Boolean Search and Evaluation using Precision & Recall

Elyesa Duru

March 2025

1 Introduction

Natural Language Processing (NLP) and Information Retrieval (IR) are crucial in modern technology. In this report, I explore a simple Boolean search implementation and evaluate its effectiveness using precision and recall. The programming language used for this project is Python, due to its efficiency and vast ecosystem of libraries for text processing.

2 Dataset

The dataset consists of the top 100 Billboard songs (US) as of March 24, 2025. The lyrics were obtained from Lyricsify and stored as ‘.txt‘ files. Each file includes metadata such as the song title, artist, and lyrics.

3 Boolean Search Implementation

The Boolean search allows the use of the logical operators AND, OR, and NOT to filter documents based on keyword presence. The search is currently does not yet fully support mixed Boolean queries (e.g., ‘”love AND (happy OR sad)”‘).

4 Search Queries

To analyze the performance of the Boolean search, I tested the following queries:

1. **beat (Song beat vs. beating someone)**
2. **like you AND love (searching for happy love songs)**
3. **leave me NOT love (searchign for sad love songs)**
4. **locked (locked in mentally vs. locked in Jail)**
5. **looking AND eyes (looking into someones eyes)**

5 Evaluation: Precision and Recall

To evaluate the Boolean search, I measured precision and recall for each query.

- **Precision** $= \frac{\text{Relevant Retrieved Documents}}{\text{Total Retrieved Documents}}$
- **Recall** $= \frac{\text{Relevant Retrieved Documents}}{\text{Total Relevant Documents}}$

5.1 Results for Each Query

Each query's results were manually evaluated by reviewing all the songs and tagging them with descriptors such as "love," "happy," "sad," "English," "rap," etc. Additionally, for queries like "love you," I also searched for synonyms like "adore" and "cherish you" to capture all contextually relevant songs that might not have been retrieved by the Boolean search. The table below summarizes the precision and recall findings for each query:

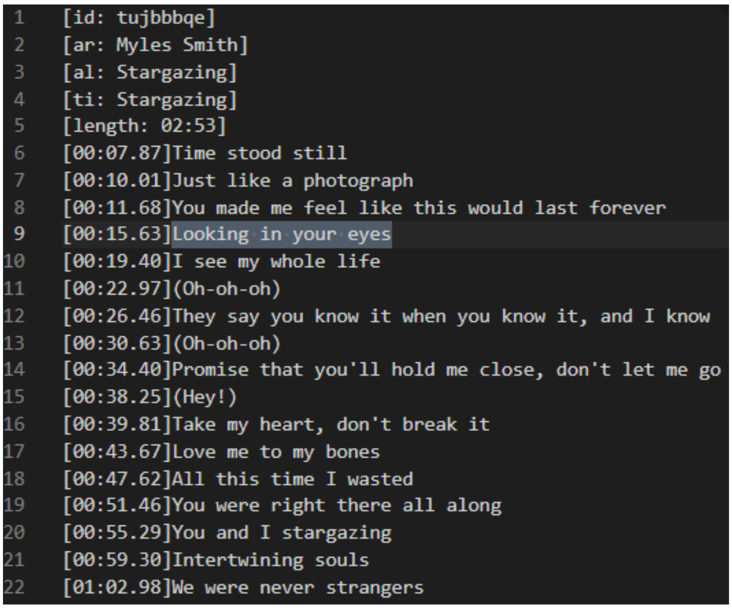
Query	**Precision**	**Recall**
looking AND eyes	1.00	0.40
beat	0.27	1.00
like you AND love	0.00	0
leave me NOT love	1.00	0.105
locked	0.50	1.00

Table 1: Precision and Recall for the selected queries.

5.2 Detailed Query Descriptions

Below is a brief explanation of what each query aimed to find:

- **looking AND eyes:** Here the query aimed to search for someone looking into someones eyes like in the picture below. The query returned 2 songs of wich booth where relevant.



```
[id: tujbbbqe]
[ar: Myles Smith]
[al: Stargazing]
[ti: Stargazing]
[length: 02:53]
[00:07.87]Time stood still
[00:10.01]Just like a photograph
[00:11.68]You made me feel like this would last forever
[00:15.63]Looking in your eyes
[00:19.40]I see my whole life
[00:22.97](Oh-oh-oh)
[00:26.46]They say you know it when you know it, and I know
[00:30.63](Oh-oh-oh)
[00:34.40]Promise that you'll hold me close, don't let me go
[00:38.25](Hey!)
[00:39.81]Take my heart, don't break it
[00:43.67]Love me to my bones
[00:47.62]All this time I wasted
[00:51.46]You were right there all along
[00:55.29]You and I stargazing
[00:59.30]Intertwining souls
[01:02.98]We were never strangers
```

Figure 1: Looking in your eyes

- **beat:** The query retrieved 11 results, but only 3 were relevant to my intended meaning. The word "beat" often appeared in different contexts one instance referred to "beating" someone, while four occurrences were simply part of the phrase "XYZ on the beat" in song intros, making them irrelevant.

- **like you AND love:** The query retrieved 4 results, but none matched the intended search for happy love songs. Instead, all retrieved songs had a sad or negative tone, showing that the Boolean search did not effectively capture the emotional context of the lyrics. One song as example had these lyrics **But ain't no love, shot him in his heart, ain't have no chance**

- **leave me NOT love:** The query retrieved 2 results, both of which matched my intention. The results were about sadness and loss

- **locked:** The query retrieved 2 results, but only one matched my intended meaning of being "locked in mentally." The other result referred to a different context, showing a limitation in distinguishing between metaphorical and literal uses of the word.

For "like you AND love" and "leave me NOT love," I searched all songs where "love" appeared and divided them into three categories: sad love, happy love, or neither, so I could calculate the recall. Out of the 44 matches with the keyword "love," I found 19 sad love songs, 8 happy love songs, and 17 that could not be placed in either category.

6 Conclusion and Future Improvements

The Boolean search provided a basic but effective way to retrieve documents. However, its limitations include:

- Lack of ranking (all matching results are treated equally).
- Poor handling of context (e.g. "love").
- Problems with synonyms (e.g. "beat").
- Limited precision when searching for common words.

Final Thoughts and Experience: The Boolean search was a great starting point for real-world applications, but it lacks a lot of context, making it difficult to find all the relevant texts that were not retrieved. I feel like people with large collections of texts will either have to manually sift through and identify relevant information or improve their basic Boolean search with some external help. Otherwise, they would need to proofread everything to ensure they didn't miss anything. It was also very challenging to come up with queries for songs, since the sentences are very short and lyrics often function more as artistic expressions, where the overall context, as well as the beat and instruments, play a crucial role. I think searching within larger, simpler texts might be slightly easier, but Boolean search would still lack context, and there is no reliable way to identify all relevant texts that were not retrieved. Adding to all of that, it is very important to choose good and fitting keywords in the Boolean search, as you can see in the love example with which I demonstrated it.

Proof: I have some proof on the proofs-file including pictures as well as a video on my GitHub.