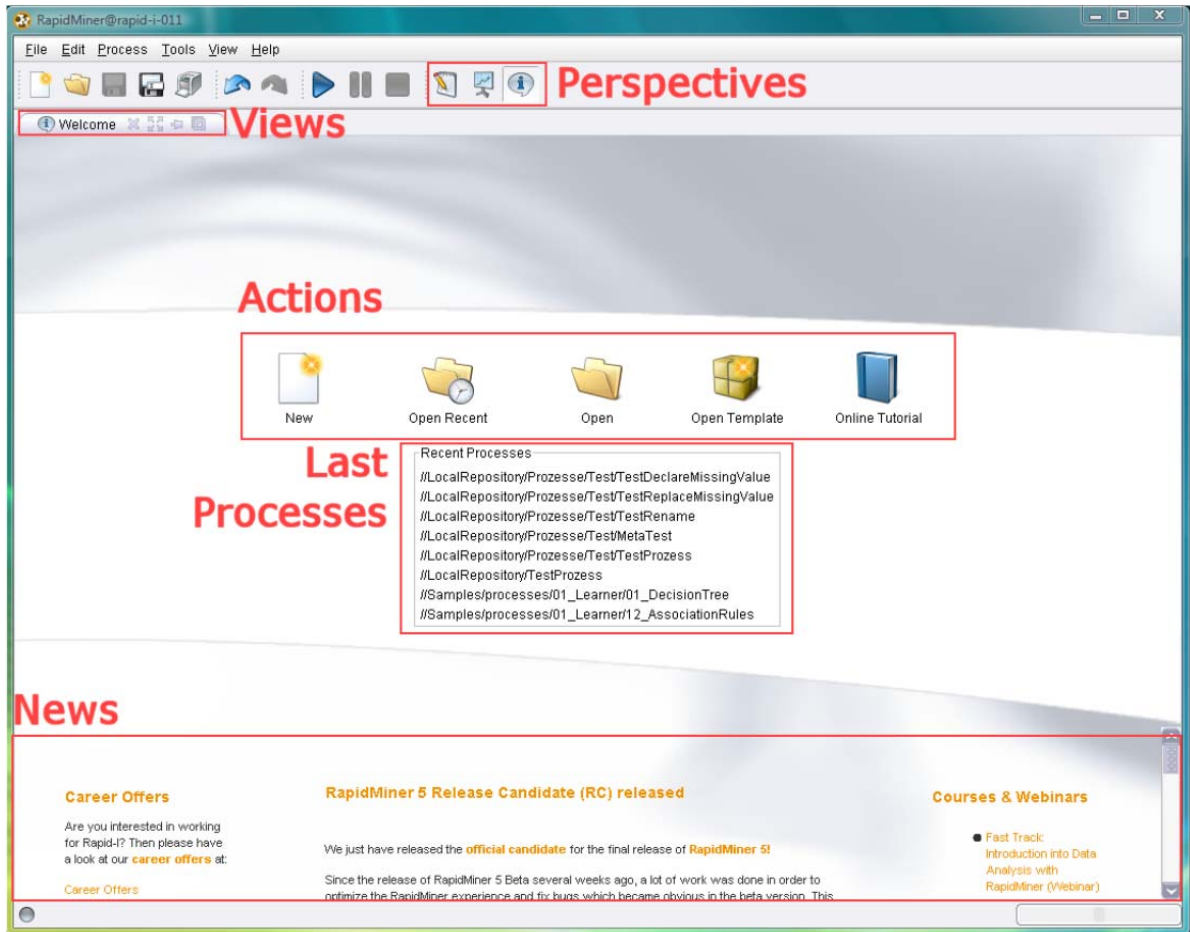


Fig. 2.1: The welcome screen for RapidMiner

The design perspective is shown below in Figure 2.2. On the left hand side the figure shows Operators and Repositories View; this section provides a large selection of operators that are efficiently classified and can be involved in the current process by double clicking. The operators can be searched by the giving keywords in the search box. In the center, the figure displays the Process View that shows the various processes and their interconnections. The Parameter view on the right hand side allows changing the parameters to gather different results and choose the best settings for the experiment.

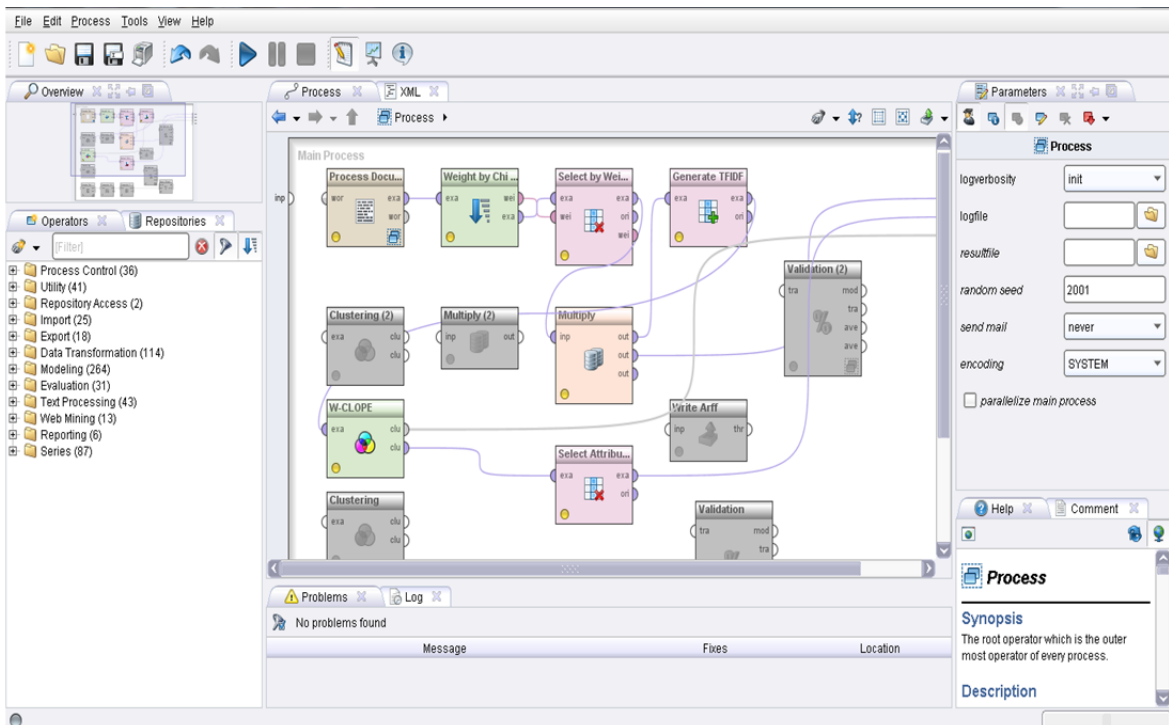


Fig. 2.2: Design Perspective in RapidMiner

The Figure 2.2 is an actual screenshot from the lyrics classification process that was performed using RapidMiner. The steps starting from preprocessing to clustering all can be seen in a single view. The results were generated by running the process. The syntax was constantly checked and any mistake was immediately flagged by the software.

Chapter 3

Related Work

3.1 Survey of Text Mining

Before we jump into high level music features and online social tagging, we will give some background in text mining as lyrics used in this thesis are set of text documents. Lyric Mining is a specific application of the *Text Mining* theory in which we extract information from lyrical text data. We will be looking into the techniques used for Text Mining and how they have developed over the period of time.

Text Mining methodology emerged in the mid of 1980s but it has recently advanced in the last ten years as a powerful field. This advancement can be credited to the exponential increase in the amount of documents that are available in digital format. Almost 80% of information is currently stored as text. The requirement to organize, systemize and categorize this data is what led to the strengthening of text mining. The field's progress has been noticeable in both the research and the industrial field (Sebastiani, 2002).

This survey will focus on how the research community is handling text mining. The primary approach has been Machine Learning techniques. The learning model has the basic underlying concept of “*a general inductive process using a set of pre-classified documents to build an automatic text classifier*”. We will focus on Text Categorization as it shares the traits of Machine Learning and Information Retrieval, thus giving the perfect

instance of Text Mining. Text Categorization has been defined by (Sebastiani, 2002) mathematically as:

Text categorization is the task of assigning a Boolean value to each pair $(d_j, c_i) \in D \times C$, where D is a domain of documents and $C = \{c_1, c_2, \dots, c_{|C|}\}$ is a set of predefined categories. A value T is assigned to (d_j, c_i) indicates a decision to file d_j under c_i , while a value F indicates a decision not to file d_j under c_i . The task is to approximate the unknown target function $\phi: D \times C \rightarrow \{T, F\}$ by means of a function $\Phi: D \times C \rightarrow \{T, F\}$ called the *classifier* such that ϕ and Φ coincide as much as possible.

We will now look into some of the classical and popular methods of building the inductive text classifiers. The concept of classification (selecting suitable category/categories for a document) depends on the relevance of that document. This relevance is expressed by the *category status value (CSV)* function. $CSV_k(d_j)$ amounts the relevance of the document d_j to the category c_k :

The classification can be categorized as hard (fully automated) and soft (semi-automated) based on the quality of training data available. If the training data needs some contextual information it is better to follow semi-automated classification. The CSV function takes up different roles for different classifiers as we will see in the discussion below.

In hard classification we can define the function CSV_k similar to the soft classification but followed by definition of threshold τ_k such that $CSV_k(d_j) \geq \tau_k$ is inferred as true and $CSV_k(d_j) \leq \tau_k$ is inferred as false. Let us see how CSV is viewed by probabilistic classifiers:

- Probabilistic Classifiers: They interpret $CSV_i(d_j)$ in terms of $P(c_i | \vec{d}_j)$, i.e. the probability of a document being represented by a vector $\vec{d}_j = \langle w_{1j}, \dots, w_{|T|j} \rangle$ of terms belonging to c_i , and this is calculated by application of *Bayes' Theorem* by

$$P(c_i | \vec{d}_j) = \frac{P(c_i)P(\vec{d}_j | c_i)}{P(\vec{d}_j)}$$

The problem that possible vectors for \vec{d}_j is a large number is solved by assuming no two coordinates in vector \vec{d}_j are statistically dependent. The classifiers implementing this assumption are called *Naïve Bayes* classifiers.

$$P(\vec{d}_j | c_i) = \prod_{k=1}^{|T|} P(w_k | c_i)$$

The most popular probabilistic approach to Text Categorization can be seen in papers such as (Li and Jain, 1998), (Liu and Lv, 2005), (Meena and Chandran, 2009) etc. The Binary Independence Model allows us to compute the probability of a document's presence in a certain class without the requirement of the probability of the document itself. If binary valued vector representation is used for documents, we observe the following in Text Categorization; the document space is divided into c_i and \bar{c}_i such that we get the following equations as shown below:

$$\log P(c_i | \vec{d}_j) = \log P(c_i) + \sum_{k=1}^{|T|} w_{kj} \log \frac{p_{ki}}{1 - p_{ki}} + \sum_{k=1}^{|T|} \log(1 - p_{ki}) - \log P(\vec{d}_j)$$

$$\log(1 - P(c_i | \vec{d}_j)) = \log(1 - P(c_i)) + \sum_{k=1}^{|T|} w_{kj} \log \frac{p_{ki}}{1 - p_{ki}} + \sum_{k=1}^{|T|} \log(1 - p_{ki}) - \log P(\vec{d}_j)$$

Here p_{ki} stands for $P(w_{kx} = 1 | c_i)$. The binary independence permits the classifier to search through infinite number of documents without the requirement of calculating the probability of each classification, (Lewis, 1998). Further research in probabilistic classifiers confronted the tasks of “lowering the independence assumption”, “for vectors not necessarily being binary valued”, “lead to document length normalization” etc. We have used Naïve Bayes approach in our method for lyrics classification, the probability values evaluation is shown in the next chapter.

Other commonly used text categorization approaches are Decision Tree Classifiers (DT), Regression Methods, Rocchio Methods, Examples based Classifiers, Support Vector Machines (SVM) etc.

3.2 Concept of Emotion and Mood

Before we survey Lyrics Mining in depth we need to understand the concept of emotion and mood. The *New Oxford American Dictionary* defines *emotion* as “a natural instinctive state of mind deriving from one’s circumstances, mood, or relationships with others,” “any of the particular feelings that characterizes such a state of mind” and it defines *mood* as “a temporary state of mind or feeling”. Based on these definitions we can draw the conclusion that emotion is instinctive, occurs with full awareness of mind and the object is often present, whereas mood is persistent, occurs with little awareness of mind and the object is never present.

For a long period of time, music information has been categorized using metadata like artist, genre, composer, date etc. Recently studies have shown that for highly effective and operational music information retrieval systems, the metadata discussed above is not sufficient. (Futrelle & Downie, 2002) specified the need for contextual information along with the other basic information. (Lee & Downie, 2004) suggested the need for not only context metadata but associative and relational metadata. The survey also used the “public information-seeking” behaviors for reviews, ratings etc.

3.3 Early models of mood and emotion words

In early stages of research, the work for music information appraisal focused more on the ontologies designed by using emotional adjectives and labels that are commonly used (for example passionate, sad, satisfying etc.). We will quickly discuss some of the categorical models like (Hevner, 1936), (Schlosberg, 1952) to present their ideas on human emotions and how the models visualized this information.

(Hevner, 1936) was the first to experiment with 450 listeners to 26 samples of classical music and for each sample choose from one of the 66 adjectives that felt appropriate. The 66 adjectives were then distributed into 8 groups in circular fashion (Figure 3.1). The emotions steadily change as we move across the circle. Hevner’s motive for the experiments was to equate the score samples and the adjective groups picked by the listener.



Figure 3.1: Hevner's adjective model

(Schlosberg, 1952) described the model in a circular style resembling the Hevner Model organization of the emotional words as seen in Figure 3.2. The model used Ekman emotional words (Ekman, 1982), namely anger, fear, happiness, disgust, surprise and sadness, with “sad” being changed to “contempt”.

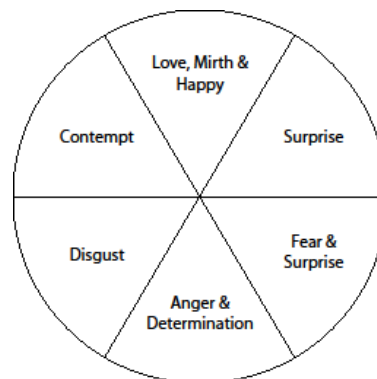


Figure 3.2: Schlosberg's facial expression categories

3.4 Higher Dimensional models for mood and emotion words

A highly popular model suggests that emotions can be identified in a continuous multidimensional space. The most effective dimensions would be *Valence-Arousal* and *Dominant-Submissive* as discussed in (Russell, 1980) and (Thayer, 1989). We will discuss both Russell and Thayer's model shortly. The dimensions can be more than two, for instance (Wedin, 1972) studied and recognized three dimensions, *Intensity vs. Softness, Pleasantness vs. Unpleasantness and Solemnity vs. Triviality*.

Russell's model categorizes 28 emotion words in a valence by arousal axes (see Figure 3.3). The x-axis is valence that spans between happy and unhappy and arousal that spans between activity and inactivity. Russell model has been followed by researchers like (Schubert, 1996), (Laurier et al., 2008) etc.

Thayer model is modeled after Russell model, it has a two dimensional emotion mapping with x-axis spanning through energy and tiredness whereas y-axis spanning through calmness and stress (see Figure 3.4).

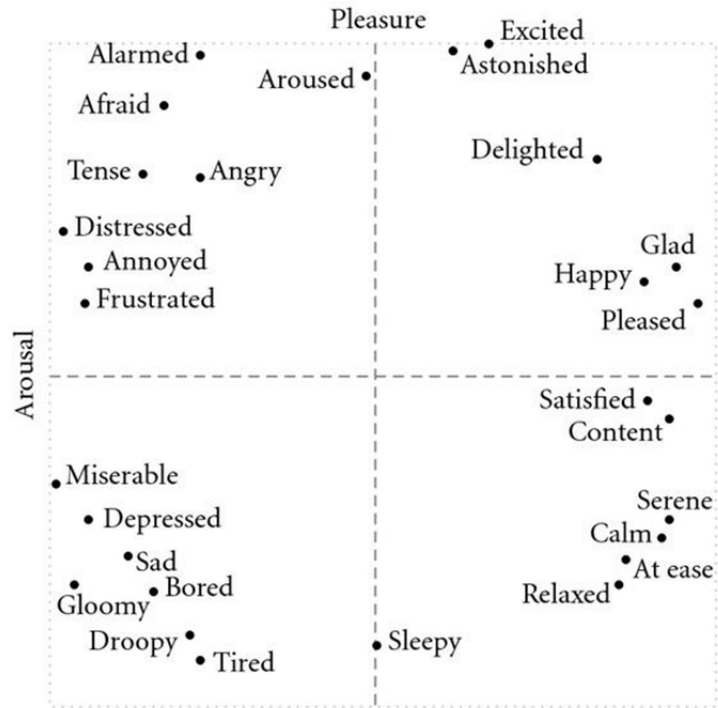


Figure 3.3: Russell's model with two dimensions

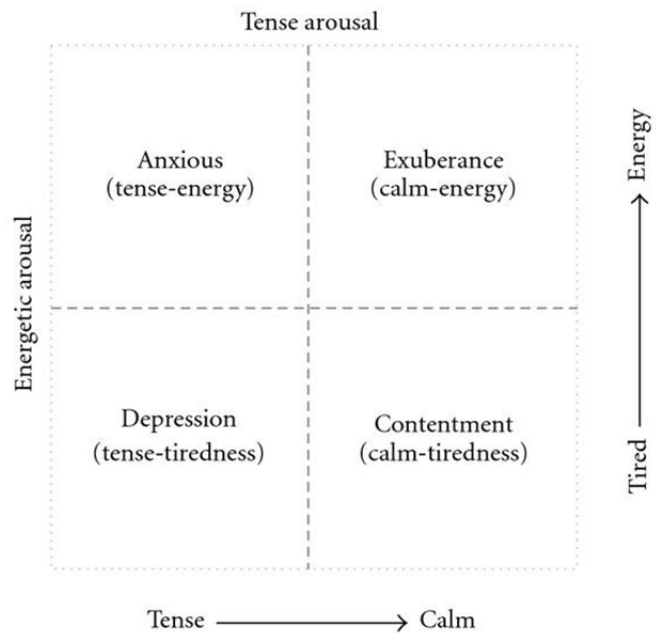


Figure 3.4: Thayer's Model having four mood clusters

3.5 Collection Emotion and Mood Labels

For researching large data sets of music it is not always possible to obtain a data set with labels. There are multiple ways to collect the labels which we will discuss in this section. The obvious one would be Manual Labeling by single or multiple humans. This approach has been used by (Li & Ogihara, 2003), (Yang & Lee, 2004) etc. These papers worked on audio pieces but the intention was similar to our goal which is to ultimately reach the emotion and mood categories. The results for these papers brought out some of the drawbacks with human labeling method. The most crucial one was the conflicting nature of the labels. The labels are subjective to different human perceptions of the music data.

The recently popular method for collecting the labels has been Social Tags. There is a huge online society which is continuously expanding and has members expressing their views on various songs and music pieces. Last.fm is an example of people providing labels which can be used to estimate emotion and mood. The drawback with this approach would be the *noise* in the tags. For instance, tags regarding the year of song release, title of the album will not impart any information towards mood and emotion classification. The Social Tagging approach has been made more interesting by involving games to collect the tags. Examples for that would be (Law et al., 2007) – *TagATune*, (Mandel & Ellis, 2007) – *MajorMiner* etc.

3.6 Examples of Music Information Retrieval Systems

3.6.1 Lyrics based

In the field of Music Data Mining, lyrics haven't been used to their full potential. In terms of extracting the emotion and mood information, lyrics are equally effective if not better than audio. We will discuss some of the major MIR works using lyrics as primary data set. (Logan et al., 2004) in which the authors semantically analyzed the lyrics to determine music similarity. The technique used was Probabilistic Latent Semantic Analysis (PLSA). The method converted each document to a characteristic vector and measured the distance in the vector space. The results obtained from the experiments were compared to acoustic counterparts and even though the result weren't better, a combination of both methods was suggested.

(Li and Ogihara, 2004) studied use of semi supervised learning from different information sources. They studied different lyric features other than the most commonly used bag of words. The paper discussed the possibility of using function words, POS statistics, orthographic features etc.

Another MIR work using lyrics was (Dhanaraj and Logan, 2005) where the purpose was to identify and predict hit songs. The analysis of lyrics was done by using Probabilistic Latent Semantic Analysis (PLSA) in which each lyric was converted to a vector that expressed the tendency of the song to be about a pre-learned topic. Their results concluded the lyric feature being more effective than acoustic features.

3.6.2 Tags Based

A fairly recent feature that has been introduced to the field of MIR is Social Tags. Music Listeners all around the world can provide the information about the music, lyrics, emotion, mood etc. of the song on a common platform.

Currently, the largest and commonly used website for obtaining tags is Last.fm. In the paper by (Levy & Sandler, 2007) tags from last.fm and mystrands.com were used. The paper showed the effectiveness of tags for purpose of music similarity and its future uses as music metadata. (Geleijnse et al., 2007) used Last.fm to collect tags for classifying, clustering and categorizing songs into genres. The experiments were done with 224 artists and 100 nearest neighbors. The work focused on contributing a ground truth for artist similarity and tags.

Thus, social tags are the way to the future, like mentioned in Section 3.5, various interactive games have been developed to get more users to contribute tags and enhance the MIR performances.

3.7 Examples of Music Mood Classification

3.7.1 Lyrics based

The use of lyrics as features independent of acoustic features is a fairly recent trend in music mood classification. (Besson et al., 1998) mentions, psychological science showed that lyrics had extra contextual information that can help to understand the real emotion and mood behind a song. (Juslin & Laukka, 2004) claims that 29% of the listeners have acknowledged the fact that lyrics play an important role in defining mood and emotion of

a song. The papers discussed in this section uphold the point that lyrics are a useful feature when we are trying to extract high level information.

We will discuss two papers, (Hu, Chen & Yang, 2009) and (He et al., 2008), to show the text oriented mining systems. (He et al., 2008) employed three different types of preprocessing methods. The bag of words features were analyzed in unigrams, bigrams and trigrams. Three feature representation models *tf-idf*, *absolute term frequency* and *Boolean* were used. The experiments included using Naïve Bayes, maximum entropy classification and support vector machines in which maximum entropy coupled with unigram, bigram and trigram gave best classification accuracy.

Now we will look into (Hu, Chen & Yang, 2009) and discuss mood clustering using lyrics. The language of the songs in the paper is Chinese but the method is effective for any language. The initial feature extraction didn't use bag of words but instead recognized a substantial set of Chinese affect words and mapped each work to a point in a 2-D real space. The initial level was generated by translating Affective Norms for English Words (ANEW) into Chinese. There were approximately 1000 words in the first level to which 2000 more synonyms were added hence the total collection spread out to 3000 words. Since they used seven labelers and kept only those songs in which at least six labelers agreed so the number of songs reduced from 1000 to 400. The experiment resulted with some high proximity to human perception which was measured using F-scores.

3.7.2 Tags Based

As mentioned in the introduction, the amount of work done in this field is rare and there is a lack of state-of-the-art work. The recent work that we studied for tags based work reference is (Laurier et al., 2009); the paper is entitled “Music Mood Representations from Social Tags”. For tags resource, Last.fm was used. The paper used Latent Semantic Analysis (LSA) to generate mood space and by the representation attempted to show the consistency between the community and experts. The results were encouraging and suggested strongly that tags can be used for Music Mood Classification.

Chapter 4

Proposed Method

In this chapter we will focus on the architecture and algorithm of the classification system. As discussed earlier, our method can be divided into 4 major parts. Firstly due to the inevitable noise present in the tags pool we have to clean them. The second part we will delve into generating clusters from the clean tags. This subsection gives a strong foundation to the framework of the project. After that we will present our algorithm to classify lyrics. This subsection will explore and utilize basic text mining techniques to extract as much semantic information as possible. Finally, the last subsection will do the ultimate tag filtering and probability computations will evaluate the accuracy.

4.1 Refining tags

The submission of tags is done regularly and by a large number of users on the internet. The most common issue with the submitted tags is spelling mistakes, slangs and use of abbreviations. These tags are in the same group as other useful tags. Before we can do any further processing of the tags we need to refine the erroneous tags right away.

We use a standard dictionary WordNet, (Fellbaum, 1998) for the refining work. WordNet is a lexical database developed in Princeton University by George A. Miller. WordNet is used commonly in research related to information sciences as it deals with both lexical and semantic relationship between word sets. We will use WordNet and various WordNet based algorithms in our work.

Another concept we need to know is “Levenshtein distance”. It is defined as the least number of transformations required to change one string to another. The allowed edits are insertion, deletion or substitution. We apply the Levenshtein distance in the WordNet for finding similarities between tag and a word. We choose the word with highest similarity to the tag as the correct word for that tag from the WordNet ontology. There are cases possible in which there will be multiple candidates having high similarity characteristics to the tag. We select only one of the candidates to replace the tag.

In the pursuit of refining incorrect tags, we use a phonetic algorithm “Metaphone”. Metaphone algorithm was published by Lawrence Philips in 1990 works by indexing words based on their pronunciation. In our collection of tags we observed errors in tags caused by similar pronunciation of the tag to the original correct word. Metaphone algorithm changed the characters in the wrong tag to other characters that made the tag correctly spelled according to the dictionary. An example of Metaphone algorithm in action would be transforming characters such as *ck* to *k* and *ph* to *f*. The correction in the proposed method is done by inverse transformation of Metaphone algorithm

The user submitted tags can range from the information about the song title, genre, year, artist name etc. to their subjective opinion of the song like awesome, great tune, love it etc. Our main goal is to extract tags pertaining to mood and topic of the song. Hence, any other tag not related to mood or topic information needs to be eliminated from the collection. The method to tackle elimination of each unnecessary type of tag is as follows:

The “year” tags that generally represents the release year of the song/album can be very easily deleted as these are the only numerical tags in the collection of words. To handle the case of tags related to genre and instruments we use lexicon described in (Wang et al., 2010). The paper by (Wang et al., 2010) had successfully mapped social tags to a predefined taxonomy hence allowing the capability of recognizing high level semantic information. Thus, by applying the ontology-based methods we successfully got rid of genre and instrument related tags.

The tags related to artist name are also easily recognizable as they are accessible from Last.fm. Once recognized the removal process is quite straightforward. Now we direct our focus towards a tricky set of tags, namely opinion tags. The opinion tags in the collection are usually subjective, i.e. they differ according to different users. The information imparted by opinion tags is not useful at all for music retrieval or classification because individual users may have conflicting views towards the song for instance, one user may love the song while the other might dislike it. We deal with these tags by collecting all the opinion words either manually for a small dataset or by using *WordNet-Affect*, an extension of WordNet. In *WordNet-Affect*, we have A-labels also known as emotion categories that are suitably organized to cover all classes of emotion. Specifically, since we are looking for affective meaning imparted by the word, *WordNet-Affect* allows us to identify between direct affective words and indirect affective words. The distinction based on semantic similarity can be applied to identify and remove opinion words.

4.2 Creating clusters of tags

In the first subsection we used WordNet and other resources to successfully clean our collection of tags and get mood and topic related tags. The next step would be to create clusters of the noise free tags. The method we employed generates the category automatically as opposed to some predefined category. If we analyze carefully, predefined categories does not have the same effectiveness in representation of songs. Since our method will see the majority of its usage in Music Information Retrieval systems, predefined categories' lacking in proximity with user queries will be an issue. With automatically generated categories, the classification results can be easily mapped to tags and because most of the user queries are based on tags we observe effective Music Information Retrieval performance. The idea behind this step is when we have a large collection of tags, if we create clusters of tags in which each cluster has tags related to each other through some semantic similarity then that cluster will most likely represent a specific concept.

Now to actually generate these clusters we used WordNet Similarity module (Pedersen et. al, 2004). WordNet::Similarity provides us to measure the semantic similarity and relatedness between concepts and words. The main lexical database is WordNet as discussed earlier. Since our goal is to look for semantic similarity and WordNet::Similarity has six measures of similarity (three of which are dependent on path lengths between concepts and other three are dependent on information content). We chose Vector algorithm to assess the similarity by computing the longest common path of any two words' parent paths. The length of common path can be defined as the similarity of the two words/concepts based on semantics. We modified one of the sample perl files

as shown in Figure 4.1 and implemented it on the tags to get the similarity values along with a proper error report for tags that couldn't find a match in the WordNet database.

```

print STDOUT "done.\n";

open(MYINPUTFILE, "<words.txt");
while(<MYINPUTFILE>)
{
# Good practice to store $_ value because
# subsequent operations may change it.
my($line) = $_;
# Good practice to always strip the trailing
# newline from the line.
chomp($line);
# Convert the line to upper case.
$line =~ tr/[A-Z]/[a-z]/;
my @words = split(' ', $line);
# Print the line to the screen and add a newline
#print "$line\n";
# Get the concepts.
my $wps1 = @words[0];
my $wps2 = @words[1];
unless (defined $wps1 and defined $wps2) {
print STDERR "Undefined input\n";
print STDERR "Usage: sample.pl synset1 synset2\n";
print STDERR "\tSynsets must be in word#pos#sense format (ex., dog#n#1)\n";
exit 1;
}
# Find the relatedness of the concepts using each of
# the measures.
my $value = $vector->getRelatedness($wps1, $wps2);
($error, $errString) = $vector->getError();
die $errString if($error > 1);
print "***** $line:$value\n";
}
__END__

```

Figure 4.1: WordNet::Similarity Perl script for Vector Algorithm.

After the similarity values of tags with one another are evaluated, we propose the scheme to visualize the results from the Vector algorithm. We obtained a similarity matrix with various tags and distances between them, the numerical values can be used to build a tree. Considering each tag as a node and the inverse of similarity value (distance) is viewed as an edge, we can successfully transform the matrix into a complete graph. Moving on to the actual creation of tag clusters we use Hierarchical Clustering method. The Figure 4.2 shows the step in coding where we use the Similarity Matrix for creating tags cluster.

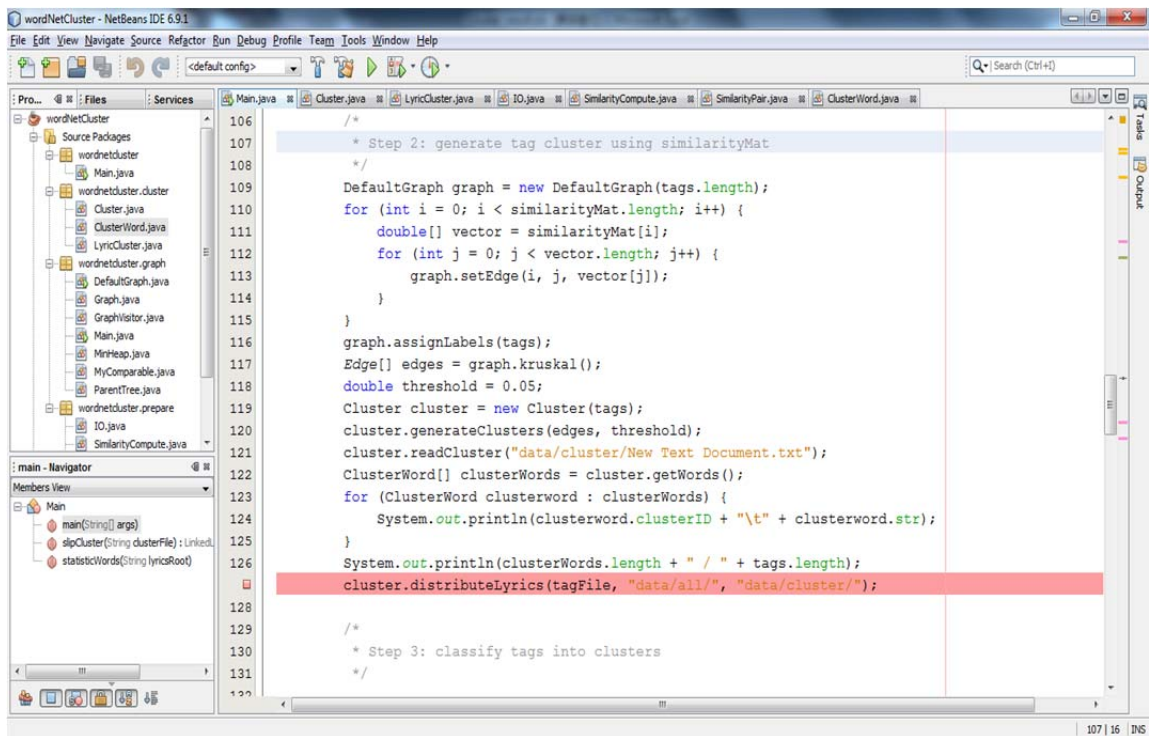


Figure 4.2: Applying hierarchical clustering to create tag clusters

Hierarchical Clustering is a very commonly used method of clustering in the field of data segmentation. The main goal in cluster analysis is to segment a collection of objects, in our case tags, into various clusters such that within each cluster the objects have high proximity and relation. The way Hierarchical Clustering works, the partitioning of data into a specific cluster is not done in a single step but a sequence of steps. The end result of which may be a particular cluster accommodating all objects or having equal number of clusters to the number of objects. Two types of Hierarchical clustering methods are *Agglomerative and Divisive*. In the agglomerative method each object starts in its own cluster and then pairwise merging moves the hierarchy upwards. In divisive method all objects start in a single cluster then repeated divisions performed

successively to obtain fewer clusters. The agglomerative method is used more commonly. Before applying the clustering we convert the complete graph into minimum spanning tree using Kruskal's algorithm.

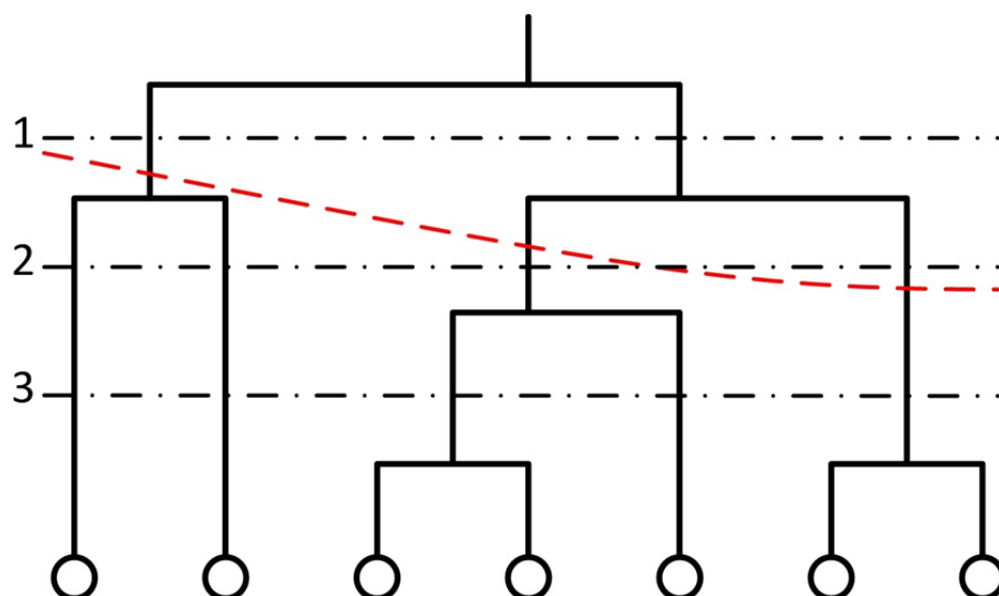


Figure 4.3: Cluster structure (black) and optimum threshold setting (red).

In classic hierarchical clustering method if we have a minimum spanning tree, any edge that has a higher value than the threshold is removed and the different subtrees remaining are treated as clusters. The Figure 4.3 shows how the cluster structure for our tags look like, the reason for it being WordNet has a tree like structure. This makes it imperative to choose the threshold value to remove edges. As evident in Figure 4.3, if the threshold is assigned a lower value as in dotted line 1, the clusters will end up being too general and no specific concept will be shared by the members of the cluster. The example of that could be two conflicting mood type tags appearing the same cluster which is not desirable for most retrieval systems. On the other hand, if the threshold is set to a higher value like dotted dash 3, the clusters will be too specific. The examples for

this would contain clusters with single data object again deceiving the purpose. When the threshold is assigned according to dotted line 2, we lose one cluster and 2 nodes. Therefore, the clusters that are produced must be handled and combined properly. The results need to be studied for each threshold value and as shown by the red line in the figure a minimum and maximum threshold needs to be set; hence we can get the desired clustering results.

Algorithm 1 below shows here how to cluster tags after a minimum spanning tree has been created from the graph. The algorithm below takes as input the minimum and maximum threshold values that are decided by observing various clustering results. The process starts by initializing a cluster list to retain the various output clusters. The algorithm also starts of by initializing tags set to store the tags. The first *for* loop goes from maximum threshold to minimum threshold and collects all the cluster results by cutting off edges. The clusters that contain only one member are discarded. All the iterations generate a new clusters list. Now to handle clusters, we use the nested loops shown below. The first loop iterates through the m clusters list. In second loop, we go through each cluster in the cluster list. The third and the innermost loop iterates through the tags in the cluster. Each tag is checked if it belongs in tags set S , if not the tag is added into the cluster. In the end we merge the clusters containing single member into its neighboring cluster. We will also be using cluster ID as the label.

Algorithm 1: Generating and Combining hierarchical clusters (based on different thresholds)

Input: Minimum spanning tree T , minimum threshold θ_{\min} , maximum threshold θ_{\max}

Output: Clusters $C = \{c_1, c_2, \dots, c_n\}$

Process:

Initialize clusters list $C = \{C_1, C_2, \dots, C_m\}$ to record clustering result with different threshold;

Initialize tags set S

for $thres = \theta_{\max}$ **to** θ_{\min}

 Cut off edges whose weight is less than $thres$ in T ;

 Collect members of every cluster;

 Remove clusters that contain only one member;

 Generate clusters C_i ;

end

for $i = 1$ **to** m **do**

for $j = 1$ **to** size of C_i **do**

 New cluster c ;

for $k = 1$ **to** size of c_j **do**

if $tag_k \notin S$ **then**

 Add tag_k into c ;

 Add tag_k into S ;

end

end

 Add c into C ;

end

end

Merge the cluster c_p that has only one member into the cluster c_q that contains sibling nodes of c_p ;

return C

4.3 Lyrics mining for the classification problem

From the section 4.2 we can get automatically generated categories from the minimum spanning tree after the algorithm was applied. Now the problem statement changes to a

classification problem. The reason being we collected a large number of tags, successfully created clusters in which the tags share certain semantic properties, now we must classify a song to the correct category it belongs to with the use of lyrics as a feature. In Music Mood Classification and Music Information Retrieval, lyric is an important feature that in many occasions have outperformed acoustic features. The availability of this resource is very convenient. There are many leading websites that contain lyrics in text forms and you can download them manually by parsing the html files. It is easy to implement and updating the lyrics content is relatively easy too.

Our method looks for tags related to mood and topic, the categories were generated based on mood and topic, thus our resource for classification purpose must express this music information too. Thus, we choose lyric feature that have been concluded to suggest the mood and topic information in (Hu, Downie & Ehmann, 2009).

There are plenty of text documents that are always mined for information, for instance blogs, news articles, emails, medical documents, reviews etc. Lyrics are in the same group as text documents that are mined but they have certain differing properties. The lyrics have shorter lengths and lack any grammatical structure (Hu, Chen & Yang, 2009). Hence, it is difficult to extract information from lyrics. The attribute of having short length has a negative effect because in order to properly take care of all the different words that occur in the lyrics we prefer to have a long vector with the number of features up to a few thousand. However, the actual lyric length is small and for a vector in the lyric we get hardly a hundred or so cells with non-zero value. This brings about the *data sparseness problem*.

In our method lyrics are represented by a vector consisting of real values. Similar to data mining, lyrics' mining also requires preprocessing of the lyrics documents prior to analysis of the data for representation. The main part in preprocessing is to clean the data, which for lyrics consists of word correction, notation removing and stemming. The preprocessing steps are done in RapidMiner as shown in Figure 4.4:

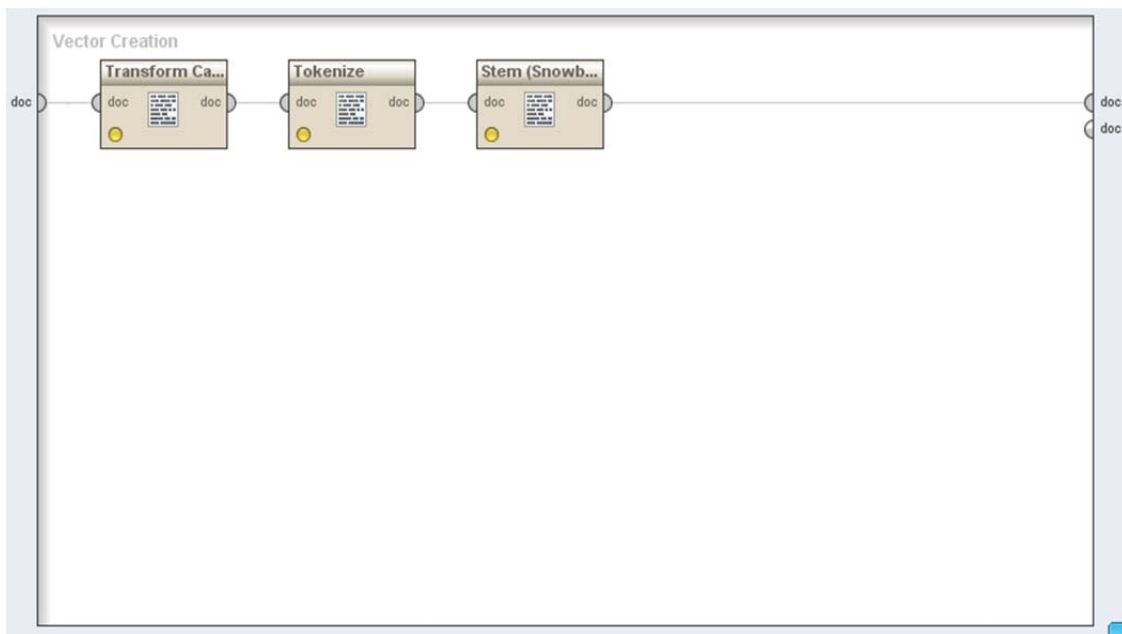


Figure 4.4: Preprocessing done in RapidMiner

The first step transforms documents into words; the tokenization is done by the second operator which carries the task of dividing any sentence into pieces, called *tokens*, and at the same time getting rid of certain characters, such as punctuation. The third operator is for the purpose of stemming. Stemming is the reduction of a word to its basic root. For instance, swimming, swimmer will be reduced to swim.

For text representation generally documents can be represented as feature vectors. The weight given to each feature may vary. The most commonly used TF-IDF (term

frequency-inverse document frequency) scheme is used for text representation. Since we already have the extracted lyrics dataset and corresponding vectors, TF-IDF is used to set the values of the vectors. The equations are shown below:

$$\text{tf}_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

$$\text{idf}_i = \log \frac{|D|}{1 + |\{j : t_i \in d_j\}|}$$

$$(\text{tf-idf})_{i,j} = \text{tf}_{i,j} \times \text{idf}_i$$

where, in a specific document d_j and $n_{i,j}$ is the number of occurrences of the term t_i in document d_j , $(\text{tf-idf})_{i,j}$ indicates the measure of importance of the term in the document. The symbol $|D|$ represents the total number of documents in the collection and $|\{j : t_i \in d_j\}|$ is the number of documents where the term t_i appears in a specific document.

As discussed previously in this section, data sparseness problem has been known to affect the method due to the presence of the redundant purposeless features. The problem needs to be dealt by trying to remove as many unnecessary features and thus strengthening the classification process. *Feature Selection* is the technique to successfully reduce the features and select meaningful and purposeful attributes. Since TF-IDF method gives us a measure of how important the term is in document, we get plenty of values for terms and we don't need all of them for classification purpose. The feature reduction method we decided to implement is Chi-square, χ^2 static. For any term present

in the lyric the importance regarding the classification is weighted by χ^2 static. The equations below represent the capability of term t to identify a class c .

$$\chi^2(t, c) = \frac{(A + B + C + D)(AD - CB)^2}{(A + C)(B + D)(A + B)(C + D)}$$

$$A = |D : d_i \in c, t \in d_i|$$

$$B = |D : d_i \notin c, t \in d_i|$$

$$C = |D : d_i \in c, t \notin d_i|$$

$$D = |D : d_i \notin c, t \notin d_i|$$

We find the χ^2 static for term t , i.e., $\chi^2(t)$ by computing the maximum value of static value among all the classes, the equation is shown below:

$$\chi^2(t) = \max_i \chi^2(t, c_i)$$

We keep as feature, terms which has $\chi^2(t)$ value over the threshold value.

Feature selection generally enhances speeds up and improves the classification process but what we discovered was loss of a portion of semantic information along with the valid value in the vector being reduced. The experiments were performed for distinct magnitudes of portion of features used to represent the lyrics. The results indicated that making the feature selection stringent by assigning a higher threshold value the classification performance declined whereas when a larger portion of features are used the performance shows some progress.

Hence, we introduce extension of semantic information. The concept is very simple; a lyric when extends the semantic information of another lyric, the valid value in the vector increases and the lyrics information is expressed suitably. This can be explained clearly by an example, for instance, a particular lyric has words like **cry**, **heartbreak**, **gloomy** but does not have the word **sad** and another lyric contains the word **sad** and resembles the previous lyric in terms of semantic information. We can extend the first lyric and add **sad** into it. With experimental results we were confirmed that by adding lyrics extension the classification performance improved as the probability values for identifying the correct class increased.

After feature selection is applied we observe that lyrics with similarities form a cluster and the centroid of the cluster is a mixture of the content of the lyrics present in the cluster. The centroid is just another instance object from the cluster but it shares contents from other members of the cluster. To cluster the lyrics we will be applying CLOPE, (Yang et. al, 2002). The Figure 4.5 below shows the various operators in RapidMiner that does preprocessing, feature selection, apply Chi square method and also the CLOPE clustering method. The output is used in evaluating probability values.

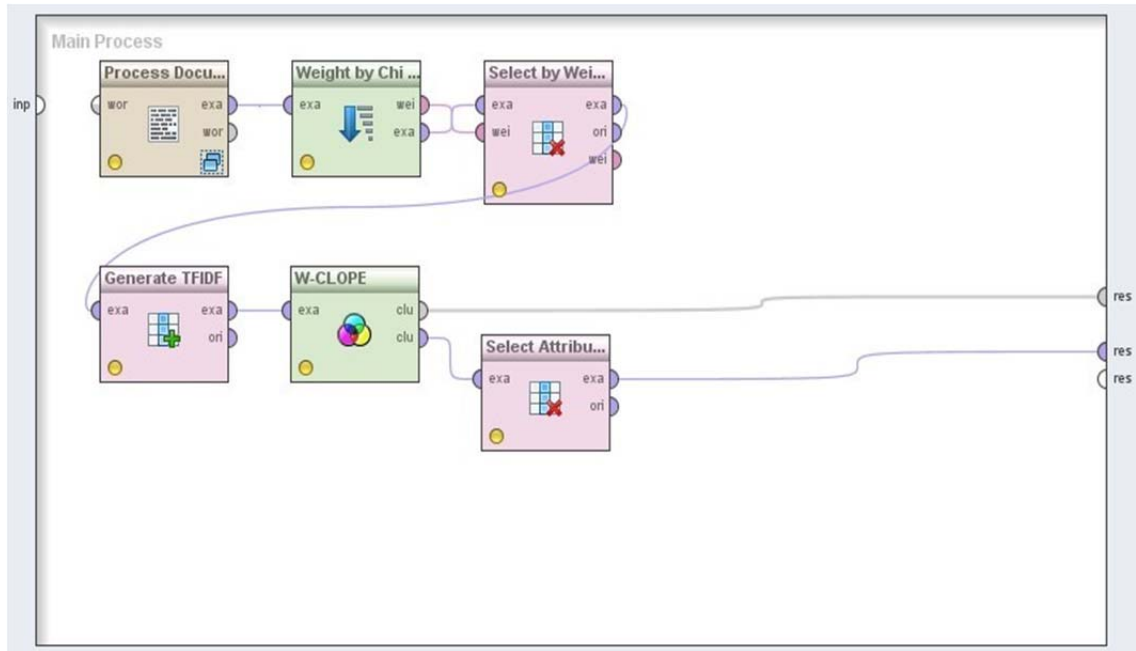


Figure 4.5: Feature selection and CLOPE clustering in RapidMiner.

CLOPE is a clustering algorithm which we will discuss in detail. The reason for choosing this particular clustering algorithm and how to utilize the result of the algorithm involved are shown below. The most distinguishing feature of CLOPE is that we don't need to assign the number of clusters as opposed to say K-Means clustering algorithm. It is successfully capable of representing the lyrics in feature space. In (Yang et. al, 2002) we see that CLOPE is used as an algorithm for transactional data. The work also shows that CLOPE works effectively for transactional data, if we can show parallels between transactional data and lyrics data we can justify the choice made for clustering algorithm. Firstly, transactional data has large volume and high dimensionality so does the lyrics data. Secondly, transactional data is numerically represented with vectors containing all the products and cells representing the amount of an item purchased. This representation causes vector to be quite long but the cells containing non-zero values are limited.

Similarly for lyrics data we discussed how due the short length of the lyrics we end up having fewer non zero cells in the large vectors. Thus, we can make a comfortable analogy of the lyrics being similar to the transaction and the features being similar to the products involved in the transaction.

CLOPE as discussed in (Yang et. al, 2002) employs a global criterion function. The function increases the intra-cluster overlapping of the transaction items by increasing the height to width ratio of cluster histogram. We can safely conclude that CLOPE is the correct choice for the data we have and the goals we want to achieve.

Subsequently, the CLOPE clustering process will provide us with various lyrics clusters and within each cluster the lyrics may or may not belong to the same class. We still observe one majority class occurring maximum number of times. Now that we have taken all the suitable measures to improve the classification performance let us take a look at the steps involved.

Firstly, to evaluate the probability that a class ω_i given a cluster Θ_j is:

$$p(\omega_i | \Theta_j) = \frac{|\{x : x \in \omega_i, x \in \Theta_j\}|}{|\Theta_j|}$$

Now, for a test lyric x and the centroid of a cluster θ_j , we can compute the similarity between the two by measuring the cosine distance. The probability value of the cluster Θ_j given test lyric is estimated by the cosine distance:

$$p(\Theta_j | x) \sim \frac{\theta_j \cdot x}{|\theta_j| |x|}$$

To find the probability of a specific class ω_i given an instance lyric x we have to use Probabilistic classifier theory also known as Naïve Bayes method.

$$\begin{aligned}
 p(\omega_i | x) &= \sum_j p(\omega_i, \Theta_j | x) \\
 &= \sum_j p(\omega_i | \Theta_j, x) p(\Theta_j | x) \\
 &\sim \sum_j p(\omega_i | \Theta_j) p(\Theta_j | x)
 \end{aligned}$$

The probability value obtained by the above equations are then utilized for achieving the main task of the project, i.e., to establish the accuracy of tags that are labeled for the lyric. This is explained in the next section.

4.4 Recognizing the true tags

The last subsection of this chapter will discuss the last step in our proposed method. To summarize, we have all the lyrics categorized automatically and each class contains its own set of tags from the cluster of tags. The similarity between the tag belonging to a particular song and all the tags present in the cluster are evaluated. The equation used for this purpose is shown next:

$$p(\text{correct} | \text{tag} \in x) = \max_i \left[\max_j \text{sim}(\text{tag}, t_j \in \omega_i) \cdot p(\omega_i | x) \right]$$

On the left hand side we have $p(\text{correct} | t)$ that represents the probability value of the tag being tested is the correct one. An optimum threshold value is selected, when it

exceeds the probability value we can consider the tag to be actually useful and imparting the information related to topic and mood.

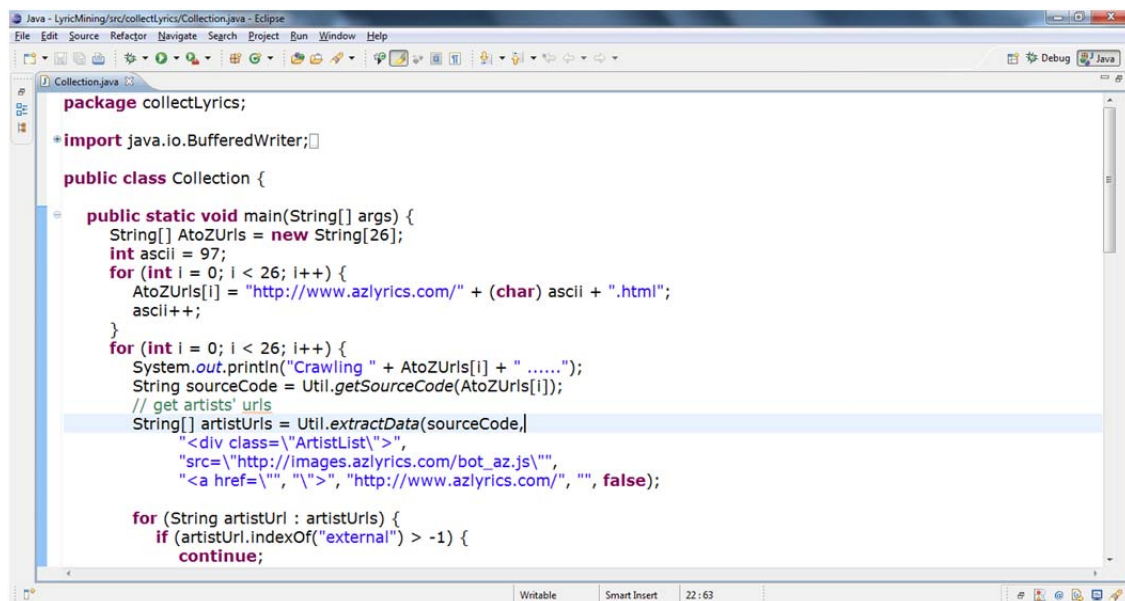
Chapter 5

Experiments and Results

5.1 Building the dataset

For the method discussed in chapter 4, we now look into the experiments performed using actual training and test data sets. The resources used for experimenting are available online for use. The algorithms and equations discussed in last chapter are implemented here.

Firstly, we downloaded lyrics online from the website www.azlyrics.com, a popular website with a huge collection of more than 100,000 lyrics. The lyrics were parsed from the html files by using Java code. The screenshot below shows the lyrics being downloaded, all lyrics were in English language and lyrics with multi-lingual text had to be removed.

The image shows a screenshot of the Eclipse IDE. The main editor window displays the source code for a Java class named 'Collection'. The code is as follows:

```
package collectLyrics;

import java.io.BufferedWriter;

public class Collection {

    public static void main(String[] args) {
        String[] AtoZUrls = new String[26];
        int ascii = 97;
        for (int i = 0; i < 26; i++) {
            AtoZUrls[i] = "http://www.azlyrics.com/" + (char) ascii + ".html";
            ascii++;
        }
        for (int i = 0; i < 26; i++) {
            System.out.println("Crawling " + AtoZUrls[i] + " .....");
            String sourceCode = Util.getSourceCode(AtoZUrls[i]);
            // get artists' urls
            String[] artistUrls = Util.extractData(sourceCode,
                "<div class=\"ArtistList\">",
                "src=\"http://images.azlyrics.com/bot_az.js\"",
                "<a href=\"", "\">", "http://www.azlyrics.com/", "\"", false);

            for (String artistUrl : artistUrls) {
                if (artistUrl.indexOf("external") > -1) {
                    continue;
                }
            }
        }
    }
}
```

The IDE interface includes a menu bar (File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help), a toolbar with various icons, and a status bar at the bottom showing 'Writable', 'Smart Insert', and '22:63'.

Figure 5.1: Collection of lyrics from HTML parsing.

For tags we used Last.fm, an online resource for tags on album, artist, songs, mood, topic etc. The tags are extracted by generating the URL for the song. As shown in the Figure 5.2, using the artist and song title that we know from the lyrics file, we can parse the html file to get all the tags available for the song.

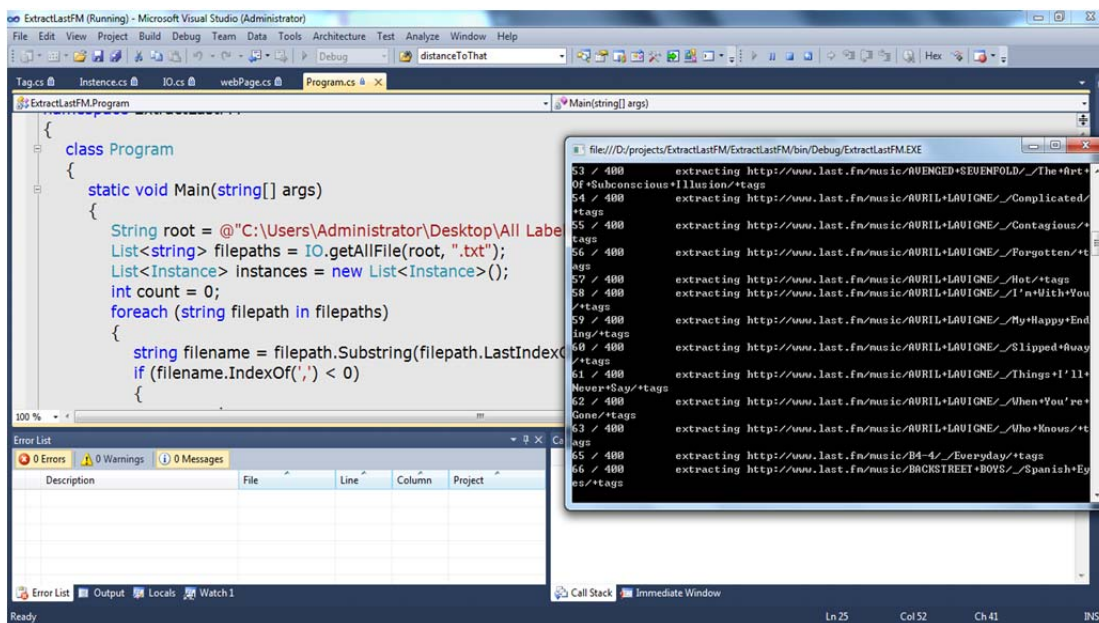


Figure 5.2: Collection of tags from Last.fm

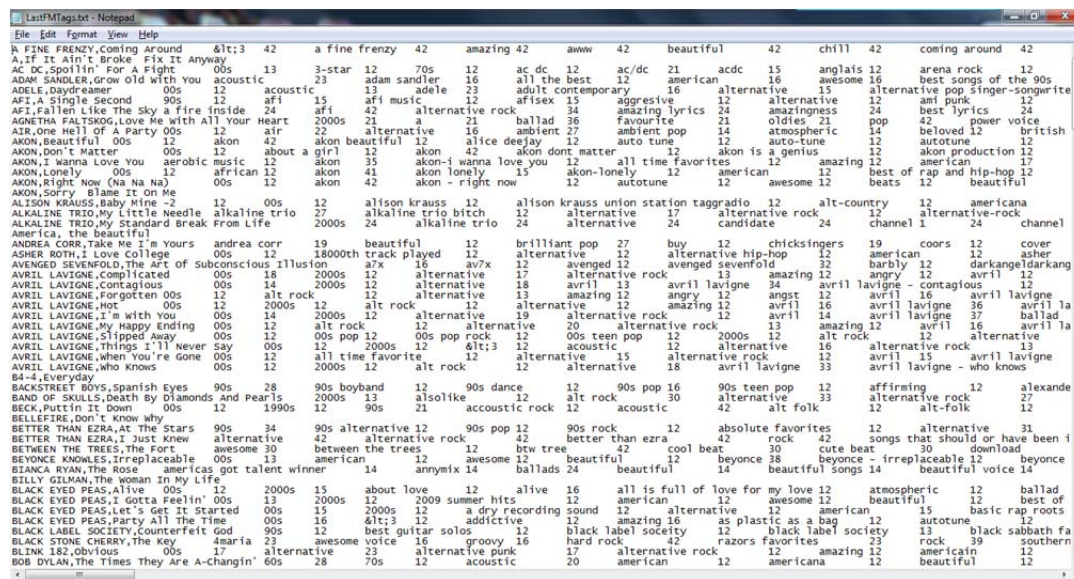


Figure 5.3: Tags collected are stored in a text file

The tags are stored in a text file as shown in Figure 5.3. The tags are represented according to song file names. For further analysis, the data can always be transferred to easily readable platforms like Microsoft Excel.

5.2 Tags clustering results

We compiled 309 songs for our experiment. The focus was for the dataset to consist of different genre and artists hence the mood and topic can range onto number of categories. Out of the 309 songs, 125 songs consisted of conflicting tags. The tags were representing information exactly opposite to each other, for instance happy and sad, and our experiments must succeed in correcting songs with opposing tags.

As shown above the tags that were collected were cleaned for any spelling mistakes, abbreviations, opinion related tags etc. There were 262 tags remaining after the refining process.

```
File Edit View Search Terminal Help
Argument "and" isn't numeric in numeric lt (<) at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 534, <GEN11> line 2130.
Use of uninitialized value in unpack at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 777, <GEN8> line 12862.
Use of uninitialized value $offset in seek at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 778, <GEN8> line 12862.
Use of uninitialized value $offset in seek at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 582, <GEN11> line 2131.
Illegal hexadecimal digit 's' ignored at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 586, <GEN11> line 2132.
***** gothic#a#1 smile#r#1:0
***** gothic#a#1 smile#r#1:0.0383016288068665
***** gothic#a#1 smile#r#1:0
***** gothic#a#1 smile#r#1:0.031454578372193
***** gothic#a#1 smile#r#1:0
***** gothic#v#1 smile#r#1:0.0206495820483954
***** gothic#v#1 smile#r#1:0
***** gothic#v#1 smile#r#1:0.0173633308514695
***** gothic#v#1 smile#r#1:0
***** gothic#r#1 smile#r#1:0.0206495820483954
***** gothic#r#1 smile#r#1:0
***** gothic#r#1 smile#r#1:0.0173633308514695
***** gothic#r#1 smile#r#1:0
***** gothic#n#1 glad#a#1:0.0380152717920509
***** gothic#n#1 glad#a#1:0.0343090118415819
Use of uninitialized value in unpack at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 840, <GEN8> line 12910.
Use of uninitialized value $offset in seek at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 842, <GEN8> line 12910.
Illegal hexadecimal digit 's' ignored at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 555, <GEN9> line 2735.
Argument "and" isn't numeric in numeric lt (<) at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 562, <GEN9> line 2735.
Use of uninitialized value in unpack at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 777, <GEN8> line 12911.
Use of uninitialized value $offset in seek at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 778, <GEN8> line 12911.
Illegal hexadecimal digit 's' ignored at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 528, <GEN9> line 2736.
Argument "and" isn't numeric in numeric lt (<) at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 534, <GEN9> line 2736.
Use of uninitialized value in unpack at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 777, <GEN8> line 12912.
Use of uninitialized value $offset in seek at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 778, <GEN8> line 12912.
Use of uninitialized value in unpack at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 777, <GEN8> line 12913.
Use of uninitialized value $offset in seek at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 778, <GEN8> line 12913.
Use of uninitialized value in unpack at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 777, <GEN8> line 12914.
Use of uninitialized value $offset in seek at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 778, <GEN8> line 12914.
Use of uninitialized value in unpack at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 777, <GEN8> line 12915.
Use of uninitialized value $offset in seek at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 778, <GEN8> line 12915.
Illegal hexadecimal digit 's' ignored at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 528, <GEN9> line 2740.
Argument "and" isn't numeric in numeric lt (<) at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 534, <GEN9> line 2740.
Use of uninitialized value in unpack at /usr/local/share/perl/5.10.1/WordNet/QueryData.pm line 777, <GEN8> line 12918.
```

Figure 5.4: WordNet::Similarity Perl script output

We generate results for every part of speech (e.g. verb, noun, adjective etc.) and we get different values as seen in Figure 5.4 and we keep the highest one. The similarity matrix generated is converted to a complete graph and undergoes Hierarchical Clustering. The purpose of clustering method is to generate a minimum spanning tree. Based on concepts discussed in section 4.2, we perform several experiments on the tree to assign the minimum threshold value as 0.03 and maximum threshold value as 0.08. Algorithm 1 as discussed in last chapter will give us the result, i.e., 19 clusters containing 112 tags. In table 5.1, we can see a portion of result.

Cluster	Social tags
1	painful wrenching
2	rage pissed poignant wistful sadness anger ...
3	depressing depressingly
4	funny stories
5	drama dramatic melodramatic
6	glad peace enjoyable joyful happiness happy
7	Jesus Christ
8	relaxed relax comfort
9	aggressive energetic
10	relaxing passionate haunting
11	lover intimate
12	venomous hurt
13	praise enchanting sensual survive fun ...
14	bouncy positive cheer cheerful upbeat
15	motivate driven
16	romantic romance
17	bitter miserable nostalgic crying sentimental ...
18	hopeful inspiring inspirational inspiration
19	courage encouraging faith

Table 5.1: The clusters of social tags

5.3 Lyrics classification and correctness evaluation results

The penultimate step in our method is to cluster the lyrics using CLOPE clustering method. We use TF-IDF (term frequency-inverse document frequency) to describe text feature. The output from the CLOPE clustering method gives us the information about which and how many tag clusters does the lyric belong too. Figure 5.5 shows the output in data view. The label column shows which class the lyric belongs to. The last column shows the cluster ID of the lyric. The lyrics that are clustered together also represent one majority class. The output shows we have 13 clusters of lyrics.

Row	label	metadata_file	metadata_path	id	cluster
1	17	A FINE FRENZY,Coming Around.bt	G:\wordNetClusterIda1	M 1	cluster0
2	17	,Who Knows.bt	G:\wordNetClusterIda1	M 2	cluster0
3	17	BOB DYLAN,The Times They Are A-Changin'.bt	G:\wordNetClusterIda1	M 3	cluster0
4	17	BON JOVI,It's My Life.bt	G:\wordNetClusterIda1	M 4	cluster0
5	17	BRITNEY SPEARS,Stronger.bt	G:\wordNetClusterIda1	M 5	cluster0
6	17	,Here I Am.bt	G:\wordNetClusterIda1	M 6	cluster0
7	17	CECE WINANS,He's Concerned.bt	G:\wordNetClusterIda1	M 7	cluster0
8	17	,8 Mile.bt	G:\wordNetClusterIda1	M 8	cluster1
9	17	,Lose Yourself.bt	G:\wordNetClusterIda1	M 9	cluster2
10	17	FORT MINOR,Remember The Name.bt	G:\wordNetClusterIda1	M 10	cluster1
11	17	GREEN DAY,Good Rid (Time Of Your Life).bt	G:\wordNetClusterIda1	M 11	cluster0
12	17	,The Climb.bt	G:\wordNetClusterIda1	M 12	cluster0
13	17	TAMMIN SURSOK,Whatever Will Be.bt	G:\wordNetClusterIda1	M 13	cluster10
14	18	ADELE,Day.bt	G:\wordNetClusterIda1	M 14	cluster0
15	18	AIR,One Hell Of A Party.bt	G:\wordNetClusterIda1	M 15	cluster0
16	18	AKON,Lonely.bt	G:\wordNetClusterIda1	M 16	cluster1
17	18	AKON,Right Now (Na Na Na).bt	G:\wordNetClusterIda1	M 17	cluster0
18	18	ASHER ROTH,I Love College.bt	G:\wordNetClusterIda1	M 18	cluster1
19	18	,Complicated.bt	G:\wordNetClusterIda1	M 19	cluster0

Figure 5.5: Output of the CLOPE clustering method on RapidMiner

The tags are enumerated in the clusters and the Table 5.2 shows occurrence of a lyric in tag classes and in the lyric cluster. The rows of the table are represented by ω_i and columns are represented by Θ_j . For instance, if the song is classified to tag classes ω_{13} and ω_{14} also the lyric for the song belong to the cluster Θ_2 . As a result we will increment the cell value of (ω_{13}, Θ_2) and (ω_{14}, Θ_2) by one. The table values obtained by the experiment are:

(ω_i, Θ_j)	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1	0	1	0	0	1	0	0	0	0	0	0	0
2	3	4	3	18	11	11	22	6	33	6	0	4	1
3	0	0	1	1	0	0	2	0	7	0	0	1	0
4	0	0	0	0	0	1	0	1	0	0	0	0	0
5	1	1	1	1	1	0	0	1	0	0	0	0	0
6	35	10	5	1	1	0	0	1	0	1	0	0	2
7	3	0	1	0	1	1	0	2	0	2	0	0	3
8	8	16	5	7	3	5	0	2	3	0	0	0	0
9	3	1	1	1	0	3	5	1	3	2	0	0	0
10	6	20	5	2	2	1	1	1	3	0	0	0	0
11	1	0	0	0	1	0	0	0	0	0	0	0	0
12	0	0	1	1	2	0	0	0	0	0	0	0	0
13	19	25	20	17	16	10	7	9	7	3	4	9	0
14	9	8	3	9	5	7	6	4	0	0	2	0	0
15	1	0	1	0	0	0	0	2	0	0	0	0	0
16	7	12	6	11	9	11	1	0	0	0	6	0	0
17	4	3	5	1	0	4	7	0	0	0	1	0	0
18	8	2	1	0	0	0	0	0	0	0	1	0	0
19	54	36	9	5	10	6	0	0	0	0	3	0	0

Table 5.2: Distribution of tags among lyric clusters

The table clearly shows each lyric cluster has one major class or multiple classes with similar mood and topic. For instance, class 6, 13 and 19 all represent positive mood information that can be present in multiple lyrics. In the latter case the centroid of a cluster is just viewed as extension of cluster's members. The final step in experimentation is the 5-cross validation of the data and we get the result show in Table 5.3.

	Songs with incorrect tags	Songs without incorrect tags
Precision	0.7264	0.9458
Recall	0.8965	0.9458
F-measure	0.8025	0.9458

Table 5.3 Experimental results of tag identification

We compute the Precision, Recall and F-measure to validate the experiments and divide the results into two categories. The column results for songs with incorrect tags give us the true idea of the success of method employed. The results are on the higher percentile and suggest an effective performance. Highly relevant results were received and a small amount of data was missed, hence we get a high F-measure value.

Chapter 6

Conclusions

This thesis delineated a new approach in filtering social tags for the purpose of Music Information Retrieval and Music Mood classification. The goal was to reach a robust and scalable algorithm that integrated different useful ideas and provided high performance. The source code was written purely in Java and using modules from Perl and general data mining software, method was implemented with convenience. The idea was to involve lyrics and social tags within a novel architecture. The song after undergoing classification into an automatically generated category, allows us to compare tags with other tags in the class and decide whether the tag belongs to mood and topic or not.

Based on the experimental results, we can confidently conclude that desired set of goals for building the method has been achieved. We can see the results were great for both songs with and without any erroneous tags. The test and training data included songs from different genres to make sure we get different moods and topic information.

The future work may include increasing the dataset size and building a MIR system that is solely based on the proposed method. Hence, we can better evaluate the approach for wider applications.

BIBLIOGRAPHY

Besson, M., Faïta, F., Peretz, I., Bonnel, A-M. & Requin, J. (1998). Singing in the brain: Independence of Lyrics and Tunes. *Psychological Science*, vol. 9(6), pp. 494-498.

Cooper, M. & Foote, J. (2002). Automatic music summarization via similarity analysis. In *Proc. Int. Symposium on Music Information Retrieval (ISMIR)*, pp. 81–85.

Dhanaraj, R. & Logan, B. (2005). Automatic prediction of hit songs. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR'05)*. London, UK, pp. 488-491.

Ekman, P. (1982). *Emotion in the Human Face*. Cambridge University Press, Second ed.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*, Cambridge, MA: MIT Press.

Futrelle, J. & Downie, J. S. (2002). Interdisciplinary communities and research issues in music information retrieval. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR'02)*. Paris, France, pp. 215-221.

Geleijnse, G., Schedl, M. & Knees, P. (2007). The quest for ground truth in musical artist tagging in the social web era. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR'07)*, Sep. 2007, Vienna, Austria.

He, H., Jin, J., Xiong, Y., Chen, B., Sun, W. & Zhao, L. (2008). Language feature mining for music emotion classification via supervised learning from lyrics. *Advances in Computation and Intelligence, Lecture Notes in Computer Science*, 5370: 426-435. Springer Berlin / Heidelberg.

Hevner, K. (1936). Experimental studies of the elements of expression in music, *American Journal of Psychology*, no. 48, pp. 246–268.

Hu, X., Downie, J.S. & Ehmann, A.F. (2009). Lyric text mining in music mood classification. In *Proceedings of 10th International Society for Music Information Retrieval Conference*, pp. 411–416.

Hu, Y., Chen, X. & Yang, D. (2009). Lyric-based song emotion detection with affective lexicon and fuzzy clustering method. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR'09)*, Kobe, Japan, pp. 123-128.

Juslin, P. N. & Laukka, P. (2004). Expression, perception, and induction of musical emotions: A review and a questionnaire study of everyday listening. *Journal of New Music Research*, 33(3): 217-238.

- Laurier, C., Grivolla, J., & Herrera, P. (2008). Multimodal music mood classification using audio and lyrics. In *Proceedings of the 7th International Conference on Machine Learning and Applications (ICMLA'08)*. Dec. 2008, San Diego, California, USA, pp. 688-693.
- Laurier, C., Sordo, M., Serra, J. & Herrera, P. (2009). Music mood representation from social tags, In *Proceedings of 10th International Society for Music Information Retrieval Conference*, pp. 381–386.
- Law, E., von Ahn, L., Dannenberg, R. B. & Crawford, M. (2007). TagATune: a game for music and sound annotation. In *Proceedings of the 8th International Symposium on Music Information Retrieval*, pp. 361-364.
- Lee, J. H. & Downie, J. S. (2004). Survey of music information needs, uses, and seeking behaviours: preliminary findings. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR'04)*, Barcelona, Spain, pp. 441-446.
- Levy, M. & Sandler, M. (2007). A semantic space for music derived from social tags. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR'07)*. Sep. 2007, Vienna, Austria.
- Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of ECML-98, 10th European Conference on Machine Learning* (Chemnitz, Germany, 1998), 4–15.
- Li, T. & Ogihara, M. (2003). Detecting emotion in music. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR'03)*. Baltimore, Maryland, USA, pp. 239-240.
- Li, T. and Ogihara, M. (2004). Content-based music similarity search and emotion detection. In *Proceedings of the 2004 IEEE International Conference on Acoustic Speech and Signal Processing (ICASSP-04)*, pp. V705-V708.
- Li, T. & Ogihara, M. (2004). Semi-supervised learning from different information sources. *Knowledge and Information Systems*, 7(3): 289-309.
- Li, Y. H. & Jain, A. K. (1998). Classification of text documents. *The Computer Journal* 41, 8, 537–546.
- Logan, B. & Salomon, A. (2001). A content-based music similarity function, Tech. Rep. CRL 2001/02, Cambridge Research Laboratory.
- Logan, B., Kositsky, A. & Moreno, P. (2004). Semantic analysis of song lyrics. In *Proceedings of the 2004 IEEE International Conference on Multimedia and Expo (ICME'04)*. Jun. 2004, Taipei, Taiwan, pp. 827- 830.

- Lv, L. & Liu, Y. (2005). Research and realization of naive Bayes English text classification method based on base noun phrase identification. *Information and Communications Technology, 2005. Enabling Technologies for the New Knowledge Society: ITI 3rd International Conference on*, vol., no., pp.805-812.
- Mandel, M. I. & Ellis, D. P. W. (2007). A web-based game for collecting music metadata. In *Proceedings of the 8th International Symposium on Music Information Retrieval*, pp. 365-366.
- Meena, M. J. & Chandran, K.R. (2009). Naïve Bayes text classification with positive features selected by statistical method. *Advanced Computing, 2009. ICAC 2009. First International Conference on* , vol., no., pp.28-33.
- Pedersen, T., Patwardhan, S. & Michelizzi, J. (2004). WordNet::Similarity: measuring the relatedness of concepts. In *Proceedings of Fifth Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pp. 38–41.
- Russell, J. A. (1980). A circumplex model of affect. *Journal of Personality and Social Psychology*, 39: 1161-1178.
- Schlosberg, H. (1952). The description of facial expressions in terms of two dimensions, *Journal of Experimental Psychology*, Vol. 44. No. 4.
- Schubert, E. (1996). Continuous response to music using a two dimensional emotion space. In *Proceedings of the 4th International Conference of Music Perception and Cognition*. pp. 263-268.
- Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1): 1-47.
- Thayer, R. E. (1989). *The Biopsychology of Mood and Arousal*. New York: Oxford University Press.
- Wang, J., Chen, X., Hu, Y. & Feng, T. (2010). Predicting high-level music semantics using social tags via ontology-based reasoning. In *Proceedings of 11th International Society for Music Information Retrieval Conference*, pp. 405–410.
- Wedin, L. (1972). A multidimensional study of perceptual-emotional qualities in music. *Scandinavian Journal of Psychology*, 13: 241–257.
- Yang, D., & Lee, W. (2004). Disambiguating music emotion using software agents. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR'04)*. Oct.2004, Barcelona, Spain.

Yang, Y., Guan, X. & You, J. (2002). CLOPE: A fast and effective clustering algorithm for transactional data, In *Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 682–687.