Lyrical Analysis For Understanding Music

Gautham Koorma Pratik Aher Romit Barua

1 Introduction

Lyrics Information Processing (LIP) is a field of research that seeks to bridge the gap between Natural Language Processing (NLP) and Music Information Retrieval (MIR) by applying NLP techniques to extract music features from lyrics text that can be used in various MIR applications (Watanabe and Goto, 2020). Music Information Retrieval (MIR) models tend to favor the use of features extracted from the audio of a song over features extracted from the lyrics since all songs have an audio component associated with it. We believe that generating useful features from lyrics and evaluating them may help identify if certain lyrical features are useful in MIR models.

A musical piece is associated with chords that use multiple notes to form a harmonic set of frequencies when heard simultaneously. The harmony in each chord tends to be associated with certain emotions. For example, ”major chords” tend to sound bright and happy, while ”minor chords” tend to sound dark and sad. A musical piece is also associated with a ”key” which is a collection of pitches or notes that set the fundamental harmonic tone for a song. Chord progressions are two or more consecutive chords whose order contributes to the changing tonality of a piece on a given key. Chord structures are often repeated in different sections of a song and can be indicative of the piece's musical genre. Chord progressions are denoted using roman numerals and are determined by the chord's scale distance from the song's root. Chords and chord progressions, just like lyrics, can be analyzed using NLP techniques since they can be represented as textual data.

Popular streaming media platform Spotify provides a developer platform that includes an API for retrieving numeric scores for attributes such as danceability, acousticness, valence, and energy computed from the audio of a song. Our project explores the creation of features from lyrics and chords, and the use of these features to create a regression model that predicts the numeric scores provided by the Spotify API.

We started our work with the hope that the analysis would help us determine whether textual features such as lyrics and chords are good predictors of the danceability, acousticness, valence, and energy of a song. We were also hoping to find some empirical evidence for prior beliefs throughout music theory, such as specific chord patterns believed to be represented in danceable songs, chord structures that have higher and lover valence scores, topics and chord progressions that are found acoustic, and words that have higher energy. With this information, musicians who are trying to maximize specific attributes can use information derived from our most predictive features to help design their music. For example, we might show that acoustic songs have more complex chord patterns, informing musicians who are writing acoustic ballads to pick specific chord types. We began our project with a plan to develop a set of features using chords and lyrics that can be used as predictors for more abstract attributes of a song and hoped that these features would be useful in MIR system. Our analysis shows some of the challenges in creating useful features from lyrics and chords and we hope that the analysis informs other researchers in the LIP domain.

2 Background

(Watanabe and Goto, 2020) introduce LIP as a field of research and discuss techniques for the structural and semantic analysis of lyrics. Structural analysis of lyrics includes rhyme scheme identification, lyrics segmentation, and verse-bridge-chorus labeling techniques. Semantic analysis of lyrics includes mood estimation, topic modeling, and storyline modeling techniques. (Fell, 2020) also discusses various approaches to extract knowledge from song lyrics across textual aspects of structure, content, and perception.

(Mohammad, 2021) provides a survey of various techniques used for sentiment analysis by detecting valence and emotions from text data. Dictionaries with annotated valence and arousal values have been traditionally used for mood estimation. Researchers have also explored the use of bimodal methods that combine audio and lyrical features using neural networks (Delbouys et al., 2018).

(Fell and Sporleder, 2014) experimented with N-gram lyrics models and thirteen feature classes related to the dimensions of vocabulary, style, semantics, orientation, and song structure for three music classification tasks: genre detection, distinguishing best and worst songs and predicting publication time. They observed that an n-gram model is typically a good starting point and that extending the feature space with more sophisticated features improved their results for the classification tasks.

(Gillick and Bamman, 2018) used the Latent Dirichlet Allocation technique (Blei et al., 2003) for generating 50 inferred topics from movie scripts. They then generated a document-topic matrix used in regression models to analyze the impact of the topics from scripts on audio features for the songs in a movie. LDA has also been used by (Kleedorfer et al., 2008), (Sasaki et al., 2014), (Tsukuda et al., 2017) for lyrics topic modeling.

Our review of LIP literature suggests that the association between musical chords and lyrics has been studied by analyzing publicly available data on guitar tablatures (Kolchinsky et al., 2017). Such empirical studies on musical chords reconciled with prior beliefs from music theory that ”major chords” are associated with higher valence lyrics than ”minor chords.” Previous research has also explored the use of chord progression simplification schemes such as N-gram constructions and the distance of chord intervals for music retrieval (Absolu et al., 2010).

Studies have also been done to learn the shared vector representations of lyrics and chords in music wherein such representations are used in classification tasks. (Greer et al., 2019) used the shared representation of chords and lyrics 3-grams on an emotion detection task and achieved a 7 percent increase in model accuracy over a majority class baseline model. However, during our review, we did not find literature that used such shared representations of chords and lyrics combined with other features such as LDA topics, emotions, and text complexity in regression models that predicted attributes of songs such as their danceability, acousticness, valence, and energy.

3 Data

We gathered data from various sources for training our model and running experiments. Our first source of data is the music streaming platform Spotify. We used the Python library Spotipy to interact with Spotify’s Web API. We created a list of 18,000 songs across various playlists and genres from Spotify. A benefit of using Spotify playlists was getting songs from different genres in the same playlist because Spotify’s playlists are generally curated by ”Mood” or ”Vibe.” This approach ensured that our model would not be biased to a particular genre. We also pulled musical attributes[1] like danceability, acousticness, Valence, and energy for each of the 18,000 songs. Spotify computes these attributes using audio signal processing, and the attributes have a score between 0.0 to 1.0. We used these attribute scores as the dependent variable in our regression experiments.

Spotify API’s definition of the four variables we intend to predict using our model is cited below for reference:

- **danceability**: ”Describes how suitable a track is for dancing based on a combination of musical elements, including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable, and 1.0 is the most danceable.”

- **Acousticness**: ”A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.”

- **Valence**: ”A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g., happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g., sad, depressed, angry).”

- **Energy**: ”A measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.”

[1] Spotify API Reference

Our second source of data is the website Genius.com which is the world's largest collection of song lyrics and musical knowledge. We used LyricsGenius, a python client for the Genius.com API, to pull the full text lyrics for relevant songs from the Spotify list by searching for the song using the following method provided by the Genius API: `genius.search_song(<song_name>, <artist_name>)`

We performed data cleaning and pre-processing on the lyrics pulled from Genius.com. Data preprocessing is required for our LDA model discussed further below. Our first step was to clean the data since Genius.com returned bad data for many songs. Some of these issues were due to discrepancies in the song's title retrieved from Spotify and the song's title on Genius.com. For example, songs with the term "Remaster" in the title, such as old rock songs, returned erroneous results for the Genius search. We used regular expressions and string substitutions to identify such cases and fix them before re-running the Genius lyrics retrieval step. However, in some cases, we could not identify why Genius provided us with erroneous results and had to drop the songs entirely. For example, the Genius API would occasionally return screenplay or scripts instead of lyrics for songs. The length of such erroneous results, which included scripts and screenplays were far longer than song lyrics. We dropped items in our dataset where the length of the returned text was greater than the 92nd percentile length in the dataset to account for this bad data.

After removing erroneous data to the extent possible, we used various regular expressions to clean the data further. For example, the lyrics retrieved from Genius had markers that needed to be cleaned, such as [INTRO], [CHORUS], [BRIDGE], [VERSE], etc., along with the artist's/featuring artist's names inside some of the "[]" marker boxes. The lyrics also contained an "EmbedShare" text pattern with some numeric characters that needed to be cleaned. Once the lyrics data was further cleaned, we used pre-processing techniques such as tokenization, stop words removal, punctuation removal, non-alphanumeric character removal, and lemmatization to prepare a set of words from the lyrics for the LDA model.

Our third source of data is ultimate-guitar.com, an online catalog of 1 million+ crowd-sourced chords and tablatures for songs. We wrote a web scraper for ultimate-guitar.com that scrapes the chord progression tablatures for relevant songs in our list of 18,000 songs. The scraper script searches the website for each song in our list and returns relevant results from which we extracted the URL for the first available result and the corresponding tablature from the URL. We used some off-the-shelf python libraries for fetching the URL and scraping the required chord progressions. Like the Genius lyrics pull, one key issue that we faced here was that we needed to drop the song if the search was unable to return a valid result on Ultimate Guitar. Another key issue is the inaccuracy in chord data, especially for less popular songs, since Ultimate Guitar is a crowd-sourced website that relies on individual contributors to add the correct information. Without a more robust validation system, less popular songs cannot be vetted adequately for accurate information.

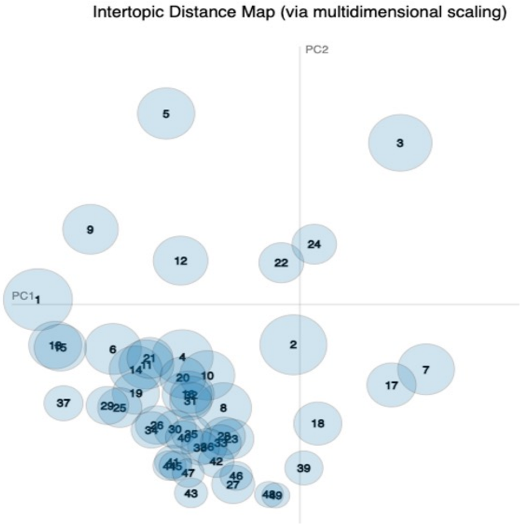
We also experimented with getting a song's emotion as a feature for our regression model. For detecting emotions from lyrics and using the emotions as features, we use text2emotion which gave us a five attribute(anger, happy, sad, surprise, fear) score representation for the text. Another approach we used for emotion detection was the VADER (Valence Aware Dictionary and sEntiment Reasoner) lexicon approach. VADER gives us four scores (neutral, compound, positive, negative) for some input text data. We divided lyrics into equal blocks of data and performed emotion detection on these data blocks. Once we had scores for these individual data blocks, we aggregated them to create scores on the song level. text2emotion and Vader are both lexicon-based approaches.

One issue with emotion detection is that it works well with shorter-length data. We found that emotion detection approaches with long text data yielded diminishing returns. One particular issue with emotion detection in songs is that different parts of the song have different emotions, and these variances are not accurately captured in the overall emotion scores.

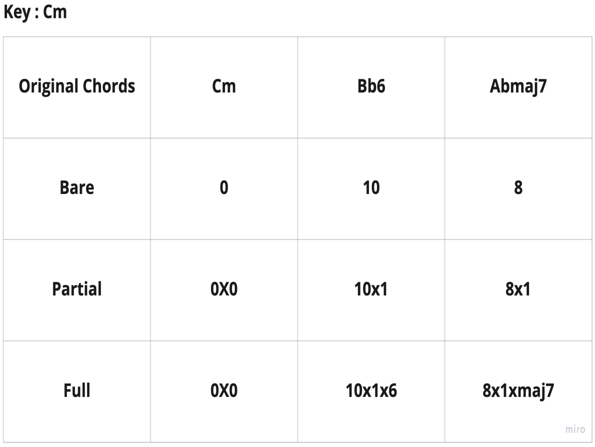
Since we had to drop songs during the lyrics and chords extraction phase, we performed an inner join of the reduced data sets with the list of 18,000 songs to arrive at a reduced list of 12,000 songs we used in our experiments. We split the data set into train and test for training our model, where train consists of 80 percent of our data while test consists of 20 percent of our data.

4 Features

We used LDA to generate a document-topic representation for each song in our combined dataset. For implementing the topic model, we used the Gensim Python library's models.ldamodel implementation. We chose k=50 for our number of topics after running topic coherence tests based on model perplexity scores. The document-topic matrix contains the proportion of each of the 50 topics in each document. We also identified the dominant topic for each song. After generating the topic model once, we pickled the model and used the same one in all our experiments. The image below shows a visualization of the topics weused in our model. The visualization was generated using the library pyLDAvis.



In addition to the LDA features, we generated features by applying NLP techniques to the chord progressions of songs. In music, the distance between two notes is called an interval. A chord is created when two or more notes are played simultaneously. Each chord is characterized and identified by its root note (De Haas et al., 2013). In addition, songs have a tonal center, the most stable note in the piece. Often a song's chords are characterized by the distance between each chord's root note and the song's tonal center. To pick the correct value for the key of a song for normalizing chords, we experimented with various techniques to find the key of the song. Spotify provides us the key of a song, but the key is not always accurate. We therefore used an additional method developed by Ben Ma for the experiments in (Greer et al., 2019) available on the GitHub project page timothydgreer/chord_lyric_reps. We also used the First note key or Last note key technique. We found that in many songs, the First note or the Last Note itself is the root of the song. Based on this observation, we use the first note as the key of the song if the First note and the Last note are same in a song, else we use the note calculated by the Greer-Ma technique described above as the root of the song. We applied a normalization process to the chords by converting them from their letter names (C - F - G) and representing them as interval distances from the root note/key (1 - 4 - 5).



Key : Cm

Original Chords	Cm	Bb6	Abmaj7
Bare	0	10	8
Partial	0X0	10x1	8x1
Full	0X0	10x1x6	8x1xmaj7

We used three types of chord feature types: Full, Partial, and Bare. Bare consists of just the distance of a given chord from the song's tonal center. Partial consists of '1' or '0' depending on if a given chord is major or minor, along with the information in Bare. Full consists of a more complete representation of the chord. For example, if the chord is 'Dmaj7', its representation is '2X0Xmaj7'. We used the 'X' character as a delimiter in the full representation since other common delimiters like '-' gave us some issues. We build a count vectorizer representation for the normalized chords for every song.

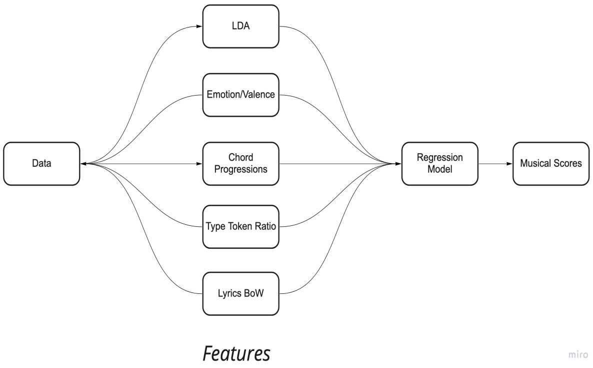
Another feature that we used in our models was text complexity using the Type-Token Ratio (TTR) which is a measure of lexical richness. TTR is the total number of unique words (types) divided by the total number of words (tokens) in each segment of text. We computed the TTR for every song. We then used this score as one of our features for our regression model. The intuition behind using this feature is that danceable songs tend to use simple phrases with repetitive words as opposed to acoustic songs, which may use lyrically rich material to convey deeper themes and metaphors.

We also built a lyrics bag of words representation by using the Count Vectorizer on the cleaned and preprocessed words we used for our LDA model. Finally, we represented the VADER sentiment scores for each song as a feature for regression model as well.

5 Methods

We first built our model using an OLS Regression but ran into three main issues: the model predicted attribute scores outside Spotify's 0.0-1.0 threshold, the regression struggled with the curse of dimensionality, and our results were inconsistent due to the non-linearity in our data. We found that a Random Forest Regressor addressed all these issues. First, the model's predictions are bounded by the lowest and highest values in the training dataset, so the predicted scores remained within Spotify's bounds. Additionally, due to the feature selection mechanism of random forest, we expected our predictions to be less impacted by the curse of dimensionality and allowed us to capture non-linear clusters in our dataset.

Our model was trained first using each feature type (Lyrics Bag-of-Words, Normalized Chord Progression, LDA, VADER, etc.) independently. Then we combined what we felt would be complimentary features to train more holistic and comprehensive models. Due to the high dimensionality nature of the feature set and the presence of sparse matrices, we applied Singular Value Decomposition (SVD) and Document-Frequency limits to reduce the dimensions of our feature set.



6 Analysis

Once the random forest regression models were trained for each attribute, we took the test data set and predicted the scores for the four attributes: danceability, acousticness, valence and energy. For each attribute, we computed four key metrics: Mean Absolute Error (MAE), Percentage Improvement Over Baseline Error, Training Dataset R-Squared and Test Dataset R-Squared.

To generate our baseline against which we compared the predictions, we calculated the Mean Absolute Error between the actual attribute scores and the mean value of each attribute score (majority class). As shown in Figure 1, danceability has the lowest baseline MAE with 0.122 and acousticness had the highest baseline MAE with 0.266. Percentage Improvement over Baseline Error was calculated by taking the change between the model prediction error and the baseline error as a percentage of the baseline error.

As can be seen in the tables above, the Bag-of-Words for lyrics was the most predictive feature set across all Spotify attributes, with a greater than 10% improvement over the baseline in all instances.

The normalized chord progression features were most predictive of acousticness with a 13.2% improvement over the baseline. Additionally, when combined with the Lyrics Bag-of-Words, the model showed a 17.7% improvement over the baseline. Across all other attributes, the normalized chord progression under-performed and did not improve upon the lyrics bag-of-words model.

Additionally, we found that topic models did not add any incremental value to predicting the Spotify attributes over the lyrics Bag-of-Words. In fact, for valence we found that combining the bag-of-words for lyrics and the LDA topic distribution resulted in less predictive models than just using lyrics bag-of-words features.

Attribute	Baseline Error
Danceability	0.122
Acousticness	0.266
Valence	0.205
Energy	0.186

We additionally chose to calculate both the Training R-Squared and Test R-Squared separately. The Training R-Squared explained how well the model was fitted to the training data, while the Test R-Squared provided insight into how generalizable the trained model is to unseen data.

Finally, we found that our VADER method for sentiment analysis was the least predictive feature across all four attributes, including valence. Similar to our other feature set, we found that the VADER sentiment method was most predictive of the Acousticness attribute.Additionally, we find that the models we generated have a tendency to overfit the training data, but do not generalize well to the test data.

Danceability				
Model	MAE	% Improvement Over Baseline	Train R-Sq	Test R-Sq
Lyrics BoW	0.107	12.3%	0.889	0.184
Chord Progression	0.117	4.1%	0.793	0.030
LDA	0.113	7.4%	0.877	0.104
Sentiment	0.117	4.1%	0.847	0.030
Lyrics BoW + Chord Progression	0.107	12.3%	0.887	0.175
Lyrics BoW + LDA	0.107	12.3%	0.888	0.180
Chord Progression + LDA	0.112	8.2%	0.881	0.118

Acousticness				
Model	MAE	% Improvement Over Baseline	Train R-Sq	Test R-Sq
Lyrics BoW	0.225	15.4%	0.882	0.110
Chord Progression	0.231	13.2%	0.812	0.038
LDA	0.236	11.3%	0.875	0.114
Sentiment	0.247	7.1%	0.845	0.012
Lyrics BoW + Chord Progression	0.219	17.7%	0.889	0.147
Lyrics BoW + LDA	0.225	15.4%	0.888	0.126
Chord Progression + LDA	0.223	16.2%	0.883	0.124

Across all models for the four attributes, we found that the training R-Squared values are greater or equal to 0.8. On the other hand, we found the R-squared for the test dataset to be below 0.2, with the chord progression and sentiment features often having an R-Squared of less than 0.05. We tried doing Random Search based hyperparameter tuning to find parameters that would help increase our training R-squared value. However, we observed that while the training R-squared increased, the test R-squared continued to remain below 0.2.

Valence				
Model	MAE	% Improvement Over Baseline	Train R-Sq	Test R-Sq
Lyrics BoW	0.182	11.2%	0.888	0.165
Chord Progression	0.192	6.3%	0.814	0.063
LDA	0.187	8.8%	0.881	0.110
Sentiment	0.196	4.4%	0.852	0.033
Lyrics BoW + Chord Progression	0.187	8.8%	0.884	0.120
Lyrics BoW + LDA	0.183	10.7%	0.888	0.148
Chord Progression + LDA	0.187	8.8%	0.886	0.125

Energy				
Model	MAE	% Improvement Over Baseline	Train R-Sq	Test R-Sq
Lyrics BoW	0.164	11.8%	0.883	0.149
Chord Progression	0.176	5.4%	0.814	0.01
LDA	0.169	9.1%	0.876	0.113
Sentiment	0.179		0.845	0.01
Lyrics BoW + Chord Progression	0.164	11.8%	0.889	0.146
Lyrics BoW + LDA	0.165	11.3%	0.885	0.151
Chord Progression + LDA	0.166	%	0.883	0.137

7 Conclusion

Overall, we found that our feature engineering of lyrics and chord progressions alone did have some predictive capabilities of Spotify's audio signal-based attributes relative to the baseline, but may add more value in a multi-modal model. As shown by the large decrease in the test R-Squared value in comparison to the training R-Squared value, we believe that our model did not generalize well to unseen data. During our modeling process we faced two key issues that we believe led to underperformance of our model.

First, cleaning and collecting accurate data was a challenge. The lyrics API, Genius, often gave bad results and required significant pre-processing before use in feature generation such as bag-of-words and LDA. Similarly, our chord progression data and key selection faced issues in terms of accuracy and reliability. As previously mentioned, we used Spotify's key, Timothy Greer's method and First/Last Chord to determine the key of the song. However, there was often inconsistency between the different sources, requiring us to create a custom system to choose what we thought was reasonable. It is likely that many songs had an incorrect key during the normalization process. Similarly, we found that the ultimate guitar tab was inconsistent in terms of chord accuracy and level of detail provided for each individual chord. By creating a hand-annotated and verified chord dataset, we believe that our model could further improve and find accurate patterns.

Second, we struggled with the curse of dimensionality. Using the bag-of-words with n-grams for lyrics and chord progressions, we generated thousands of features represented in sparse matrices. To address the high dimensionality issue, we applied two key dimension reductions methods: increasing the count frequency requirement in the count vectorizer and applying singular value decomposition (SVD). While this improved the accuracy, it drastically reduced the interpretability of the output for our models. In future work, it will be important to try new feature selection methods that focus on maintaining the interpretability. Additionally, increasing the number of songs may help us reduce impact of the high dimensionality.

Our original goal was to empirically prove notions that musicians have about lyrics, chord progressions in relation to specific song attributes like danceability, valence, acousticness and energy. While our prediction models had some improvement over the baseline, the loss of interpretability means that we are at this point unable to answer these questions as we are unable to interpret the features that led to this improvement. Code to support this work can be found at https://github.com/gthmk/anlp21-project

References

Brandt Absolu, Tao Li, and Mitsunori Ogihara. 2010. Analysis of chord progression data. In *Advances in Music Information Retrieval*, pages 165–184. Springer.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

W Bas De Haas, Frans Wiering, and Remco C Veltkamp. 2013. A geometrical distance measure for determining the similarity of musical harmony. *International Journal of Multimedia Information Retrieval*, 2(3):189–202.

Rémi Delbouys, Romain Hennequin, Francesco Piccoli, Jimena Royo-Letelier, and Manuel Moussallam. 2018. Music mood detection based on audio and lyrics with deep neural net. *arXiv preprint arXiv:1809.07276*.

Michael Fell. 2020. *Natural language processing for music information retrieval: deep analysis of lyrics structure and content*. Ph.D. thesis, Université Côte d'Azur.

Michael Fell and Caroline Sporleder. 2014. Lyrics-based analysis and classification of music. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical papers*, pages 620–631.

Jon Gillick and David Bamman. 2018. Telling stories with soundtracks: an empirical analysis of music in film. In *Proceedings of the First Workshop on Storytelling*, pages 33–42.

Timothy Greer, Karan Singla, Benjamin Ma, and Shrikanth Narayanan. 2019. Learning shared vector representations of lyrics and chords in music. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3951–3955. IEEE.

Florian Kleedorfer, Peter Knees, and Tim Pohle. 2008. Oh oh oh whoah! towards automatic topic detection in song lyrics. In *Ismir*, pages 287–292.

Artemy Kolchinsky, Nakul Dhande, Kengjeun Park, and Yong-Yeol Ahn. 2017. The minor fall, the major lift: inferring emotional valence of musical chords through lyrics. *Royal Society open science*, 4(11):170952.

Saif M. Mohammad. 2021. Sentiment analysis: Automatically detecting valence, emotions, and other affectual states from text. In Herb Meiselman, editor, *Emotion Measurement (Second Edition)*. Elsevier.

Shoto Sasaki, Kazuyoshi Yoshii, Tomoyasu Nakano, Masataka Goto, and Shigeo Morishima. 2014. Lyricsradar: A lyrics retrieval system based on latent topics of lyrics. In *Ismir*, pages 585–590.

Kosetsu Tsukuda, Keisuke Ishida, and Masataka Goto. 2017. Lyric jumper: A lyrics-based music exploratory web service by modeling lyrics generative process. In *ISMIR*, pages 544–551.

Kento Watanabe and Masataka Goto. 2020. Lyrics information processing: Analysis, generation, and applications. In *Proceedings of the 1st Workshop on NLP for Music and Audio (NLP4MusA)*, pages 6–12.