

# CS6200 Information Retrieval – Final Project Report

## Lyric Search

Name: Luming Yang

NUID: 001030060

GitHub Source Code: [https://github.com/LumingYangRose/CS6200\\_Information\\_Retrieval.git](https://github.com/LumingYangRose/CS6200_Information_Retrieval.git)

### Introduction:

Listening to music can be entertaining and beneficial to health. Music offers amplification and encouragement. A good piece of music can release the pressure that built up in daily life and bring latent emotions to the front of our consciousness. As of June 2019, 68 percent of adults aged between 18 and 34 years old reported listening to music every day. Apple Music, Spotify, and Pandora Radio ranked top of the popular online music streaming services. With the help from these modern music applications, people are able to read lyric lines asynchronously with the music.

As a key component of songs, lyrics greatly contribute to the prevalence of music. Lyrics allow songwriters to put their emotional expression into the music through the form of language. Lyric also vividly documents current culture and society. Social scientists studied lyrics to learn about the trend of popular culture and the change of history reflected through songs. Lyrics plays a meaningful role for the public as well. When listening to music, people relate their past experience to the content of lyrics, resonate with the songwriters, and see things from their perspective. That explains the typical needs of checking lyrics after people listen to a loved song, the name of which they may or may not know before.

Online music applications provide *Lyric-Search feature*, that allows users to search lyrics by track, album, artists, and lyric sketches. In other words, by typing in the name of the song, the name of the album, the artist who sing or create the song, or parts of the lyrics, users are able to retrieve the lyrics of the song they are looking for. This stand-alone project intends to implement a lyric-search engine that satisfies users' need of fetching lyrics.

## Problems identified:

The focus of research is a website named: lyricfinder.org:

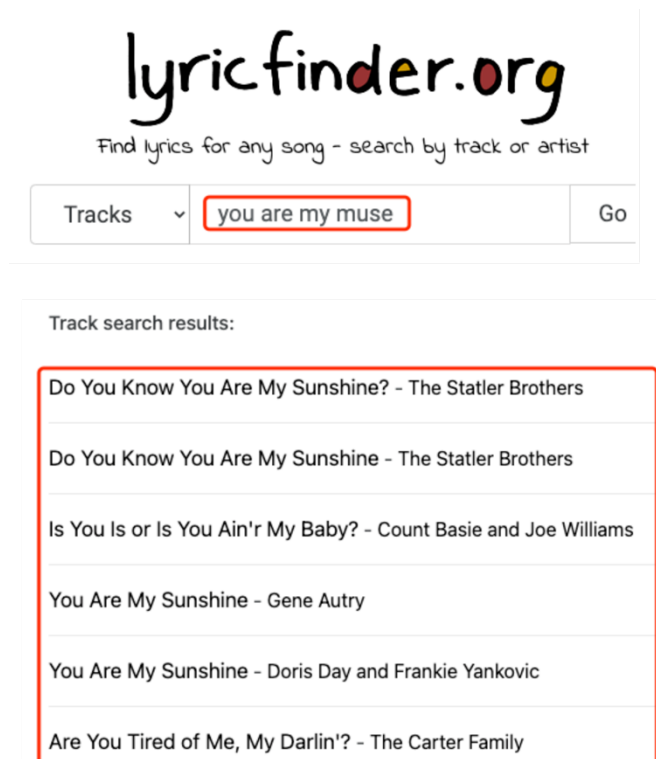
<https://www.lyricfinder.org/>

As a popular music search website, lyricfinder.org is among the top three lyrics search website when one queries Google search for lyrics. It supports field query of four kinds: Tracks, Lyrics, Albums, and Artists through a main search box. Users could also retrieve songs through alphabetical catalog of artists names on the header of the navigation page. On the average, lyricfinder.org satisfies the lyric search need. However, there are also some issues with its searching design that hurts the searching quality and hinders the user experience.

Here are the types of issues identified in searching on lyricfinder.org:

### 1. Weight Distribution among Query Terms:

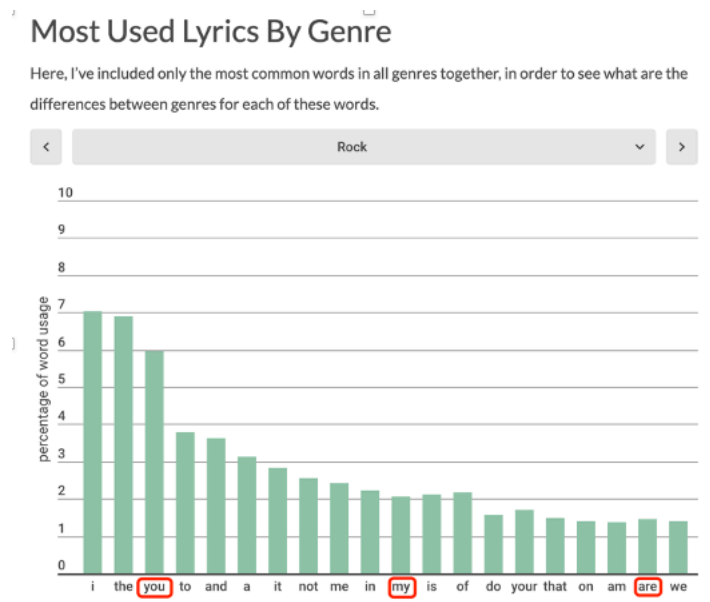
Here is an example:



The screenshot shows the lyricfinder.org website. At the top, the logo 'lyricfinder.org' is displayed with the tagline 'Find lyrics for any song - search by track or artist'. Below this is a search bar with a dropdown menu set to 'Tracks'. The search input field contains the text 'you are my muse', which is highlighted with a red box. To the right of the input field is a 'Go' button. Below the search bar, the section 'Track search results:' is shown. A red box highlights a list of six search results:

- Do You Know You Are My Sunshine? - The Statler Brothers
- Do You Know You Are My Sunshine - The Statler Brothers
- Is You Is or Is You Ain'r My Baby? - Count Basie and Joe Williams
- You Are My Sunshine - Gene Autry
- You Are My Sunshine - Doris Day and Frankie Yankovic
- Are You Tired of Me, My Darlin'? - The Carter Family

The query entered is “*you are my muse*”. The track search results are mostly dominated by “*You Are My Sunshine*”. It is obvious to see that the search results mainly match with the first three terms in the query-- “*you*”, “*are*”, and “*my*”. None of the search result is about the fourth term: “*muse*”. When we analyze the query, we can see that the frequency of each term in the query is quite different.



As the above graph showed (statistics from Coding in Tune), “*you*”, “*are*”, and “*my*” are among the most used lyrics. While the fourth term “*muse*” is considerably infrequent compared to the others. Therefore, searching for songs containing “*muse*” could be much more relevant to the query than searching for common terms in lyrics. From a technical aspect, the fourth term should gain much more weight than the first three terms. As this search result suggests, there may be issues with the inverse document frequency design of lyricfinder.org.

## 2. Duplicate Results that Link to the Same Page

As the previous example showed, there are two duplicate results for the query “*you are my muse*”: “*Do You Know You Are My Sunshine*”. When some other different queries are tried, duplicate results persist to exist.

Here are some examples:

Example 1:

lyricfinder.org  
Find lyrics for any song - search by track or artist

Tracks ▾ you are too precious Go

You Are Too Beautiful - Oscar Peterson

You Are Too Beautiful - Oscar Peterson

Are You from Dixie ('Cause I'm from Dixie Too) - Jerry Reed

You Are Too Beautiful - Thelonious Monk

Are You From Dixie (Cause I'm From Dixie Too) - Jerry Reed

Detailed description: This is a screenshot of the lyricfinder.org website. The search bar contains the text 'you are too precious'. Below the search bar, there are five search results. The first two results are 'You Are Too Beautiful - Oscar Peterson', which are highlighted with a red rectangular box. The third result is 'Are You from Dixie ('Cause I'm from Dixie Too) - Jerry Reed', highlighted with a blue rectangular box. The fourth result is 'You Are Too Beautiful - Thelonious Monk'. The fifth result is 'Are You From Dixie (Cause I'm From Dixie Too) - Jerry Reed', also highlighted with a blue rectangular box.

Example 2:

lyricfinder.org  
Find lyrics for any song - search by track or artist

Tracks ▾ smoke gets in your eyes Go

Track search results:

(When Your Heart's on Fire) Smoke Gets in Your Eyes - Ray Conniff

(When Your Heart's on Fire) Smoke Gets in Your Eyes - Vic Damone

Smoke Gets In Your Eyes - The Platters

Smoke Gets In Your Eyes - The Platters

Smoke Gets In Your Eyes - The Platters

Smoke Gets In Your Eyes - Dinah Washington

Smoke Gets in Your Eyes - Nat King Cole

Smoke Gets In Your Eyes - Artie Shaw

Detailed description: This is a screenshot of the lyricfinder.org website. The search bar contains the text 'smoke gets in your eyes'. Below the search bar, there are seven search results. The first two results are '(When Your Heart's on Fire) Smoke Gets in Your Eyes - Ray Conniff' and '(When Your Heart's on Fire) Smoke Gets in Your Eyes - Vic Damone'. The next three results are 'Smoke Gets In Your Eyes - The Platters', which are highlighted with a red rectangular box. The last two results are 'Smoke Gets In Your Eyes - Dinah Washington' and 'Smoke Gets in Your Eyes - Nat King Cole'. The final result is 'Smoke Gets In Your Eyes - Artie Shaw'.

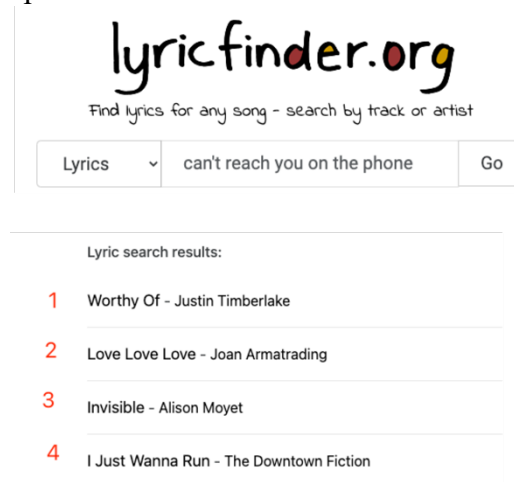
In example 1, query entered is “you are too precious”. There are four duplicate results appeared: two “You Are Too Beautiful” and two “Are You from Dixie”. For example 2,

the query goes as “smoke gets in your eyes” and there are three identical search results “Smoke Gets in Your Eyes”.

The duplicates query results may due to the one-to-many feature of lyric search. In other words, one sketch of lyrics of a song can belong to multiple songs. The current duplicates are probably caused by a lack of filter design that prevents duplicates from appearing in the output.

### 3. Rank Results

Here is an example:



The screenshot shows the lyricfinder.org website. The search bar contains the text "can't reach you on the phone" and the "Go" button is visible. Below the search bar, the results are listed as follows:

Lyric search results:	
1	Worthy Of - Justin Timberlake
2	Love Love Love - Joan Armatrading
3	Invisible - Alison Moyet
4	I Just Wanna Run - The Downtown Fiction

#### First Result:

##### Justin Timberlake - Worthy Of

Yeah, yeah, yeah  
Oh, yeah, yeah

Girl I find you so amazing  
With everyday I learn more about you  
The way you work day out and day in  
And still find time for us too, yeah

I love the fact that you're there for me when I feel down  
No matter what I'm going through  
Lately I've been questioning myself  
Am I good enough for you?

Am I worthy of?  
Am I man enough?  
Am I strong enough?  
To maintain this love  
You've got me questioning (yeah)  
Got me wondering (yeah, yeah)  
Could I be that man?  
That can keep you happy

I know that there'll be times when I'll be miles away  
And you'll be all alone  
I constantly fight my insecurities  
When I **can't reach you** by phone

#### Second Result:

##### Joan Armatrading - Love and Affection

I am not in love  
But I'm open to persuasion  
East or West  
Where's the best  
For romancing

With a friend  
I can smile  
But with a lover  
I could hold my head back  
I could really laugh  
Really laugh

Thank you  
You took me dancing  
'Cross the floor  
Cheek to cheek  
But with a lover  
I could really move  
Really move  
I could really dance  
Really dance  
Really dance  
Really dance  
I could really move  
Really move  
Really move  
Really move

#### Fourth Result:

##### The Downtown Fiction - I Just Wanna Run

I just wanna run  
Hide it away  
Run because they're chasin' me down  
I just wanna run  
Throw it away  
Run before they're findin' me out

I just wanna run!

I just wanna run  
I'm out here all alone  
I tried to call your house  
**Can't reach you on the phone**

The query entered here is *“can’t reach you on the phone”*. The first ranking result has missed by one word: *“by”* instead of *“on”*. The second result has no exact match with any of the query sketches and also missed the key word: *“phone”*. The fourth result has the exact match but are surpassed by the previous ones that have much less relevance comparatively. Therefore, the ranking algorithm of lyricfinder.org also needs further improvement.

### **Solution Proposed:**

In this project, I created a stop word list that included common words used in lyrics with the help of online music statistics. Query terms are normalized, and infrequent terms are given more weight for more precise search and more recalls of results. So, for example, when the query of *“you are my muse”* is entered, there are at least one out of ten results for a song containing the key word *“muse”*.

I also wrote several filters, so duplicates are eliminated or largely reduced while in indexing process and in output rendering. I also added functions in query processing to help limit the one-to-many cases. As hiding duplicates could hurt the recalls, a checkbox is ideal for users to decide whether they would like to suppress duplicate results.

I also calibrated the weight of terms according to the contents of lyrics. The following example can elaborate on this point. The example query is *“you don’t know how to fly until you fall”*. One song has one and only one line that goes as *“you don’t know how to fly”* and the other song contains twelve duplicate lines: *“fly, fly, fly, fly, fly”*. If the search only conducts a simple count of exact words matching, the second song would have higher relevance scores. However, it is obvious that the first song is more likely to be the matching result. This is a very common scenario that needs to be considered in design for lyric search as lyrics are highly prone to repetitions. To deal with this problem, if high scores are resulted from repetitions of same lyrics sketches, the score of the tokens should be significantly lowered.

**Dataset:**

To equip my search engine with lyrics, a popular and resourceful website about lyrics was used as a data source for this project:

<https://www.lyrics.com/>

Lyrics.com is a comprehensive online lyrics resource and currently the largest resource for music, songs, and lyrics online. It is a part of the STANDS4 network that provides online reference and educational resources for over 19 types that include abbreviations, anagrams, biographies and etc.

The header of this website is an alphabetical catalog of artists names. Under each artist is albums that contains songs. Therefore, I implemented a crawler that would first go to artists, then visit each album, and finally fetch the lyrics of songs. The overall crawler follows the Breadth-First-Search algorithm and retrieved all together 20225 songs of lyrics after 2.5 hours of running time.

The lyrics are normalized into tokens after all the non-alphabetical elements are stripped and upper-case words are converted into lower-case. Then the data are organized into python dictionaries with the title of the song as keys and a list of normalized lyric tokens as values. The BeautifulSoup parser facilitates the crawling process by fetching designated data pieces with provided tags. In the end, python Pickle objects are used to store the data into streams that can later be retrieved conveniently by other Python files.

**Code Implementation:**

Here is a brief overview for the tools used in this project to achieve the lyric search function:

- Crawling: BeautifulSoup (Python package for parsing HTML and XML documents)
- Backend: Elasticsearch (search engine based on the Lucene library)
- User Interface: Flask platform (micro web framework with Python)

Since the dataset section above has a detailed description of my crawler, more elaborations will be given to the backend and user interface of my lyric search engine design in the following pages.

### **Back End:**

After conducting researches on the music technology website—Coding in Tune (<https://codingintune.com/2018/04/09/statistics-most-used-words-in-lyrics-by-genre/>), I selected 30 most frequent words in lyrics through a cross-reference comparison over songs of different genres. I then stored the words into the common-words-in-lyrics list that will gain less weights when they appear in the query.

Next, data are retrieved from the pickle object and then are indexed using the Elasticsearch. Therefore, the data set for the backend is established. In processing the queries, after the normalization of words, a flag will be assigned to each token identifying whether they are common words in the list. If any token has a true flag, its weight will be reduced into half after the usual computation.

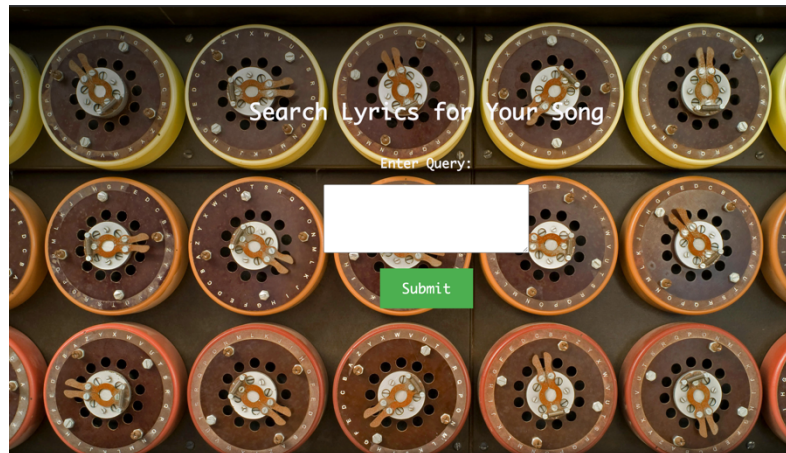
Several filters are implemented in the project to ensure that identical songs are not indexed or appeared in the output for searching results. Also, the weight of token word will be decreased if the frequency of that word exceeds a threshold value to deal with the scenarios mentioned in the solution part.

The Lucene's Practical Scoring Function is the default ranking algorithms of Elasticsearch. It is a similarity model that based on Term Frequency (tf), Inverse Document Frequency (IDF), and the Vector Space Model (vsm). This project will continue using the Lucene score for the ranking function.

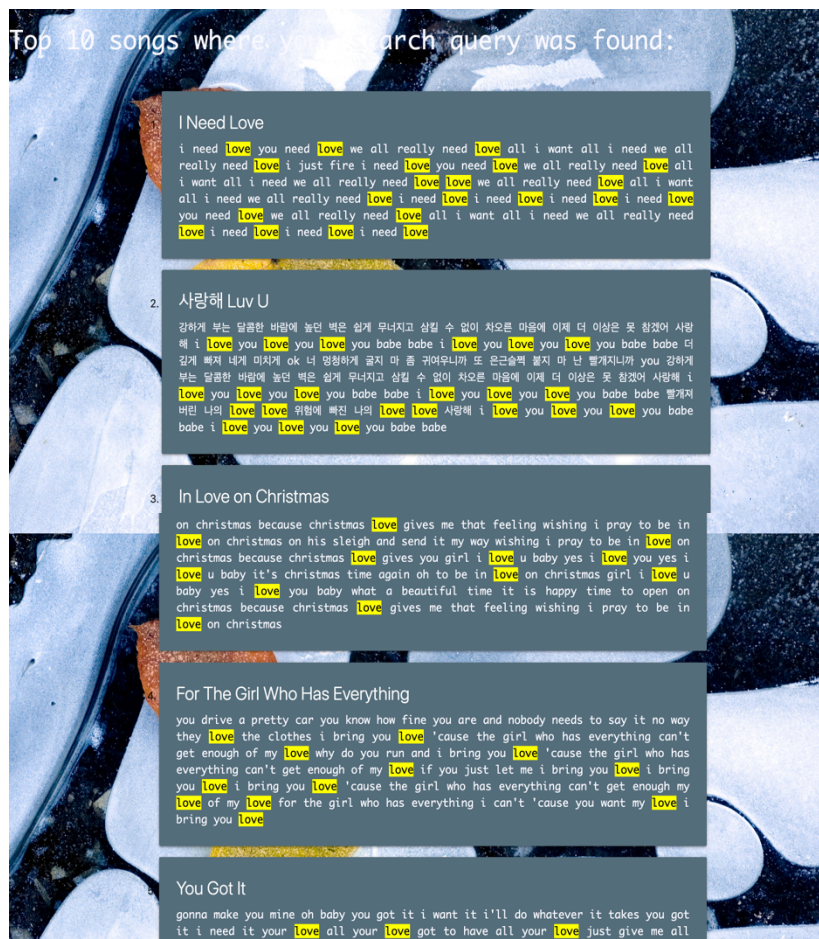
### **User Interface:**



The user interface includes two parts, the search page and the result page. The first page allows users to enter the lyric sketch they would like to search, and the second page include the top ten most relevant songs as the searching results with the query term highlighted in yellow. Overall, the UI is written in JavaScript and CSS languages.



Results for the query: “love”:



**Conclusion:**

The lyric search engine is built to help satisfying the user need of searching lyrics and name of songs with lyric sketches. After identifying rooms for improvement in lyricfinder.org, I designed the searching architecture and implemented a lyric search engine that remedies the identified problems, through making use of the tools, libraries, and the knowledge I acquired from a semester's study in information retrieval. Although the course wraps up with this report, I would continue reinforcing and exploring the intriguing knowledge of information retrieval.