Automatic Mood Classification Using TF*IDF Based on Lyrics.

CONFERENCE PAPER · JANUARY 2010

Source: DBLP

2 AUTHORS, INCLUDING:

Menno van Zaanen
Tilburg University

74 PUBLICATIONS **408** CITATIONS

SEE PROFILE

AUTOMATIC MOOD CLASSIFICATION USING TF*IDF BASED ON LYRICS

Menno van Zaanen
Tilburg Center for Cognition and Communication
Tilburg University
Tilburg, The Netherlands
mvzaanen@uvt.nl

Pieter Kanters
Tilburg Center for Cognition and Communication
Tilburg University
Tilburg, The Netherlands
pieterkanters@gmail.com

ABSTRACT

This paper presents the outcomes of research into using lingual parts of music in an automatic mood classification system. Using a collection of lyrics and corresponding user-tagged moods, we build classifiers that classify lyrics of songs into moods. By comparing the performance of different mood frameworks (or dimensions), we examine to what extent the linguistic part of music reveals adequate information for assigning a mood category and which aspects of mood can be classified best.

Our results show that word oriented metrics provide a valuable source of information for automatic mood classification of music, based on lyrics only. Metrics such as term frequencies and tf*idf values are used to measure relevance of words to the different mood classes. These metrics are incorporated in a machine learning classifier setup. Different partitions of the mood plane are investigated and we show that there is no large difference in mood prediction based on the mood division. Predictions on the valence, tension and combinations of aspects lead to similar performance.

1. INTRODUCTION

With the current boost in music sharing (alongside sharing files in other formats) [6], Celma and Lamere [4] state that we see a transformation from albums to individual MP3s and mixes. This also changes the way people interact with their music collection and the demands they place on the software that allows this interaction.

Due to the increasing size of online or digital music collections, users would like to be able to access their collections through more and more advanced means [13]. For instance, users would like to be able to search for songs based on various properties, such as year, genre, play count, online recommendation (Web 2.0) or even based on a set of songs used as seed to find similar ones. One particular property that people use when creating playlists is mood [17]. Currently, this is often done manually by selecting songs that belong to a particular mood and naming the playlist according to the mood, such as "relaxing". Here we investigate the possibility of assigning such information automatically, without user interaction.

In recent years, automatic playlist generation has been introduced to cope with the problem of the tedious and time consuming manual playlist selection. Furthermore, browsing the entire music library manually to select songs for the playlist is felt to be difficult by most music listeners. This becomes especially difficult if music collections become prohibitively large, as the user will not know or remember all songs in it.

In fact, it turns out that people have large amounts of music in their music collection that they never even listen to. This phenomenon is called *The Long Tail* [2] as many songs fall in the region of songs that are hardly ever listened to, which is visualized as a long tail on the right size in a histogram. Automatically generating playlists based on certain properties, such as mood, can expose songs from the long tail and allow for the user to explore "lost music" in their music collections.

Here, we will focus on the automatic classification of music into moods, which is sometimes called "music emotion classification" [18]. Given a music collection containing songs that do not yet have these moods assigned to them (or when adding a new, untagged song to the collection), the process automatically adds the mood tags to the song, allowing selection of songs based on moods.

We think the melodic part of songs contains important information that can help the mood classification [10]. However, we will currently focus on the linguistic aspects of songs only. The idea is that lyrics contain lexical items that emphasize a certain mood and as such can be used to identify the underlying mood. Even though in spoken language, just like in music, other aspects such as loudness and pitch may also be important triggers to identify the song's emotion, we assume here that the actual words can have an emotional load without being spoken or sung. For instance, words such as "happy" or "dead" do not have to be pronounced to have an emotional load. This corresponds to Beukeboom and Semin's idea [3] that mood affects word choice and that lexical items can express moods.

There has been previous work on the influence of lyrics on the mood of a song, such as approaches that concen-

trate on more generic properties present in the lyrics [7], combining music and lyrics using LSA and vector space models [12] or using only the musical aspects [9]. Our approach is quite similar to the work presented in [8], where multi-label classification is performed, resulting in a different classification task. In this paper we concentrate on the influence of the different divisions of classes on the final results.

2. APPROACH

In this article, we describe the development of a machine learning approach to classifying songs, based on lyrics only, into classes that describe the mood of the song [11]. For this, we need several components. Firstly, we need to identify which classes (moods) we are going to classify into. Secondly, we need to select a collection of features that allow us to describe properties of the lyrics. Using these components, we can take a collection of lyrics (describing songs), extract the features and classify the lyrics into their mood class. This mood information can be used to automatically create mood-oriented playlists.

2.1 Class divisions

When building a system that can classify songs into moods, we require a set of classes that describes the allowable moods. For this, we follow [15] in which two dimensions are identified: arousal and valence. These dimensions create a two-dimensional plane with four areas, dividing the plane in positive and negative parts on both dimensions. The moods in this plane range from "angry" and "nervous" (negative valence, positive arousal) to "happy" and "excited" (positive valence and arousal) and from "sad" and "sleepy" (negative valence and arousal) to "relaxed" and "calm" (positive valence, negative arousal). These words used here are only used as examples, indicative of the area in the plane. Other emotionally-laden words can also be placed in this plane.

To be able to work with a more fine grained set of classes, we partition the arousal/valence plane into sixteen parts. This divides both arousal and valence axes into four parts (two on the positive and two on the negative side of the axis). The arousal parts are called A–D and the valence parts 1–4, which leads to individual classes described by a letter and a number. Based on this division, we investigate four different class divisions. The first division uses all sixteen classes. This division is called *fine-grained* and ranges from A1–D4. The second, *arousal*, and third, *valence*, focus only on one aspect of the emotional plane. These class divisions are created by merging four classes. They correspond to only using A–D and 1–4 of the fine-grained division, respectively. Finally, we will use the *Thayer* division, which clusters all fine-grained classes into four areas based on the positive/negative areas in Thayer's arousal/valence plane.

2.2 Features

We will experiment with a collection of features. These are divided into two classes: *global* and *word-based*. The global features describe an aspect of the lyric as a whole. Here, we have experimented with very simple features, which should be treated as an informed baseline. We consider character count *cc* (the number of characters in a lyric), word count *wc* (the number of words in a lyric) and line count *lc* (the number of lines in a lyric).

The word-based features are more complex and use information of the specific words used and their typical occurrence in the lyrics of a particular mood. These features are heavily influenced by metrics from the field of information retrieval [16]. In particular, we use the tf*idf metric and its components. This is a powerful technique to emphasize the importance of a term (word) compared to all documents in a large document collection [14]. Originally, tf*idf was devised to search for relevant documents in large document collections given one or more search terms. The metric is used to compute relevance of the documents with respect to the search terms.

In this research, we consider the use of tf*idf to describe the relative importance of a word for a particular mood class. In contrast to the typical context, however, we start with the lyrics of a song instead of search keywords. The tf*idf value of each word in the lyrics under consideration is used as weights to indicate relevance with respect to mood classes. This allows us to compute which mood is most relevant given lyrics, where the mood is described by the combined lyrics of all songs that have that particular mood assigned.

The approach sketched indicates that we take the lyrics of all songs of a particular mood and combine them as if they are one document. This "document" can be seen as describing a particular mood. This means that there will be as many documents as there are moods. Each mood class corresponds to one document.

The tf*idf metric consists of two components: term frequency (tf) and the inverse document frequency (idf). These components are multiplied when computing the tf*idf.

The first word-based feature is the term frequency (tf). This metric measures the importance of word t_i in document, i.e. mood, d_j with $n_{i,j}$ occurrences of the word in document d_j, divided by the sum of the number of occurrences of all words in document d_j.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad (1)$$

In this situation, it measures the number of times a word occurs with a particular document (or mood). Words occurring more often in the lyrics of a particular mood will have a higher tf for that mood.

The problem with using term frequency is that most words that typically occur very often are function words, such as "the", "a" or "in". These words are not likely to help in classifying lyrics to moods as they do not represent terms that typically describe a mood. What we are really interested in are words that occur in only a sub-set (or only

one) of the moods. The inverse document frequency (idf) measures the importance of the word with respect to a document.

$$idf_i = \log \frac{|D|}{|\{d_j : t_i \in d_j\}|} \quad (2)$$

The total number of documents (representing moods) D is divided by the number of documents in which the word (t_i) appears, and taking the logarithm of that quotient. The idf measures the importance of a word combined with a specific mood against all moods. In this particular situation, idf will be high if it occurs in the text of one or only few moods and will be low when it occurs in multiple moods (or even zero when it occurs with all moods).

The idf value by itself is not particularly useful as it is too course-grained (especially when there are only a handful of moods), but can be multiplied to weigh the tf value, resulting in the tf*idf.

$$tf * idf_{i,j} = tf_{i,j} \times idf_i \quad (3)$$

The tf*idf is used to calculate the relevance of a word for a particular mood: high tf*idf values indicate high relevance of the word to the mood.

The tf*idf provides for one particular word, a score or weight for each of the classes. Lyrics typically contain more than one word, which allows for a more robust computation of the relevance of the mood document for the lyrics under consideration. Practically, we can combine the tf*idf values of all the words in the lyrics for classification by adding the values of the separate words.

Taking the lyrics of all songs of a particular mood as one document results in having between four and sixteen documents (depending on the mood division). This is significantly less than the amount of documents normally under consideration in a tf*idf setting. In fact, many words will occur in all moods, which means that in those cases the idf will be zero which results in a zero tf*idf for all mood classes for that word. This turns out to be a very useful aspect of tf*idf weights in small document collections. In particular, words that do not help in deciding the correct mood of lyrics, such as function words, are automatically filtered out, as their tf*idf value will be zero. There is no way these words can contribute to the final weight of the lyrics, so there is no need to consider these words when analyzing the lyrics.

To investigate whether the zero tf*idf scores really are useful, we also experimented with Laplace smoothing, also known as "add-one smoothing", which reduces the amount of words that have a zero tf*idf. Before computing idf, one is added to the total number of documents. This means that the idf will now always be non-zero, albeit very small. In the case where normally the idf would be zero, the idf will now be small and the same for all classes, but this allows the system to use the information from the tf (which is not possible if idf is zero).

A potential advantage of the smoothed tf*idf is that in the case of all words having a zero non-smoothed tf*idf (for example in the case of very short lyrics), which leads to a zero tf*idf for all classes (and requiring a random choice for the class), the smoothing lets the system back-off to using tf. By not multiplying tf with zero (idf), the tf is retained in the final score, which makes it still possible to classify using tf.

Another extension that is implemented normalizes over the length of the lyrics. The tf can be larger if longer lyrics are used (simply because more words are present in those lyrics). The normalized tf*idf simply divides the tf*idf values computed from the lyrics by the length (i.e. number of words) of the lyrics. This should remove the preference for higher tf in longer lyrics.

Using tf*idf has several advantages. No linguistically motivated tools are required and the approach is inherently language independent. There is no need for the lyrics to be English (or any other language). The simple occurrence of the same words in the training data and in the test data will allow for classification. Obviously, it may be the case that certain words in one language may also occur in another language, but we expect that lyrics in different languages typically use different words. However, more research into the impact of language dependency needs to be done.

3. RESULTS

To measure the effectiveness (and illustrate the feasibility) of using tf*idf in classifying songs into moods, we set up a set of experiments. Taking a collection of songs of which the mood class is known, we extract the lyrics and apply a machine learning classifier to these, allowing us to classify the lyrics into classes based on the different class divisions. For each of these combinations, we discuss the results.

3.1 Experimental settings

To be able to train a machine learning classifier and to evaluate our experiments, we require a data set containing a set of pairs of song (or at least the lyrics) and the corresponding mood. The data set is provided by Crayonroom (`http://www.crayonroom.com/`), a small company creating music applications. The data set comes from their Moody application.

Moody lets users tag songs in iTunes in order to generate mood-based playlists. The tagging is done by manually assigning colors to songs where each color corresponds to a particular mood. The user can choose between 16 moods, which are presented in a four by four square. The colors provided are similar to the hue colors of mood [1]. Note that according to Voong and Beale [17] it is easier for a user to tag using colors instead of tagging using keywords.

The mood information is stored in the comment field of the song's ID3-tag and is exported to Moody's database. The information stored in Moody's database, which contains artist and song title information combined with the mood tag can also be used to automatically tag new songs. This application relies on user input to collect the mood information, but using that information it also helps users tag more songs in their personal collection. As such, it can be seen as a Web 2.0 application, which relies on collaborative tagging of songs.

The mood set used by Moody corresponds well with the two dimensional mood plane by Thayer [15]. The sixteen classes, placed in a four by four grid, can be mapped exactly on the plane with four mood tags in each of the areas in the plane.

Crayonroom provided us with a set of 10,000 random entries from the Moody database. This is a subset of the entire database containing mostly popular songs in different genres. Most of the songs have an English title, but there has not been an explicit selection of songs that have lyrics (as this information is not present in the database itself).

The information we received is a list of pairs of artist and song title, combined with the corresponding mood tag. Based on this information we started collecting the lyrics of the songs. Many lyrics can be found online, so we used the artist and song titles to find the lyrics automatically. This was done by automatically searching a collection of lyrics databases given the artist and song information.

Unfortunately, all spaces were removed from the artist and title fields in the database. This makes automatically finding lyrics hard. Furthermore, there are situations such as "AC/DC" which may be spelled in different ways, such as "AC DC", "AC-DC", or "ACDC". We experimented with several heuristics to re-introduce spaces and reduce the punctuation problems in the artist and song fields. Applying these heuristics and trying to find the resulting artists and song titles led to 5,631 lyrics to be found in the online databases.

The lyrics were then cleaned up and normalized. All HTML information was removed, leaving plain text lyrics. Furthermore, labels such as "chorus", "repeat until fade out" and "4x" were removed as they are not properly part of the lyrics. We realize that this may influence the count of certain words in the lyrics. However, it is often unclear, for instance, where the chorus ends exactly. Similarly, it is often unclear how many repeats are required (in the case of "repeat until fade out"). Simply removing these labels will affect the tf, but apart from manually analyzing the music and correcting all lyrics, we do not see an easy solution. Manual correction is not a feasible alternative at the moment.

From the found lyrics we extracted features and combined them together with their mood tag into machine learning instances. Each instance corresponds to one song. This information is then used for training and testing (allowing for evaluation) in a machine learning setting. The different class divisions and the distributions of instances can be found in Table 1 and Table 2.

Each of the experiments is computed using ten fold cross-validation. This meant that the collection of songs is divided into ten parts and ten experiments are performed, leaving out one part for evaluation. It is important to realize that for the tf*idf features, the tf*idf values for each of the words are recomputed for each experiment. This is needed, because the distribution of words in the training data may be different for each experiment. Intermediate tf*idf tables are computed from the training data first, which are then used to compute the actual tf*idf values for the lyrics to be classified. Similarly, the tf*idf tables will be different for each of the different class divisions.

Fine-grained		Arousal A	B	C	D	
Valence	1	295	236	248	182	961
	2	387	575	564	261	1,787
	3	360	650	531	205	1,746
	4	253	413	338	133	1,137
		1,295	1,874	1,681	781	5,631

Table 1. Distribution of instances: fine-grained (16 classes), Valence and Arousal (both 4 classes).

1 = A3+A4+B3+B4	1,676
2 = A1+A2+B1+B2	1,493
3 = C1+C2+D1+D2	1,255
4 = C3+C4+D3+D4	1,207

Table 2. Distribution of instances: Thayer division (4 classes).

Also keep in mind that for the computation of the tf*idf values, all lyrics belonging to a particular class are combined to serve as one document as described above. When computing the features for each instance separately, the tf*idf values that have been computed beforehand (and stored in a tf*idf table) are used to compute tf*idf scores for each of the classes.

To classify the test data we used TiMBL, a k-NN classifier [5]. This classifier has been developed at Tilburg University and contains a collection of algorithms with many parameters to set. In the experiments described here, we simply used the default parameter setting. This means that the IB1 algorithm (k nearest distances with $k = 1$) is used with the weighted overlap metric and GainRatio weighting. This means that higher accuracy scores may be reached when fine-tuning the classifier parameters. In this paper, we are mostly interested in the feasibility of the approach.

3.2 Experimental results

The results of applying TiMBL to the mood data are summarized in Table 3. The table shows results on the four different mood divisions and different feature settings.

The baseline shown in the table is the majority class baseline. This shows that the data is relatively well balanced as can also be seen from Tables 1 and 2. Keep in mind that the Arousal, Valence, and Thayer divisions all contain four classes, whereas fine-grained is a 16 class division. A completely random distribution of instances would lead to a baseline of 25.00 (four classes) and 6.25 (sixteen classes).

All global features and all of their combinations have worse performance with respect to the baseline. It turns out that the information present in these features is simply not specific enough. For instance, one of the initial ideas we had before we started this research, that the length of the lyrics may be different for lyrics in the different classes,

	Arousal		Valence		Thayer		Fine-grained	
Baseline	33.28		31.74		29.76		11.54	
cc	30.12	(1.77)	29.02	(1.08)	28.15	(1.92)	9.73	(0.67)
wc	31.44	(0.84)	32.59	(1.62)	28.16	(1.65)	11.06	(1.09)
lc	31.81	(1.45)	29.37	(1.69)	27.58	(0.85)	9.85	(0.84)
cc+wc	29.02	(2.26)	28.84	(1.71)	28.47	(2.00)	8.77	(0.90)
cc+lc	29.61	(1.13)	27.77	(1.74)	26.94	(1.74)	8.12	(0.78)
wc+lc	28.92	(1.28)	28.43	(2.05)	27.01	(1.08)	7.74	(0.91)
cc+wc+lc	28.42	(1.69)	27.65	(1.96)	27.03	(1.84)	8.08	(0.84)
tf	33.36	(0.18)	31.77	(0.13)	29.85	(0.15)	11.45	(0.12)
tf*idf	77.18	(1.02)	76.26	(2.03)	75.79	(1.34)	70.89	(1.51)
tf+tf*idf	77.23	(1.02)	76.29	(2.07)	75.85	(1.37)	70.89	(1.50)

Table 3. Mean accuracy and standard deviation of different feature settings and class divisions.

is not true. This is also reflected in the other features used. To allow for classification into moods, more specific information is required.

All advanced features based on tf*idf (apart from tf by itself) significantly outperform the baseline. The tf by itself does not help to classify songs into the correct mood class. The reason for this is that the words that occur most frequently (typically function words, such as "the") greatly outnumber the content words. Even when function words occur approximately the same for each class, minor variations still have a large impact with respect to otherwise more useful content words, which normally do not occur very often. For classification purposes, we are mostly interested in words that help identifying the mood of the lyrics. Words that occur in the lyrics of all moods have limited usefulness. Unfortunately, because function words occur most frequent, they have a large impact on the tf.

When adding idf, which happens with the tf*idf features, the accuracy goes up dramatically. Adding idf removes (or reduces) all weights for words that occur in lyrics of all or several classes. This means that only words that do not occur in lyrics of all moods remain or have a higher impact. This metric seems to coincide with the notion of usefulness that we are trying to implement.

The words with the highest tf*idf score for a particular class are not what we expected. These are words that occur very frequently in only one song. Examples of words with high idf and tf are: "aaah", "dah", or "yoy". However, these words are not often used in classification either.

The results of the experiments using a combination of the tf*idf metric and the tf metric is slightly better than simply using the tf*idf metric only. We expect that this has to do with the situation where there are none or not many words with a non-zero tf*idf in the lyrics. This may occur, for instance, when a song contains non-English lyrics. In that case, the tf*idf values are too often zero, but the tf features allow for a back-off strategy. The differences, however, are minor and non-significant.

As mentioned earlier, we have also implemented a normalized (dividing by the number of words in all the lyrics of a particular mood) and a Laplace smoothed version of the metrics. Since the normalization and smoothing can also be applied together, this leads to three more versions of all the tf and tf*idf experiments described so far. The results of these experiments are not shown in Table 3 as these experiments yield exactly the same mean accuracy and standard deviation as the normal tf and tf*idf features. Obviously, the tf and tf*idf values for each of the words are different in each case, but the classification is the same.

We think that length normalization does not help in classification because the length of the lyrics in each class is too similar. This means that all tf*idf values are divided by a (near) constant. Effectively, similar figures are then used to classify. Furthermore, Laplace smoothing does not help because most of the time the lyrics contain enough non-zero idf words to allow for correct classification. Additionally, when smoothing, words occurring in all classes are used as well, but since they occur in all classes, they do not have a large impact in deciding the correct class.

The different class divisions (arousal, valence, Thayer, and fine-grained) were devised to show which aspect of emotion is easiest to classify. The results show that at least using the technique described here, there is no clear difference. We originally thought that valence would be easier to classify. Positive or negative moods can easily be described using words such as "happy" and "sad". However, the intensity (described by arousal) can just as easily be classified. Most interesting is the fact that the fine-grained class division can be classified effectively as well. Remember that the fine-grained division has sixteen classes whereas the other divisions only have four.

4. CONCLUSION AND FUTURE WORK

This paper describes an attempt to design, implement and evaluate a mood-based classification system for music based on lyrics. The ultimate aim is the automatic assignment of mood-based tags for songs in a users' music database, based on lyrics only. By automatically assigning mood tags to songs, users do not have to assign mood properties to all songs in a potentially large music collection manually. Having access to the mood information ultimately allows for the easy creation of playlists based on moods.

To measure the usefulness of words in lyrics with respect to the mood classes, we used a standard information retrieval metric: tf*idf. This metric is normally used to

measure relevance of terms with respect to documents in a large document collection, but when the same metric is used in a very small set of documents, it shows some interesting and useful properties. The main property used here is that in very small document collections, the tf*idf filters out words occurring in all documents. These are words that are not useful for finding out which document (mood in our case) fits best.

The results show that the tf*idf feature improves the results significantly with respect to the majority class baseline. This shows that tf*idf can be used effectively to identify words that typically describe mood aspects of lyrics. This outcome shows that the lingual part of music reveals useful information on mood.

One has to keep in mind that the experiments reported here only take the linguistic aspects of songs into account. In order to improve results further, other characteristics, such as tempo, timbre or key, should be taken into consideration as well. However, using these aspects requires access to the music (in addition to the lyrics).

The evaluation of the current system is against the mood tags provided by Moody. These tags are based on human annotation. However, it may be that different people assign (slightly) different tags to the songs. We do not know exactly how this is handled in the Moody application, but this may have an impact on the evaluation of the system. Also, we do not know what the inter-annotator agreement is. In future research we need to consider this potential spread of human annotation, for example by taking the confidence of the system for the different moods into account.

A related problem is that the boundaries between the different moods is not clear-cut. A possible solution to this problem and that of the possible variation of annotation is to evaluate using a metric that takes distances between moods in to account. For instance, classifying A2 instead of A1 is better than classifying D4.

5. REFERENCES

[1] A. Albert. Color hue and mood: The effect of variation of red hues on positive and negative mood states. *Journal of the Behavioral Sciences*, 1, 2007.

[2] Chris Anderson. The long tail. *Wired*, 12.10, October 2004.

[3] C.J. Beukeboom and G.R. Semin. How mood turns on language. *Journal of experimental social psychology*, 42(5):553–566, 2005.

[4] O. Celma and P. Lamere. Tutorial on music recommendation. Eight International Conference on Music Information Retrieval: ISMIR 2007; Vienna, Austria, 2007.

[5] Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. TiMBL: Tilburg memory-based learner. Technical Report ILK 02-10, Tilburg University, Tilburg, the Netherlands, November 2002.

[6] B. Haring. *Beyond the Charts: Mp3 and the Digital Music Revolution*. JM Northern Media LLC, 2000.

[7] Hui He, Jianming Jin, Yuhong Xiong, Bo Chen, Wu Sun, and Ling Zhao. Language feature mining for music emotion classification via supervised learning from lyrics. In *Proceedings of the Third International Symposium, ISICA 2008; Wuhan, China*, volume 5370 of *Lecture Notes in Computer Science*, pages 426–435, Berlin Heidelberg, Germany, December 2008. Springer-Verlag.

[8] Xiao Hu, J. Stephen Downie, and Andreas F. Ehman. Lyric text mining in music mood classification. In *Proceedings of the tenth International Society for Music Information Retrieval Conference (ISMIR); Kobe, Japan*, pages 411–416, October 2009.

[9] Xiao Hu, J. Stephen Downie, Cyril Laurier, Mert Bay, and Andreas F. Ehrmann. The 2007 mirex audio mood classification task: Lessons learned. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR08), Philadelphia:PA, USA*, pages 462–467, 2008.

[10] Patrik N. Juslin and Petri Laukka. 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, September 2004.

[11] Pieter Kanters. Automatic mood classification for music. Master's thesis, Tilburg University, Tilburg, the Netherlands, June 2009.

[12] C. Laurier, J. Grivolla, and P. Herrera. Multimodal music mood classification using audio and lyrics. In *Proceedings of the International Conference on Machine Learning and Applications (ICMLA08); San Diego:CA, USA*, pages 688–693, Los Alamitos:CA, USA, 2008. IEEE Computer Society Press.

[13] O.C. Meyers. A mood-based music classification and exploration system. Master's thesis, Massachusetts Institute of Technology, Cambridge:MA, USA, 2007.

[14] J. Ramos. Using tf-idf to determine word relevance in document queries. In *First International Conference on Machine Learning*, New Brunswick:NJ, USA, 2003. Rutgers University.

[15] R.E. Thayer. *The Biopsychology of Mood and Arousal*. Oxford University Press, New York:NY, USA, 1982.

[16] C. J. van Rijsbergen. *Information Retrieval*. University of Glasgow, Glasgow, UK, 2nd edition, 1979. Printout.

[17] Michael Voong and Russell Beale. Music organisation using colour synaesthesia. In *CHI '07: CHI '07 extended abstracts on Human factors in computing systems*, pages 1869–1874, New York:NY, USA, 2007. ACM Press.

[18] Y. Yang, C. Liu, and H. Chen. Music emotion classification: A fuzzy approach. In *Proceedings of the 14th annual ACM international conference on Multimedia; Santa Barbara:CA, USA*, pages 81–84, New York:NY, USA, 2007. ACM Press.