Smart Search Engine for Lyrics with Sentiment Analysis

Development Track
Project Report

Stanford Zhou (szhou36)
Le Chang (lechang3)
Yidi Yang (yyang160)
Yangyang Dai (yd10)

System Link: http://lechang3.web.illinois.edu/CS510_Project/
Github Link: https://github.com/yyang160/CS510_Project/tree/master/FinalProject

1. Problem Statement

Our project is a search engine for songs with bonus features for sentiment analysis. Music search engine is a saturated field with many existing examples such as Spotify, Apple Music, etc. These search engines provide all songs relative to a query as a traditional search engine does. With this project, we would like to extend the functionality to analysis. In addition to providing information about the songs metadata, the project provides the sentiment of the songs based on it's lyrics. The project can also recommend other songs with close sentiment scores from the artists of the most relevant search results from the query.

2. Features

- Search Engine

When a user searches one or a few words, our search engine will return the top 30 songs with the most relevant lyrics according to these words. For each song we display the name, artist, lyrics, link as well as the overall sentiment of the song. We also show a pie chart indicating the distribution of sentiments over the results corresponding to this query. A list of recommended songs are shown following the results.

- Sentiment Analysis

For each song in our dataset, we classify them based on their sentiment(positive, negative, neutral). In addition, we store "sentiment scores" that reflect the intensity of the sentiment. For example, "excellent" will have a higher score than "good", and "awful" will have a lower score than "bad". With this feature, we can extract songs with similar intensity relative to a song and recommend them to the user.

This search engine provides information of sentiment level of search results based on NLTK sentiment analysis, which most other online music search engines don't do, so that the users can have a general impression on a song. And it also provides a sentiment level distribution in the form of pie chart for each query.

3. Dataset

We used lyrics from 55000+ songs in English from LyricsFreak as our dataset for our search engine and sentiment analysis. We obtained the data from here. The data set is a CSV file, and each column corresponds to the data for one song (song name, author, link to it's LyricsFreak page, and the lyrics). The total size of the dataset is 72.4MB.

4. Methodology

a. Algorithms for Searcher

We use Metapy as our tool to achieve the function of searching, but make some modifications and extensions.

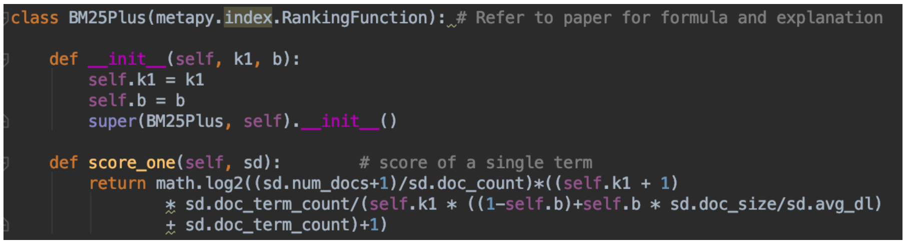
- Ranker

  We use BM25 Plus as introduced in this paper. Because this method alleviates the effect of the document length, and in our case, our documents are relatively short (lyrics). BM25 Plus is claimed to be "no matter how long the document, a single occurrence of a search term contributes at least a constant amount to the retrieval status value." We chose 1 for the parameter δ as suggested on the paper.

  Below is the formula of BM25 Plus:

$$rsv_q = \sum_{t \in q} \log\left(\frac{N+1}{df_t}\right) . \left(\frac{(k_1+1).\, tf_{td}}{k_1 . \left((1-b) + b \,. \left(\frac{L_d}{L_{avg}}\right)\right) + tf_{td}} + \delta\right)$$

  We use metapy.index.RankingFunction to implement the ranker, code snippet as below:



```
class BM25Plus(metapy.index.RankingFunction): # Refer to paper for formula and explanation

    def __init__(self, k1, b):
        self.k1 = k1
        self.b = b
        super(BM25Plus, self).__init__()

    def score_one(self, sd):        # score of a single term
        return math.log2((sd.num_docs+1)/sd.doc_count)*((self.k1 + 1)
                * sd.doc_term_count/(self.k1 * ((1-self.b)+self.b * sd.doc_size/sd.avg_dl)
                + sd.doc_term_count)+1)
```

- Evaluation

  This improved BM25 algorithm is proved to outperform the original BM25 by the authors of the paper. We also perform a simple evaluation (suggested by TAs on the feedback on our project proposal) on the two rankers using our datasets. We use some original lyrics that surely exists in the songs in the dataset as queries to test if the first song the rankers returned is a song that contain the query line exactly. It turns out BM25

Plus is the better one, providing more desired results compared to the original BM25.

- Further tuning based on sentiment of the query

  After obtaining the results from BM25 Plus, we further tuning the results list with the sentiment of the query. Our goal is to make the ranks of those songs that have similar sentiment levels to that of the query higher. So our final algorithm penalizes the the ranking scores of the songs that appear in the results list but have opposite sentiment level.

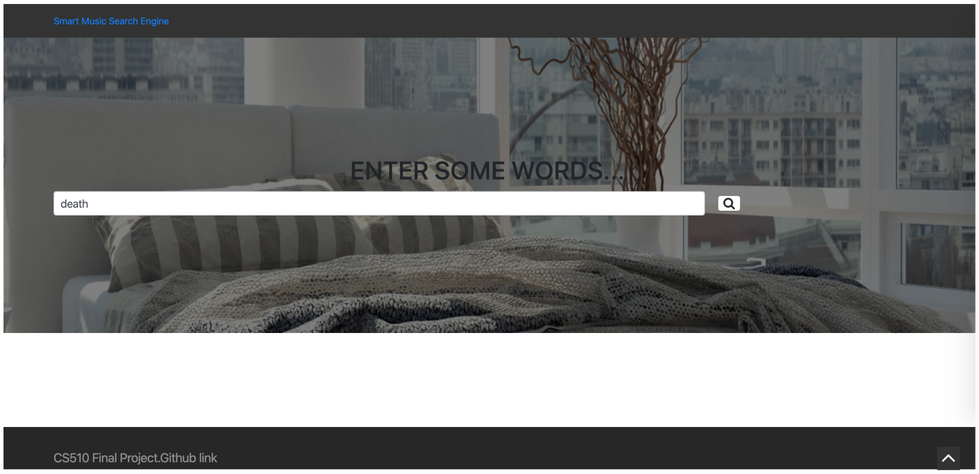
b. Sentiment Analysis

We use the python nltk sentiment VADER library functions to compute the sentiment intensity scores of the lyrics and the queries. VADER(Valence Aware Dictionary and sentiment Reasoner) uses a lexicon of sentiment-related words as a reference to compute scores for strings. The result is 4 metrics: positive, negative, neutral, and compound. The first 3 results measure the portion of the input string belonging to that particular sentiment, while the compound score is a measure of the overall sentiment of the input string, on a scale of (-1, 1) negative to positive. For the purposes of the project, we used the compound score to measure the sentiment intensity of the lyrics.

5. How to Use

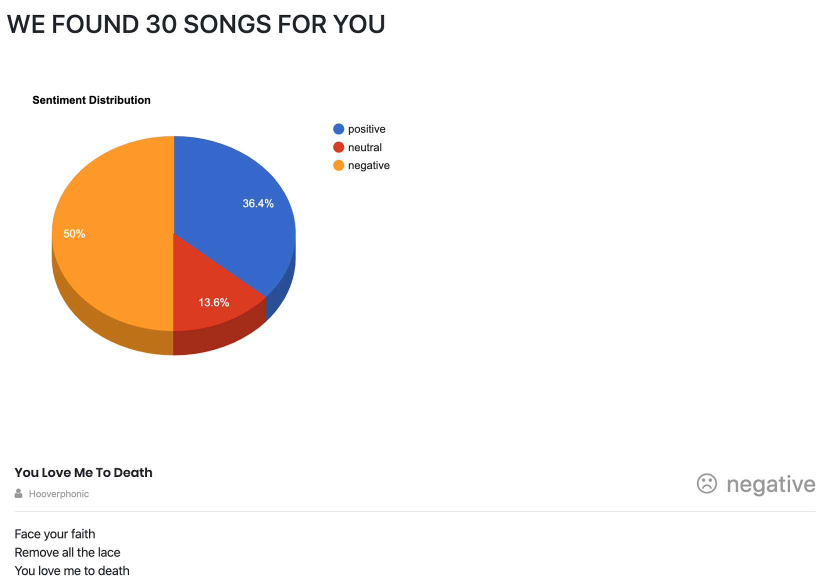
a. Go to http://lechang3.web.illinois.edu/CS510_Project/.

b. Enter some words into the text box.



death

c. Click search.



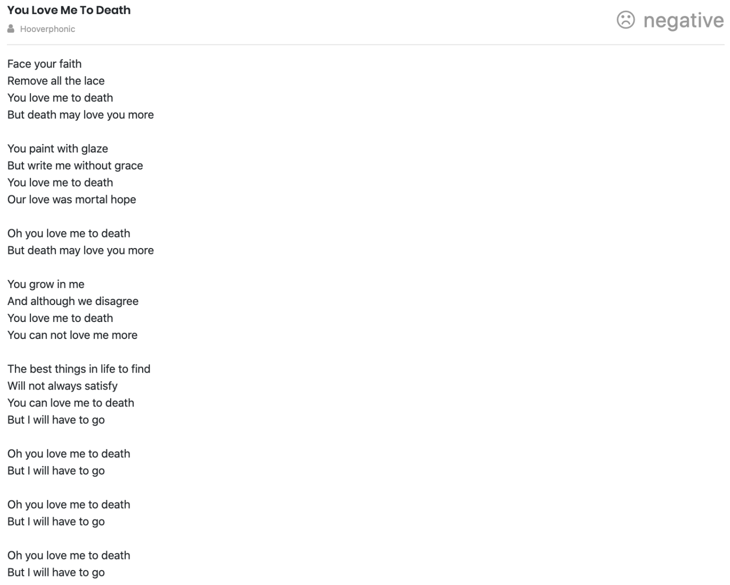
WE FOUND 30 SONGS FOR YOU

You Love Me To Death

Hooverphonic

Face your faith
Remove all the lace
You love me to death

d. The search engine will display a pie chart of the distribution of the sentiment among the search results, followed by the actual search results (songs with singer, link, sentiment, and lyrics).



You Love Me To Death

Hooverphonic

Face your faith
Remove all the lace
You love me to death
But death may love you more

You paint with glaze
But write me without grace
You love me to death
Our love was mortal hope

Oh you love me to death
But death may love you more

You grow in me
And although we disagree
You love me to death
You can not love me more

The best things in life to find
Will not always satisfy
You can love me to death
But I will have to go

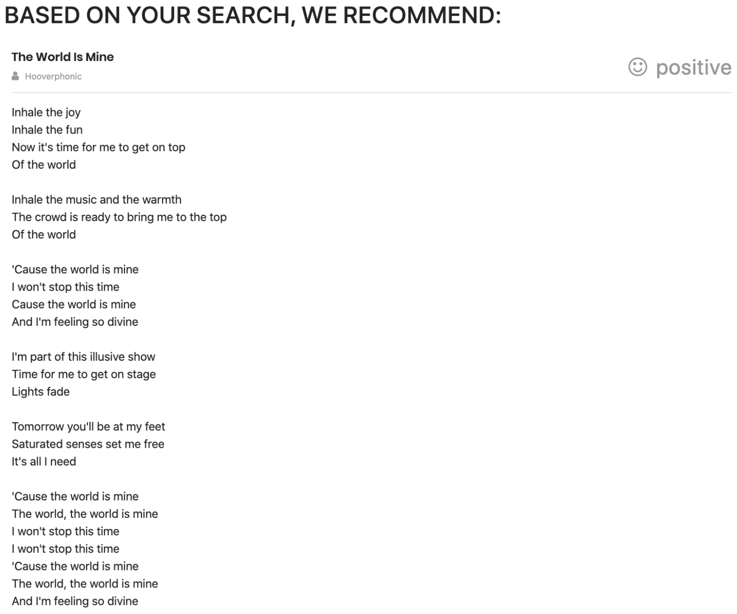
Oh you love me to death
But I will have to go

Oh you love me to death
But I will have to go

Oh you love me to death
But I will have to go

e. The title of each song is clickable and will redirect you to the song's page on www.lyricsfreak.com, where more information of the song is available and users can listen to the song.

f. Scroll down the page, following the search results is a recommendation list based on the search results.



BASED ON YOUR SEARCH, WE RECOMMEND:

The World Is Mine
Hooverphonic

☺ positive

Inhale the joy
Inhale the fun
Now it's time for me to get on top
Of the world

Inhale the music and the warmth
The crowd is ready to bring me to the top
Of the world

'Cause the world is mine
I won't stop this time
Cause the world is mine
And I'm feeling so divine

I'm part of this illusive show
Time for me to get on stage
Lights fade

Tomorrow you'll be at my feet
Saturated senses set me free
It's all I need

'Cause the world is mine
The world, the world is mine
I won't stop this time
I won't stop this time
'Cause the world is mine
The world, the world is mine
And I'm feeling so divine

6. Reference

Dataset: https://www.kaggle.com/mousehead/songlyrics
Metapy Library: https://github.com/meta-toolkit/metapy
BM25 Plus: http://www.cs.otago.ac.nz/homepages/andrew/papers/2014-2.pdf
NLTK VADER Sentiment Analysis: https://github.com/cjhutto/vaderSentiment