Comparing Lyrics Features for Genre Recognition

Maximilian Mayerl* **Michael Vötter*** **Manfred Moosleitner** **Eva Zangerle**
Universität Innsbruck
Department of Computer Science
Technikerstraße 21a, 6020 Innsbruck, Austria
firstname.lastname@uibk.ac.at

Abstract

In music information retrieval, genre recognition is the task of automatically assigning genre labels to a given piece of music. Approaches for this typically employ machine learning models trained on content features extracted from the audio. Relatively little attention has been given to using textual features based on a song's lyrics to solve this task. We therefore investigate how well such lyrics features work for the task of genre recognition by training and evaluating models based on various sets of well-known textual features computed on song lyrics. Our results show that textual features produce accuracy scores comparable to audio features. Further, we see that audio and textual features complement each other well, with models trained using both types of features producing the best accuracy scores. To aid the reproducibility of our results, we make our code publicly available.

1 Introduction

Genre recognition is the task of automatically detecting the genre(s) of a given piece of music and often relies on audio features describing the song, including spectral, rhythmic, and tonal features. On the other hand, comparatively little work exists on the effectiveness of lyrics features to build genre recognition models, especially looking into different types of lyrics features.

Ying (2012) looked into using part-of-speech (POS) information calculated on lyrics to detect genre and mood labels for songs. They used a dataset of 600 songs and trained three different machine learning models—k-nearest neighbour, naive Bayes, and support vector machines (SVM)—to determine how well these POS features perform for genre and mood detection. Tsaptsinos (2017) used a hierarchical recurrent neural network model taking word embeddings of raw lyrics as input to predict genres. They performed experiments with 20 and 117 different genres on a dataset of around 450,000 songs and achieved accuracies of 46.42% and 49.50%, respectively. Fang et al. (2017) investigated the effectiveness of various textual features for genre recognition and release date estimation, focusing on discourse-based features, as opposed to features which only take into account single sentences. They found that discourse-based features were effective for genre recognition. Mayer et al. (2008) used features capturing the song's rhythm and features reflecting the structure and statistics of rhymes, tf-idf, and POS, to train kNN, naive Bayes, and SVM algorithms on two datasets with 600 and 3010 songs, respectively. McKay et al. (2010) used a meta-learning based algorithm to predict the genre of songs, by training the algorithm on individual and on combinations of symbolic, lyrical, audio, and cultural features of 250 songs. The results of Mayer et al. and McKay et al. suggest that combining feature groups can improve results compared to training on individual feature groups.

In this paper, we perform a study on the effectiveness of various widely used textual features, computed on song lyrics, for genre recognition. We use the ALF-200k dataset (Zangerle et al., 2018) of songs with English lyrics and add genre information to the songs contained in that dataset via the Last.fm API[1]. We then train machine learning models on the genre recognition task using various sets of widely used textual features and multiple different machine learning models, to determine how well the different types of features perform for genre recognition.

*Authors contributed equally to this work.

[1] https://www.last.fm/api/

Genre	#Songs	Genre	#Songs
alternative	6,828	jazz	1,147
blues	1,101	metal	2,542
country	1,861	pop	7,861
dance	1,539	punk	1,564
electronic	2,677	rap	1,662
funk	791	rnb	1,556
hip hop	2,459	rock	17,234
indie	7,405	soul	2,710

Table 1: Number of songs per genre in our dataset.

2 Dataset

To perform our experiments, we require a collection of song lyrics for a sufficiently large set of songs. Our choice fell on the ALF-200k dataset (Zangerle et al., 2018). This dataset provides lyrics-based textual features for a collection of around 200,000 songs. Since we also require raw song lyrics for our experiments, we downloaded those using the code provided by ALF-200k. Further, we removed duplicates from the ALF-200k dataset based on artist and title where we kept the first occurrence of these songs.

In addition to lyrics, we also need genre labels for the songs in our dataset. As ALF-200k does not provide those, we obtained these via the Last.fm platform. For this, we used the API to search for the songs in our dataset based on their artist and track names, and retrieved the assigned tags from Last.fm. To get genre labels from those tags, we take the 40 most common tags and then only keep tags that represent genres. Additionally, we manually group sub-genres into parent genres based on suffix (e.g., if a song is tagged as *alternative rock*, we assign the genre label *rock* to it), resulting in 16 different genres as shown in Table 1.

Ultimately, we end up with a dataset consisting of 35,045 songs (songs which we did not find on Last.fm were removed) and 16 genre labels. Additionally, the dataset also contains 50 pre-computed textual features per song, taken from ALF-200k, and 10 audio features. The number of songs per genre in our dataset can be seen in Table 1. Note that, as the genre labels are not mutually exclusive, multiple genres can be assigned to a single song.

3 Methods and Experiments

To determine the effectiveness of different types of textual features for the task of genre recognition, and the extent to which those features complement each other, we performed a range of experiments using different types of features and machine learning models. In this section, we will first provide details about the used features and machine learning models, and then elaborate on the experimental setup.

3.1 Features

As mentioned in Section 2, the ALF-200k dataset contains 50 pre-computed textual features and 10 audio features. We used those features and grouped them into five categories (the exact list of features for each category can be found in the code[2]):

- **rhymes**: This group contains features describing the rhymes contained in the song lyrics. This includes features like rhymes per line, rhyme density, number of perfect rhymes etc. Those features were taken from ALF-200k, for which they were computed using the rhyme analyzer tool of Hirjee and Brown (2010). In total, there are 15 features in this group.

- **statistical**: This group contains statistical text features computed over the full text of a song's lyrics. Examples of features in this group include token count, line count, stopword ratio, proportion of novel words, ratio of lines that are repeated, etc. In total, there are 31 features in this group.

- **statistical_time**: This group contains statistical text features that are computed over a song's duration. Overall, there are three features in this group: words per minute, characters per minute, and lines per minute.

- **explicitness**: This group contains only a single feature, which is a binary label, as given by the Spotify API[3], indicating whether a song's lyrics are explicit or not.

- **audio**: This group consists of ten high-level audio features such as acousticness, danceability or tempo, taken from the ALF-200k dataset, for which they were obtained via the

[2] https://github.com/dbis-uibk/NLP4MusA2020

[3] https://developer.spotify.com/documentation/web-api/

Spotify API. We use these features for an audio-based baseline for our experiments.

In addition to those five feature groups containing features stemming from the ALF-200k dataset, we computed the following two additional types of features using the raw lyrics texts of the songs in the dataset with no further pre-processing:

- **tf-idf**: We computed tf-idf vectors over n-grams (uni- to trigrams) on the raw lyrics texts. To limit the length of the resulting feature vector, we only considered the top 2,000 most frequent n-grams.

- **lda**: We also computed feature vectors using Latent Dirichlet Allocation (LDA) (Blei et al., 2003), to capture the topics expressed in the lyrics. We used topic vectors with 25 components (i.e., topics) for our experiments.

3.2 Models

We employed multiple different machine learning algorithms to determine how well the feature groups described in Section 3.1 perform for inferring a song's genres. This was done to be able to quantify how well the features perform, independent of the concrete machine learning model used. In total, we used five different models: k-nearest neighbors (kNN), random forests (RF), forests of extremely randomized trees (ET) (Geurts et al., 2006), support vector machines (SVM), and a self-normalizing neural network model (NN) (Klambauer et al., 2017). For all those models, we performed grid searches with five-fold cross validation to find well-performing parameter settings.

For kNN, RF, ET, and SVM, we used the implementation provided in scikit-learn[4]. For the neural model, we used a simple feed-forward architecture with two hidden layers (both with either 32 or 64 units), both of which use SELU activation and an alpha dropout of 0.1, as described by Klambauer et al. (2017). The neural model was implemented using TensorFlow[5].

3.3 Experimental Setup

As a first step, we computed a random baseline, which assigns every song to every genre with uniform probability (i.e., every given genre has 50% probability of being assigned to a given song). Following that, we calculated a baseline by training and evaluating all of the models described in Section 3.2 on the *audio* feature group. This makes it possible to compare the performance of models using only textual features to models using only audio features.

Then we evaluated the same models on all other text-based feature groups described in Section 3.1 individually for every feature group. Lastly, since we were also interested in seeing how well the textual features complement each other, and how well textual features can complement audio features, we evaluated our machine learning models on (1) a combination of all text-based features groups, and (2) a combination of all text-based feature groups plus *audio*. As mentioned before, all our experiments were performed using five-fold cross validation using the provided methods of scikit-learn.

4 Results

For our evaluation, we used the F_1 score, and since our task is a multi-label problem with an imbalanced distribution of labels, we used macro-averaging to calculate the reported scores.

A summary of the results is given in Table 2, where we depict the F_1 score for every combination of feature group and machine learning model. In every case the reported results are taken from the best model parametrization identified by the grid search. We used the score for the best performing model since we want to determine the effectiveness of the textual features, independent of the concrete machine learning model.

Comparing the random baseline to the results of the other experiments, we observe that every single feature group outperforms the random baseline. We conclude that every proposed feature group (textual feature groups or audio features) carries useful information. The smallest difference between the baseline and an actual feature group is found for both *rhymes* and *explicitness*, which both have a best F_1 score of 0.179, compared to 0.156 for the baseline.

We also observe that only one textual feature group achieved better results than *audio* features: *tf-idf*, with a maximum F_1 score of 0.310 compared to 0.277 for the models using audio features. The next best singular textual feature groups are *lda* and *statistical*, with an almost identical score of 0.233 and 0.231, respectively. The remaining textual feature groups (*rhymes*, *statistical_time*,

[4] We used version 0.23.1 for our experiments.

[5] We used version 2.2.0 for our experiments.

Feature Group	Extra Trees	Neural Network	Random Forest	SVM	kNN	Best
uniform random	—	—	—	—	—	**0.156**
audio	0.187	0.250	0.200	0.191	0.277	**0.277**
rhymes	0.107	0.105	0.116	0.179	0.157	**0.179**
statistical	0.166	0.199	0.169	0.193	0.231	**0.231**
statistical_time	0.140	0.123	0.131	0.194	0.176	**0.194**
explicitness	0.077	0.089	0.077	0.179	0.063	**0.179**
tf-idf	0.141	0.310	0.152	0.211	0.203	**0.310**
lda	0.177	0.171	0.183	0.219	0.233	**0.233**
combined	0.156	0.334	0.162	0.214	0.237	**0.334**
combined + audio	0.155	0.371	0.166	0.220	0.233	**0.371**

Table 2: Summary of our experimental results. For every feature group or a combination thereof, and every machine learning model, we report the F_1 score of the best parametrization found by the grid search.

and *explicitness*) also showed comparable performances, with scores of 0.179, 0.194, and 0.179, respectively. We can also see that combining all textual features (*combined*) leads to improved performance compared to the best performing singular textual feature group (0.334 compared to 0.310 for *tf-idf*). From this, we can conclude that the different textual features capture orthogonal information, and models can benefit from using all of them as their input. Further, adding audio features to the textual features (*combined + audio*) again improves performance. This implies that textual and audio-based features capture orthogonal information and therefore, complement each other.

Lastly, inspecting the performance of individual machine learning models, we observe that for the models using singular feature groups, the best performing machine learning models change between feature groups, with kNN producing the best results for *audio*, *statistical*, and *lda*, SVM producing the best results for *rhymes*, *statistical_time*, and *explicitness*, and the neural model producing the best results for *tf-idf*. For the combined feature sets, the best results are produced by the neural models. As *tf-idf* contains the largest number of features, this could imply that the neural model is best at handling a large number of (sparse) features.

5 Conclusion

In this paper, we investigated the effectiveness of textual features based on song lyrics for genre recognition. We found that such features can be used to train machine learning models which significantly outperform a random baseline. We also found that at least one type of textual feature, namely *tf-idf*, outperforms a simple set of audio-based descriptors.

We further looked into how well different textual features complement each other, and how well they combine with audio features. We found that combining all the textual feature types leads to a significantly increased accuracy, and that adding audio-based features boosted accuracy even more.

In future work, further combinations of features, feature groups, and models should be considered. As we would have expected an increase in performance for all models when joining *combined* with *audio*, we were surprised to only see that effect for the neural network model. To get a deeper understanding for why this happens, it is necessary to analyze the performance in more detail. For example it seems to be valuable to analyze the expressiveness of single features of individual feature groups as well as different combinations thereof. Further, since our results suggest that different machine learning models perform best for different types of features, investigating the use of an ensemble of models might be interesting.

Acknowledgments

We would like to thank Prof. Günther Specht for providing the necessary infrastructure to perform the research reported in this paper.

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.

Jiakun Fang, David Grunberg, Diane T Litman, and Ye Wang. 2017. Discourse analysis of lyric and lyric-based classification of music. In *Proc. 18th International Society for Music Information Retrieval Conference*, pages 464–471. International Society for Music Information Retrieval.

Pierre Geurts, Damien Ernst, and Louis Wehenkel. 2006. Extremely randomized trees. *Machine learning*, 63(1):3–42.

Hussein Hirjee and Daniel G Brown. 2010. Rhyme analyzer: An analysis tool for rap lyrics. In *Proc. 11th International Society for Music Information Retrieval Conference*. International Society for Music Information Retrieval.

Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. 2017. Self-normalizing neural networks. In *Proc. 31st International Conference on Neural Information Processing Systems*, pages 972–981.

Rudolf Mayer, Robert Neumayer, and Andreas Rauber. 2008. Combination of audio and lyrics features for genre classification in digital audio collections. In *Proceedings of the 16th ACM International Conference on Multimedia*, MM '08, page 159–168. Association for Computing Machinery.

Cory McKay, John Ashley Burgoyne, Jason Hockman, Jordan BL Smith, Gabriel Vigliensoni, and Ichiro Fujinaga. 2010. Evaluating the genre classification performance of lyrical features relative to audio, symbolic and cultural features. In *Proc. of 11th International Society for Music Information Retrieval Conference*, pages 213–218. International Society for Music Information Retrieval.

Alexandros Tsaptsinos. 2017. Lyrics-based music genre classification using a hierarchical attention network. In *Proc. 18th International Society for Music Information Retrieval Conference*.

Teh Chao Ying, Shyamala Doraisamy, and Lili Nurliyana Abdullah. 2012. Genre and mood classification using lyric features. In *2012 International Conference on Information Retrieval & Knowledge Management*, pages 260–263. IEEE.

Eva Zangerle, Michael Tschuggnall, Stefan Wurzinger, and Günther Specht. 2018. Alf-200k: Towards extensive multimodal analyses of music tracks and playlists. In *Advances in Information Retrieval - 39th European Conference on IR Research, ECIR 2018*, pages 584–590, Cham. Springer.