

OH OH OH WHOAH! TOWARDS AUTOMATIC TOPIC DETECTION IN SONG LYRICS

Florian Kleedorfer

Studio Smart Agent Technologies
Research Studios Austria
florian.kleedorfer@researchstudio.at

Peter Knees

Dept. of Computational Perception
Johannes Kepler University Linz, Austria
{peter.knees,tim.pohle}@jku.at

Tim Pohle

ABSTRACT

We present an algorithm that allows for indexing music by topic. The application scenario is an information retrieval system into which any song with known lyrics can be inserted and indexed so as to make a music collection browseable by topic. We use text mining techniques for creating a vector space model of our lyrics collection and non-negative matrix factorization (NMF) to identify topic clusters which are then labeled manually. We include a discussion of the decisions regarding the parametrization of the applied methods. The suitability of our approach is assessed by measuring the agreement of test subjects who provide the labels for the topic clusters.

1 INTRODUCTION

The past few years have seen a tremendous amount of scientific work in the field of music information retrieval. Much, if not most of this work was concentrated on the acoustic properties of music, while the semantic content conveyed by the words of the songs was mostly ignored. There has been some work on song lyrics, but the field has not gained momentum comparable to work on acoustic features of music. In much of the older work lyrics were used to side or contrast with acoustic-based techniques in genre classification of songs [8] or artists [7]. In newer work, lyrics have become sources for metadata generation [9] and, probably inspired by the evolution of Web 2.0, lyrics were found useful as a basis for keyword generation for songs, a technique that may ultimately lead to automatic tagging [12].

In our view, any music browsing or recommendation system is incomplete if it does not incorporate the dimension of the songs' semantic content. It is therefore our goal to create elements of such a system based on the analysis of song lyrics. In the work at hand, we present the building blocks of a system that allows for searching a collection of lyrics by selecting from a set of topics. We describe the formal details of the procedure which we propose for making a collection of songs browseable by topic. Centrally to our algorithm, we apply NMF for clustering the lyrics documents as proposed in [13]. The resulting clusters are labeled manually

so that they can be used as the dimensions of a topic space in which the documents are arranged.

The remainder of this paper is organized as follows: Section 2 describes the dataset we work with, in Section 3 we explain the details of our algorithm. Section 4 gives an account of the quality assessment we performed.

2 THE DATA

The music archive we work with is a subset of the collection marketed through Verisign Austria's¹ content download platform. This subset comprises approximately 60.000 of the most popular audio tracks² by some 6.000 artists. The tracks are affiliated with one or more of 31 genres. The most important genres are "Pop" (10817 songs), "Alternative" (9975 songs), "Hip-Hop" (4849), "Rock" (4632), "Country" (3936) and "Dance/Electronic" (3454).

The lyrics to these songs were extracted from the internet using the method presented in [5], which worked for roughly two thirds of the songs. The reason for this rather low rate of success is that the method we employed is not suitable for some classes of tracks. Such classes comprise of course the purely instrumental songs, but also works that bear a composite *artist* attribute, e.g., "*Mint Condition/feat. Charlie Wilson of the Gap Band*" or older and only locally known music. After removal of duplicates, which are also present in our dataset, we get a corpus of 33863 lyrics.

The lyrics are in 15 different languages, though the vast majority is in English. We found about 300 Spanish songs, roughly 30 in Italian, French, Gaelic and Latin; all other languages are even less frequent.

3 ALGORITHM

The method we propose requires the lyrics collection to be transformed into a vector space model in which each document is represented as a vector defining its affiliation to a set of topics. This transformation encompasses the following steps:

¹ <http://www.verisign.at/>

² i.e., the most frequently accessed tracks within a certain time span

1. *Preprocessing*
Reading text from text files and creating a term-document matrix (TDM).
2. *Term Selection*
Shrinking the TDM by dropping terms and documents.
3. *Term Weighting*
Changing the values for the terms in the TDM.
4. *Clustering*
Clustering the TDM using NMF.
5. *Labeling*
Manually assigning labels to the topic clusters.
6. *Calculating Topic Affiliation*
Automatically assigning topics to documents.

After this preparation the collection can be searched for topics using a query vector that defines the weight for each topic.

In the remainder of this section, we explain this algorithm and its parameters and name techniques for optimizing the results.

3.1 Preprocessing

To begin with, text files are read from the file system, the text is tokenized and a TDM is created without the application of stemming. For this step we use the R package *tm* [10, 3].

A closer look into our corpus reveals a correlation between the length of the text and its quality: the shorter it is, the more unlikely it is to contain the complete lyrics, but rather only a small part of it or, in some cases, only snippets of html code or text on a web page. A viable solution for improving the quality of the corpus is therefore to filter short lyrics. However, some songs actually do have very short lyrics (see Example 3.1); moreover, this is more common in some genres than in others, so filtering introduces a genre bias into the corpus. We decided to accept this trade-off on the one hand because our research is not focused on genres and on the other hand because lyrics that short do not convey much semantic content to analyze, at least not in textual form.³ Our corpus was created using a lower threshold of 200 characters for the length of lyrics. The number of songs in the collection was thus reduced by approximately 3% to 32831.

³ Note that brevity of lyrics may, however, be a valuable piece of information in a general music recommendation scenario.

Oh oh oh whoah
Oh oh oh whoah
Oh oh oh whoah

“Switch 625” by
Def Leppard

That’s right
Have more rhythm
Wooooo!
More rhythm

“Cherry Twist” by
The Chrystal Method

Example 3.1: Two songs with very short lyrics.

Our lyrics harvesting system assigns a confidence value to each extracted text (see [4]), which reflects the strength of agreement found in the different versions of the text. We use this value to identify lyrics to be deleted due to insufficient agreement and hence doubtful quality. The confidence value is a real value $\in [0, 1]$. Informal analysis showed that this value indicates poor quality quite reliably below 0.7. For lyrics with higher confidence value, a difference in this value did not seem correlated with a corresponding difference in the observed quality. These findings led us to set the lower threshold to 0.7, which further reduced the number of songs by 1.5% to 32323.

3.2 Term Selection

Reducing the number of terms is a powerful measure to influence the outcome of the subsequent clustering stage. On the one hand, this decreases the size of the TDM, making any subsequent computation faster and thus allowing more thorough experimentation. On the other hand, experiments showed that term selection can have tremendous influence on the kind of clusters that are produced.

Two approaches are used to cut down on the number of terms. First, stopword lists for a number of languages help to remove the most frequent words that are not considered to be helpful for topic clustering. We used stopwords⁴ for English, Spanish, French and German and a custom stopword list for lyrics⁵. Second, terms and documents are deleted if they do not meet conditions defined by upper and lower thresholds for the document frequency⁶ of a term (f_{max} , f_{min}) and by a minimal term count for documents (t_{min}).

Terms with particularly high or low document frequency are natural candidates for removal from the TDM. Terms occurring in only one document can hardly be useful for clustering, as they only add dimensions to the document representation in which documents are equally dissimilar from one another. We therefore chose to eliminate all terms with a document frequency of 1, which in our case reduced the memory consumed by the TDM by about 3%.

⁴ These lists come with the R package *tm* and are part of the snowball stemmer (<http://snowball.tartarus.org/>)

⁵ This list contains mainly exclamations like “*uuh*” and non-lyrics terms such as “*songlyrics*” or “*inurl*”

⁶ i.e., the number of documents a term occurs in.

Terms with a very high document frequency account for dimensions of the document representation in which a great amount of documents are similar. With respect to clustering, these dimensions may be regarded as noise that makes the real clusters more difficult to discern. This rationale suggests that the clustering result improve when frequent terms are removed. Moreover, removing the frequent terms greatly reduces the space consumed by the TDM: in our case, deleting terms with a document frequency > 500 reduced the memory consumption by 42%. However, the disadvantage of this strategy may be that clusters mainly defined by frequent terms are lost.

Early findings suggested that the use of an upper limit for the document frequency makes the resulting clusters more diverse, but more systematic experimentation convinced us that it was better to use $f_{max} = \infty$.

3.3 Term Weighting

It has been proposed to use TF×IDF [11] as the weighting scheme prior to NMF clustering [13]. In text retrieval applications, the purpose of the weighting function is to amplify the weights of the terms that are most typical for a document and to lower the weights of the other terms. This is because the main use case of the method is *retrieval*, i.e., finding documents that match a user-defined query. In our algorithm, however, the use case to optimize for is not retrieval but *clustering*. During our experiments, we learned that this requires different properties of the weighting function.

We investigated the usefulness of three different weighing schemes in our context: term frequency weighting (i.e., no change to the TDM at all), TF×IDF and binary (i.e., replacing all nonzero entries in the TDM by 1). We found that binary weighting yielded the best results both in terms of individual cluster quality and evenness of the document-to-cluster distribution. Clustering using term frequency weighting produces term clusters in which the frequent terms are clearly overrepresented. Similarly, when applying TF×IDF weighting before clustering, the terms with low document frequency are too important in the resulting term clusters.

3.4 Clustering

We decided to use NMF [6, 13] for automatic topic detection as it is a clustering technique that results in additive representation of items⁷ [6], a property that distinguishes it from most other clustering techniques. We also evaluated latent semantic analysis (LSA) [2], but found that it is not a suitable techniques for large sparse matrices due to its space complexity. Using NMF, the TDM is approximated by the

⁷ e.g., song X is represented as 10% topic A, 30% topic B and 60% topic C

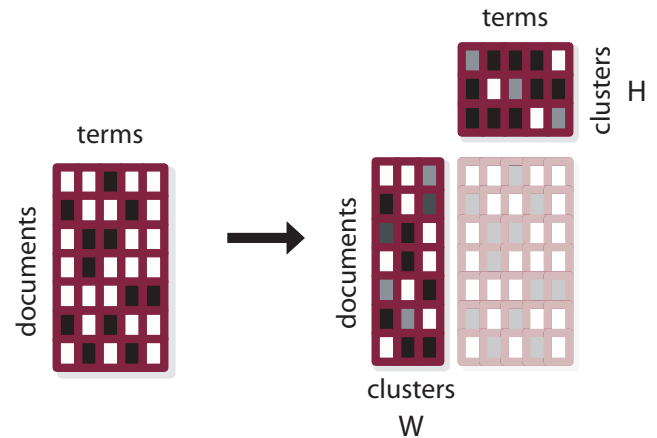


Figure 1. Non-negative factorization of the TDM. The TDM is approximated by the product WH .

matrix product of two matrices of appropriate dimensionality. NMF is parametrized most prominently by the number of clusters that it shall produce, k . More formally, let T be the TDM, W and H the factor matrices, and

$$T_{documents \times terms} = W_{documents \times k} H_{k \times terms} \quad (1)$$

The parameter k is the *inner dimension* of the factorization, i.e. the dimension that both factor matrices share. The approximation of the TDM by the NMF is depicted in Figure 1. For our purposes, the more important matrix of the two factor matrices is H , which contains the weight of each term in each cluster.

A low value for k causes NMF to produce clusters that are actually mixtures of multiple topics, which may be related hierarchically, but this is not necessarily the case. For instance, one cluster may describe the topic “love”, and on a closer look, the sub-topics “loss”, “happy” and “family” are recognizable, while another cluster could at first glance contain only the “gangsta” topic but at the same time be the strongest cluster for all spanish songs. In the first case, the clustering result is acceptable - none of the songs that fall into the cluster would really be misclassified; the only valid criticism is the lack of exactness. In the latter case, a portion of the lyrics that fall into that cluster are clearly misclassified because the cluster combines multiple different “real” topics.

When using high values for k , NMF tends to produce more specific clusters, most of which are quite interesting and useful. One of the drawbacks of this setting, however, is a tendency of NMF to find the same topics multiple times. Another noticeable side effect of high k values is that the important terms simply tend to be strong co-occurents of the first term in the cluster, so the cluster represents not really a topic but rather contains the standard result of co-occurrence analysis.

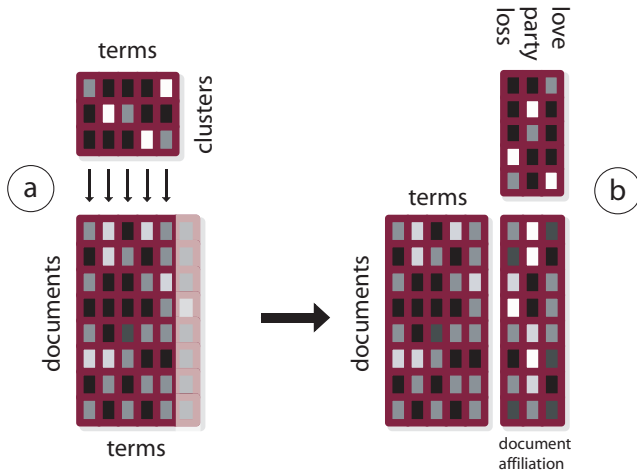


Figure 2. Computation of the documents' affiliation strength to the clusters. The manually assigned cluster labels in this example are 'love', 'party' and 'loss'.

After carefully weighing the benefits and drawbacks of both approaches, we decided to use a high value for the number of clusters in NMF. We conducted experiments for optimizing f_{max} and k , and as a result, we chose to set k to 60 (and f_{max} to ∞ , as stated above).

3.5 Labeling

In order to make the clustering useful to an end user, the clusters need some kind of identifiers associated with them that hint at the nature of each cluster's content. Labeling a cluster was done manually by reading its most important terms and assigning one or more words that best summarised those terms. The exact procedure followed in labeling the clusters, and the results are described in Section 4.

3.6 Calculating Topic Affiliation

The degree of membership of a document in each cluster is computed using the original TDM weighted with term frequencies. This step is shown in Figure 2. First, only the columns (i.e. terms) that were used in the NMF are selected (a). The resulting matrix is multiplied by the transposed factor matrix H from the NMF that contains the term weights per cluster (b). Hence, for each cluster and each document, each term belonging to the document is multiplied by the weight of the term in the cluster and the sum over these products is regarded as the weight of the cluster for the document. After the calculation of this document affiliation matrix, its rows are normalized to a length of 1 in the euclidean norm.

3.7 Query Formulation and Document Retrieval

Retrieval of documents requires defining a query specifying which clusters the documents should belong to. Resulting documents are ranked with respect to that query. A query vector is defined, assigning a weight to each cluster. The cosine similarity values between the rows of the document affiliation matrix and the query vector define the rank of all documents in the query result. The result is sorted in descending order; the first results are the most relevant ones.

4 ASSESSMENT OF LABELING

4.1 Setup

Stages 1-4 of the described algorithm were applied to our lyrics database with the settings $k = 60$ and $f_{max} = \infty$, as explained above. The resulting 60 clusters were used as basis for a two-phased experiment that was inspired by the delphi method [1] which is a common method for obtaining opinion consensus from a group of experts.

In the first phase, test subjects were shown the most important terms⁸ of each cluster and were asked to provide tags that summarise the terms.

In the second phase, the same word lists were shown to the same test subjects, but this time the task was to choose the best tags from those collected during the first phase, but not more than two. These tests were carried out with 6 subjects, all male between 20 and 32 years of age with strong background in computer science and little in music.

4.2 Evaluation Measure

The strength of agreement among test subjects cannot be measured after the first phase because they are completely free in their production of tags, so it is very unlikely that identical tags be used. In phase 2, when the subjects have to choose from the tags produced during phase 1, this is possible because everyone has to perform the very same task. For estimating the significance of the labeling outcome, we compute the probability of the actual result being attained by completely random behaviour on behalf of the subjects. The rationale is similar to that of methods for assessing inter-coder agreement: The lower this probability is, the more evidence there is that the result is due to intelligible features of the data.

During phase 2, there was a given number of tags (m) associated with a given cluster⁹. If a person chose a tag at first position, we assigned a grade of 1 to that tag. If the person chose it for the second position, the grade was

⁸ Those terms with a weight stronger than mean + std.dev. of all term weights in the cluster, sorted by weight. Not more than 30 terms were shown.

⁹ There were at least 2 tags for each cluster, at most 10; mean tag count per cluster was 6.25.

2, all other tags were assigned grade 3. Thus, In the whole session, a $(n \times m)$ grading matrix was created containing the grades for all m tags created by all n test subjects.

The behaviour of the subjects was modeled as follows: For a given cluster, i.e., for a given m , a subject could choose one of the tags as the best tag with a probability of $P(first) = p_1$ and chose none with probability $P(none) = 1 - p_1$. If a tag was chosen best, each tag was equally probable to be chosen with a probability of $1/m$. Then, the person could pick another tag as second best with probability $P(second|first) = p_2$ and no second best tag with probability $P(nosecond|first) = 1 - p_2$. If a tag was chosen as second best, again, all tags were equally probable for this choice with probability $1/(m - 1)$.

The model parameters p_1 and p_2 are computed based on the behaviour of the test subjects. p_1 is defined as the percentage of cases in which at least one tag was chosen, p_2 as the percentage of the former cases in which also a second tag was picked.

Consequently, given m , p_1 and p_2 , the probability $p(g)$ for a tag to get grade $g \in \{1, 2, 3\}$ is

$$p(1) = \frac{p_1 p_2}{m} + \frac{p_1(1-p_2)}{m} \quad (2)$$

$$p(2) = \frac{p_1 p_2}{m} \quad (3)$$

$$p(3) = \frac{p_1 p_2(m-2)}{m} + \frac{p_1(1-p_2)(m-1)}{m} + 1 - p_1 \quad (4)$$

As the result of phase 2, we get a grading matrix for each cluster. The strength of agreement is assessed by computing how probable the column means of such a matrix are a priori. As each value in each column takes the values 1,2 or 3 with the probabilities explained above, the probability for a column to contain g_1 times the grade 1, g_2 times 2 and g_3 times 3 is

$$\frac{n!}{g_1!g_2!g_3!} p(1)^{g_1} p(2)^{g_2} p(3)^{g_3} \quad (5)$$

The likelihood of reaching a given column mean (i.e., average grade for a tag) is the sum of the probabilities of all grade combinations that result in the same or a better mean.

4.3 Discussion

The result of the assesment procedure is a number of tags for each cluster. The tags are associated with a grade and a value which indicates the likelihood for the grade to result from chance agreement among subjects. Table 1 shows the best-graded tag of each cluster, provided that the said likelihood is at most 10%. Due to this selection criterion, only 41 out of 60 tags are shown. As stated in Section 3.4, the same topic are found multiple times; this is reflected here in tags occurring more than once (e.g., “dream”, “emotion” or “gangsta”).

appearance	boys_and_girls	boys_and_girls	broken_hearted
clubbing	conflict	crime	dance
dance	depression	dream	dream
emotion	emotion	family	feelings
future	gangsta	gangsta	gangsta
gangsta	going_out	gospel	hard_times
hiphop	home	leave	listen
loneliness	loss	love	love
love	music	music	nature
party	sorrow	talk	weather
world			

Table 1. Winning tags at a significance level of 10%

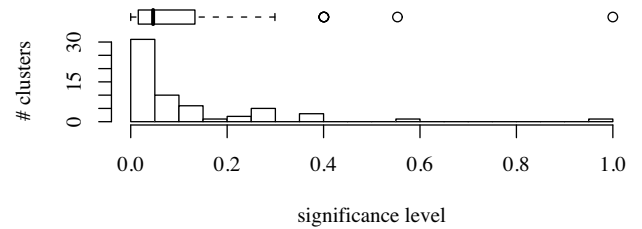


Figure 3. Number of clusters per significance level of the winning tag

Figure 3 shows the levels of significance for the most popular tag of each cluster in 5% steps. In 31 out of 60 clusters, the best tag was agreed on with a less than 5% likelihood of chance agreement. For another 10 clusters, the significance level was between 5 and 10%.

These results suggest that a reasonable portion of the clusters describes discernable topics and that they are reliably tagged.

5 CONCLUSION

The work at hand explains the structure and parametrization of an algorithm for the application of NMF to song lyrics. The focus is on showing the distinct stages of the algorithm and the considerations concerning the choice of parameter values for each stage. The most interesting choices for parameter values, in our view, are a) the high value for k (60 may still not be high enough) and b) the use of binary weighting prior to NMF clustering.

We also present an assessment of the clustering outcome indicating that most of the topic clusters resulting from our algorithm are useful for indexing our music collection. The procedure used for assessment is at the same time an integral part of the algorithm, the labeling stage, which has the convenient property that a statistically interpretable confidence value is calculated for each cluster so that it can be rejected or accepted for use in the subsequent stages.

6 FUTURE WORK

We plan to integrate the proposed method into an existing MIR system¹⁰. Only then the effects of certain parameter settings can be evaluated effectively. We also plan on evaluating our system in a user study with more test subjects than in the work at hand so as to produce statistically significant results. In addition to that, the question arises how similar clusters can be combined or hierarchically arranged for display to end users. Another interesting route of inquiry is hierarchical or iterative NMF clustering with the goal of finding less frequent topics. Taking the whole approach one step further, using NMF may turn out to be only one approach among many for creating weighted word lists (i.e., clusters) for indexing song collections. We consider the use of term co-occurrences most promising for the semi-manual definition of such lists for topics that are known to be contained in the archive but are not found by clustering. Ultimately, this task could be done by end users, allowing them to define and fine-tune topic classifiers themselves.

7 ACKNOWLEDGEMENTS

This work was supported by the Austrian Federal Ministry of Economics and Labor and Austria Wirtschaftsservice (“Impulsprogramm Kreativwirtschaft”); the Austrian Research Promotion Agency (FFG) (FIT-IT project SEMPRE); and the Austrian Science Fund (FWF) under project number L511-N15.

8 REFERENCES

- [1] Norman Dalkey and Olaf Helmer. An experimental application of the delphi method to the use of experts. *Management Science*, 9(3):458–467, 1963.
- [2] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.
- [3] Ingo Feinerer. tm: Text mining package, 2007. R package version 0.2-3.7.
- [4] Peter Knees. Addendum to “Multiple Lyrics Alignment: Automatic Retrieval of Song Lyrics”. Technical Report CPJKU-TR-2008-MLA, Dept. of Computational Perception, Johannes Kepler University, Linz, Austria, 2008.
- [5] Peter Knees, Markus Schedl, and Gerhard Widmer. Multiple lyrics alignment: Automatic retrieval of song lyrics. In *Proceedings of 6th International Conference on Music Information Retrieval (ISMIR’05)*, pages 564–569, London, UK, September 2005.
- [6] Daniel D. Lee and Sebastian H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, October 1999.
- [7] Tao Li and Mitsunori Ogihara. Music artist style identification by semi-supervised learning from both lyrics and content. In *MULTIMEDIA ’04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 364–367, New York, NY, USA, 2004. ACM Press.
- [8] B. Logan, A. Kositsky, and P. Moreno. Semantic analysis of song lyrics. In *Proceedings of the 2003 IEEE International Conference on Multimedia and Expo*, volume 2, pages 827–830, Baltimore, Maryland, USA, 2004.
- [9] Jose P. G. Mahedero, Álvaro Martínez, Pedro Cano, Markus Koppenberger, and Fabien Gouyon. Natural language processing of lyrics. In *Proceedings of the 13th ACM International Conference on Multimedia (MULTIMEDIA ’05)*, pages 475–478, New York, NY, USA, 2005. ACM Press.
- [10] R Development Core Team. R: A language and environment for statistical computing, 2006. ISBN 3-900051-07-0.
- [11] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.
- [12] Bin Wei, Chengliang Zhang, and Mitsunori Ogihara. Keyword generation for lyrics. In *Proceedings of the Eighth International Conference on Music Information Retrieval (ISMIR’07)*, pages 121–122, Vienna, Austria, September 2007.
- [13] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *SIGIR ’03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273, New York, NY, USA, 2003. ACM.

¹⁰<http://musicexplorer.researchstudio.at/>