WebIR - Project: Lyrics Search Engine

AURÉLIEN VU NGOC *2020280219*

1 INTRODUCTION

In recent years, the music industry has been flourishing. Artists from all over the world and from various horizons have participated in creating new amazing content. To support the ever growing international musical content, many online services have emerged to the extent that they are now part of our daily life. While streaming plateforms offer an almost unlimited access to musical resources, song lovers often face the problem that they are only given access to the music, not other information. Among other crucial information about songs, lyrics play an important role and should be better featured within these services. Therefore, we introduce *ILYrics*, a minimal search engine for lyrics that works in both directions: users can find their favorite song lyrics using its information or find the song title based on part of lyrics.

The objective of this project is to tackle this issue and try to implement our own lyrics service in form of a search engine. Key challenges in achieving our goal are: (i) **collect data**, because most existing websites are "biased" and only provide lyrics for a specific genre whereas we want our range of action to be as wide as possible, (ii) **index a large amount of data**, because we need fast access to such resources while being able to process a lot of songs at once, (iii) **handle diversity**, especially being able to process various languages, because song music can come from anywhere in the world, (iv) **provide high availability**, because a search engine should be 24/7 available to anyone who wants to use it.

The present report explains how we address these challenges, it gives insight about our methodology and introduces multiple experiments showcasing our project features. It has the following structure: section 2 anchors the context and recalls existing work related to our topic, section 3 demonstrates an overview of our system, section 4 reveals technical details about the inner workings of our search engine, section 5 offers an experimental analysis, and finally, section 6 casts light on new ideas and possible ways to improve our system.

2 RELATED WORK

2.1 Industry

Industry-wise, it is clear that our idea is not a novelty and many systems already exist in the market. First of all, Genius[1] collects music knowledge and song lyrics from international artists. It relies on the community to enrich musical knowledge with annotations, interpretations and stories. They provide a simple API to make requests to their servers about songs, artists, lyrics and basically anything related to music. The Genius website is our main inspiration, and also one of the sources we employ to gather information (see section 4).
Next, online collection of song lyrics such as AZlyrics[2] or Lyrics[3] give access to millions of songs. However, they often work in a one-way fashion: users can only ask for lyrics based on a song title or an artist name, whereas we also want to be able to provide results based on part of the lyrics themselves. Interestingly, either their search engine does not allow such queries, or it often fails at giving good results. Furthermore, they only support English songs.
Evidently, major search engine that serves general purposes such as Google or Yahoo also provide

[1]https://www.genius.com
[2]https://www.azlyrics.com
[3]https://www.lyrics.com

such services. Users can simply ask their favorite seach engine queries like "*La vie en rose lyrics*" to get the lyrics they want. However since these systems are not specific to music, they often fail to produce good results when songs are less popular.
Finally other applications such as Shazam[4] also complete the mission but with a different approach: queries are made by directly listening to the song. Comparatively, our objective is to make text queries rather than audio queries.

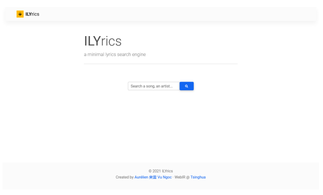
Given how big the internet is, especially when it comes to music, we are fully aware that the previous list is by no mean exhaustive and that there are high chances we miss other niche related services. However the cited resources are the most important services and altogether make a great part of what can be found in the music industry when it comes to lyrics. We definitely get inspired by many of them.

2.2 Research

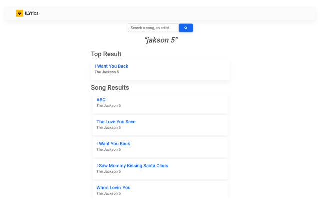
Research-wise, the project deals with text document search. Numerous techniques have been developped over the years to overcome this challenge, noticeably ranking methods to better reach the most accurate results for a given query. Text-based queries are well studied: initially, boolean models [Jones 1972] were designed to assess document relevance, later on, vector based models [Salton et al. 1975] were in vogue to rank documents, then probabilistic models took over mostly using TF-IDF ranking [Salton and Buckley 1987] and leading to the well-known BM-25 scoring function [Robertson et al. 1995]. More recently, search engines took advantage of Natural Language Understanding (NLU) progress, notably language models, to upgrade their approach by having a better comprehension of the query itself [Trotman et al. 2014]. Nonetheless, query understanding is a wide area of research with many directions remaining for exploration [Carmel et al. 2020]. As far as we are concerned, query understanding is a difficult barrier to hurdle in a musical query scenario because search terms often do not form a sentence and do not even use words from the English dictionnary (because of artists' names, song titles, . . .).

3 SYSTEM DESIGN OVERVIEW

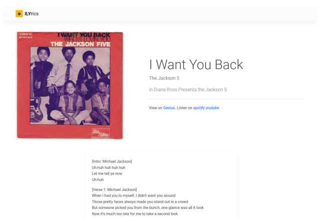
The system we design, we call it *ILYrics*, helps user find song lyrics based on a query by artist name, song title or even part of lyrics. It is composed of 3 main sections, illustrated in Figure 1: 1a the search page, 1b the results page and 1c the song page. All three pages can be accessed using the url https://ilyrics.herokuapp.com.


ILYrics
ILYrics
a minimal lyrics search engine
Search a song, an artist...
© 2021 ILYrics
Created by Aurélien 柴靈 Vu Ngoc · WebIR @ Tsinghua

(a) Search page



(b) Results page


ILYrics
I Want You Back
The Jackson 5
in Diana Ross Presents the Jackson 5
View on Genius. Listen on spotify youtube

[Intro: Michael Jackson]
Uh-huh huh huh huh
Let me tell ya now
Uh-huh

[Verse 1: Michael Jackson]
When I had you to myself, I didn't want you around
Those pretty faces always made you stand out in a crowd
But someone picked you from the bunch, one glance was all it took
Now it's much too late for me to take a second look

(c) Song page

Fig. 1. System overview: the 3 main pages

The search page features a search bar, as in any search engine, to allow users to make their queries. Once the query is formulated, the user is directed to the results organised in 3 different

[4]https://www.shazam.com

levels: *Top Results*, *Song Results* and *Lyric Matches*, each having its unique search method (more about how they respectively work in section 4). If the query is successful and the user find what they were looking for, they can click a specific result to see more information about the song including: the album, the producer, the cover, links to listen to the song, . . . and most importantly the lyrics. This minimal design with only 3 pages is a perfect fit for this project because our main concern is to produce a functional service rather than a complete service, hence the lack of advanced features. The system has plenty of room for improvements in terms of design and functionnalities which are discussed in section 6.

It consists of different components each interacting with one another as depicted in Figure 2.

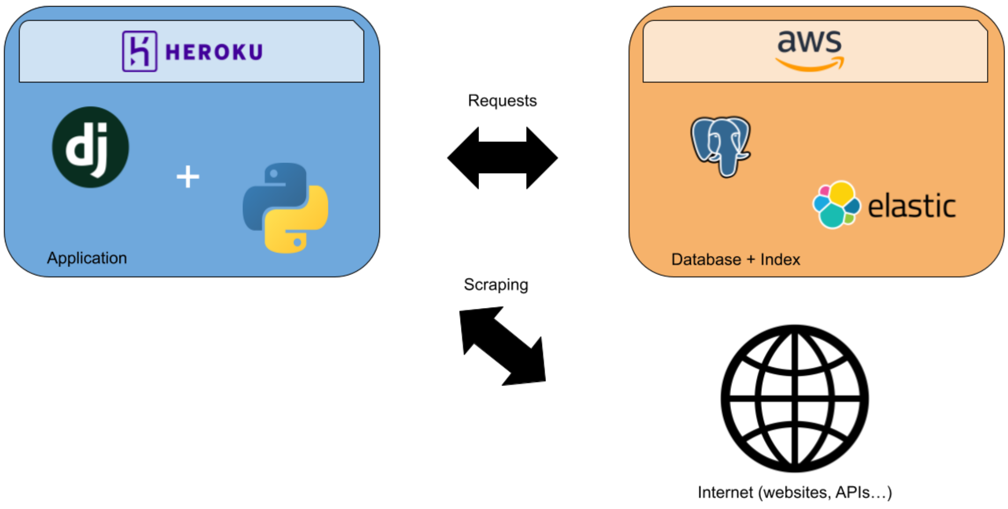


Fig. 2. Multiple components interacting with each other

In general, since our system is meant to be available online, we decide to create a web application. Running online on a remote server, the software can be accessed from virtually anywhere on the planet to make queries and get information fast. The web application is a support for the search engine we implement, relying on multiple tools that work together, each solving its own dedicated problem. **Heroku**[5] helps building such system by allowing quick deployment of data-driven application. They make sure our search engine is available, secure, easily managed and automatically upgraded following code updates. Even though building our own server from scratch at home could give us full control of the system, it would come with numerous issues such as software compatibility, environment isolation, hardware dedication and more. Simplicity and availability are thus the main reasons for us choosing Heroku.

First, its foundation relies on **Python**, as it is a very simple yet powerful programming language, and also because we are most familiar with it. More specifically, we use a Python-based free and open-source web framework, **Django**[6]. The framework follows the model–template–views architectural pattern and offers a lot of side tools allowing easy integration of external software and services. Being completely new to web application development, we decide to turn towards

[5]https://www.heroku.com
[6]https://www.djangoproject.com

Django instead of its more tunable counterpart Flask[7] because their architecture is simpler. While Flask is extremely available and adaptive to one's situation, Django encourages quick and clean development by taking care of much of the problems of Web development under the hood, and still being highly scalable. Simplicity and efficiency are thus the main reasons for us choosing Django.

Next, data storage is taken care of by an Amazon Web Service (AWS)[8] instance hosting a **PostgreSQL database**[9]. Django applications come with an implicit integration of PostgreSQL database so that the only thing to do is connect to the AWS instance using Amazon secure connection service. The advantage of using a remote database, compared to a local database, is its high availability. AWS instances also provide a Free Tier plan including a large storage capacity (up to 20Gb), a decent query processing ability and numerous ways to monitor its activity. Although it requires a lot more installation steps than just running our own database, the resulting database has high performance which allow us to concentrate more on the application rather than the services it runs on. Power and accessibility are thus the main reasons for us choosing AWS PostgreSQL database.

Finally, as any search engine would normally require, indexing the documents is achieved by **ElasticSearch**[10]. They help us search gigabytes of data in seconds while being extremely simple to integrate into our Django application. The service is also hosted within the Free Tier AWS instance, so that both the database and the index can communicate very easily. Speed and scalability are thus the main reasons for us choosing ElasticSearch.

4 TECHNICAL DETAILS

4.1 Data

Structure. Following the model–template–views architecture of Django, we create a `Song` model. Each song is composed of 3 text fields: a title, an artist and lyrics. As a result, all songs information are stored within the database, alongside some more features such as the album cover image, links to streaming plateforms where the user can listen to the song, and a list of all participating artists. All this additional information is displayed when browsing the song page. Of course this pattern is minimal but serves best the purpose of our application. More complicated and intricated structures involving an `Artist` model are conceivable, it would add a lot of features and possibilities but require careful manipulation, and are not the point of this project. We choose simplicity over abundance of features in order to focus on more important challenges, but our system can still be improved and scaled for production with advanced features that are discussed in section 6.

Source. Data comes from various sources. Since our main goal is to maintain an index of songs from all over the world, we have to multiply our sources so that diversity is preserved. First, online datasets are very useful to that effect, although they often need manual preprocessing to fit Django `Song` model requirements. After the preprocessing step, we simply add all songs to the index and database using a bulk add function we designed ourselves.

Second, online scraping. Datasets are great and offer a lot of data at once, but they are static, i.e. they are not updated as new songs are coming out. On the other hand, websites offering song lyrics are often frequently filled with new content and appears to be very up-to-date with respect to the music industry. We gather information from these websites as needed using either their build-in API if they have any, or otherwise using manual scraping.

[7]https://flask.palletsprojects.com/en/2.0.x/
[8]https://aws.amazon.com
[9]https://www.postgresql.org
[10]https://www.elastic.co

4.2 Index

Song indexing is processed either when collecting data from the previously mentionned sources, or live at query execution time. Thanks to the toolbox provided by Django, updating the index is relatively easy and can be done in no more than 2 lines of code within seconds. We need to constantly update the index because we are starting from zero and have not much indexed content, that is why we decide to implement live indexing. Each query issued by an user is send to the Genius API which responds with relevant songs that are immediately added to the index. Thus we continuously upgrade the capacity of our system to answer the users' queries.

As for the index itself, we use the 3 fields: title, artist and lyrics as entries. A unique song is defined by a combination of title + artist. These features are the only one we use at query execution time using ElasticSearch.

4.3 Query processing

Parsing. Although we tried a lot of combination for parsing and had few ideas, we faced multiple issues that prevent us from having an advanced parsing pipeline. Therefore it is only minimal in this project: lower, strip and split are the only function we apply to the query to make it ElasticSearch-friendly. We first thought of an spell check system to auto-correct typos and erroneous words but since the targets are often music names, i.e. proper nouns and invented names (of artists for instance), any spell-check model would definitely not fit. Besides one of our challenges is to make the system international, i.e. it must work for any input language, but spell-check systems often does not handle multiple languages as input. As a result, the only solution was to develop our own system, using a custom multi-lingual pre-trained language model, which would be computationally too expensive to run.

Moreover, we thought of adding a POS tagging system to better understand queries, but again, queries related to music are too specific and are often not even sentences. The idea was to dissociate an artist name from a song title (in the case of a query such as "*Michael Jackson Billie Jean*") to have more precise results, but the task is too difficult as only general knowledge would help. Queries are often extremely short, albeit very diverse, allowing no context at all for language models to understand its meaning. That is why parsing in this our application is only minimal.

Query. The application features 3 levels of query: *Top Results*, *Song Results* and *Lyric Matches*. Each using its own set of features: while *Lyric Matches* only searches within the `Song.lyrics` field, *Song Results* looks for matches in the `Song.title` and `Song.artist` fields, whereas the *Top Results* uses all indexed fields. Results are thus more accurate, because queries are more precise when looking only in a specific field, which helps the user get information faster.

Moreover we add a 'hightlight' feature using Django `query()` system and the `fragment_size` parameter. This greatly helps visibility and explainability of why such results pop up so high on the list.

4.4 Ranking

Ranking is a very, if not the most, important aspect of our system. ElasticSearch provides a natural access to a custom BM-25 score based on the Lucene implementation [Kamphuis et al. 2020]. It is a combination of Boolean models, TF-IDF processing and Vector Space models. The complete scoring function is given by Equation 4.4[11].

$$score(q,d) = queryNorm(q) \cdot coord(q,d) \cdot \sum_{t \in q} \left[tf(t \in d) \cdot idf(d)^2 \cdot t.getBoost() \cdot norm(t,d) \right] \quad (1)$$

[11] see https://www.elastic.co/guide/en/elasticsearch/guide/current/practical-scoring-function.html

New	Indexed	Cached
5.23 sec	3.31 sec	1.78 sec

Table 1. Response time in different scenarios (average)

The relevance score of a document d for a query q is given by $score(q, d)$ which is composed of: a query normalization factor $queryNorm$ which makes results of a given query comparable to another's results, a coordination factor $coord$ that helps documents that matches multiple terms in the query appear higher, a term frequency term tf, an inverse document frequency term idf, a field boost $getBoost$ defined before execution by the user to give more value to certain fields (for example `Song.title` over `Song.lyrics`) and a fiel-length normalization $norm$ to prevent long documents from being ranked too high. No changes are operated on this initial scoring function as it yielded very good performance.

5 EXPERIMENTAL RESULTS

By means of examining our system performance, we design experiments to evaluate the accuracy on the one hand, and on the other response time performance.

5.1 Accuracy

Although there is no universal method for evaluating the accuracy of an information retrieval system, we wish to provide insights about the capacities of our search engine. We develop 4 different query templates that are most likely to be used in our system: (i) a song title, (ii) an artist name, (iii) a song title and an artist name, and finally (iv) part of lyrics. For experimental purposes we target a single song: Michael Jackson's *Billie Jean* to produce comparable results. In Appendix A, we demonstrate the outcome of this experiment, proving that our system can take as input very different queries while producing decent results.

Moreover it is important to notice, for instance in Figure 1b, that the system can overcome simple typos and still give results.

5.2 Response Time

Another crucial issue for search engines is their ability to answer queries fast. To acknowledge the timing performance of our system, we need to separate contexts: when the system has to search a brand new song, when the sought song is already indexed and when results are already cached from a previous query. For each situation, we evaluate 20 queries (60 queries in total) and measure the response time after a reset of the search engine state made according to the scenario. Cache is emptied between each request in the first two scenarios and automatic indexing is disabled in the first scenario. Table 1 demonstrates the average response time for each situation. Although results can vary relative to the internet connection, our system performs decently on average but is still way slower than professional search engines such as Google. It is clear cached information largely speeds up the process and that live scraping largely slows down the response time. However there is a compromise to achieve between accuracy and response time, given that the hardware we run the server on has limited performance.

6 CONCLUSION & FUTURE WORK

Overall, the search engine we propose offers decent performance in both accuracy and response time, we are satisfied with the results. We solved the challenges that we defined when starting the project. On the other hand, there is still plenty of room for improvements in different aspects.

First, **scoring** can be enhanced by using perhaps more suited ranking function than BM25. Experiments in this direction can reveal dramatic improvements in the system accuracy. Then, **query understanding** is a very hot topic within the research community and recent progress can definitely be used in our particular application. Language models have seen tremendous advances lately allowing the comprehension of most user queries. Integrating LM knowldege in our system can help develop more features such as a searches by album where the user only describes what the cover looks like in plain text. Additionnally, **parsing** would complement the improvements achieved through query understanding. We wish to create a more advanced parsing pipeline, maybe with general knowledge, in order to help the system focus on what is important. Of course, **scale** is also a concern: having already set up strong and scalable components, it is necessary to increase the number of indexed songs and artists. Scaling can also come from enhancing the model architecture by adding new complex models such as `Artist`, `Album`, `Genre` and more. These new models would contain much more information, helping with indexing as well as giving users the opportunity to learn more about the song they looked for. Along with new models, new pages can be implemented to display these new details. Finally, increased **security** is to be sought because such systems are vulnerable to malicious persons. Even though it does not contain any sensitive information at the moment, such security update would set the scene for, say, integration of user accounts for personnalized recommendations.

In this report we presented *ILYrics*, a minimal search engine developped to find song lyrics. Users are able to query by title, name or even part of lyrics to find the desired song within seconds. We gave insights about the system architecture, its numerous components, its inner structure, and gave detailed explanations about how it actually works. After experimental work, we concluded that our system yields decent performance that matches the initial specification. Lastly we took a step back to look at the bigger picture and listed some ideas to improve the system.

All in all, this project gave us the opportunity to get familiar with a lot of new technologies. We discovered various new systems, frameworks, techniques, . . . that we are sure will be extremely helpful to us in the near future. We definitely learned a lot and hope this report will give enough insights to reflect the work we put into this project.

REFERENCES

David Carmel, Yi Chang, Hongbo Deng, and Jian-Yun Nie. 2020. Future Directions of Query Understanding. In *Query Understanding for Search Engines*, Yi Chang and Hongbo Deng (Eds.). Springer International Publishing, Cham, 205–224. https://doi.org/10.1007/978-3-030-58334-7_9

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* (1972). Publisher: MCB UP Ltd.

Chris Kamphuis, Arjen P. de Vries, Leonid Boytsov, and Jimmy Lin. 2020. Which BM25 Do You Mean? A Large-Scale Reproducibility Study of Scoring Variants. In *Advances in Information Retrieval (Lecture Notes in Computer Science)*, Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins (Eds.). Springer International Publishing, Cham, 28–34. https://doi.org/10.1007/978-3-030-45442-5_4

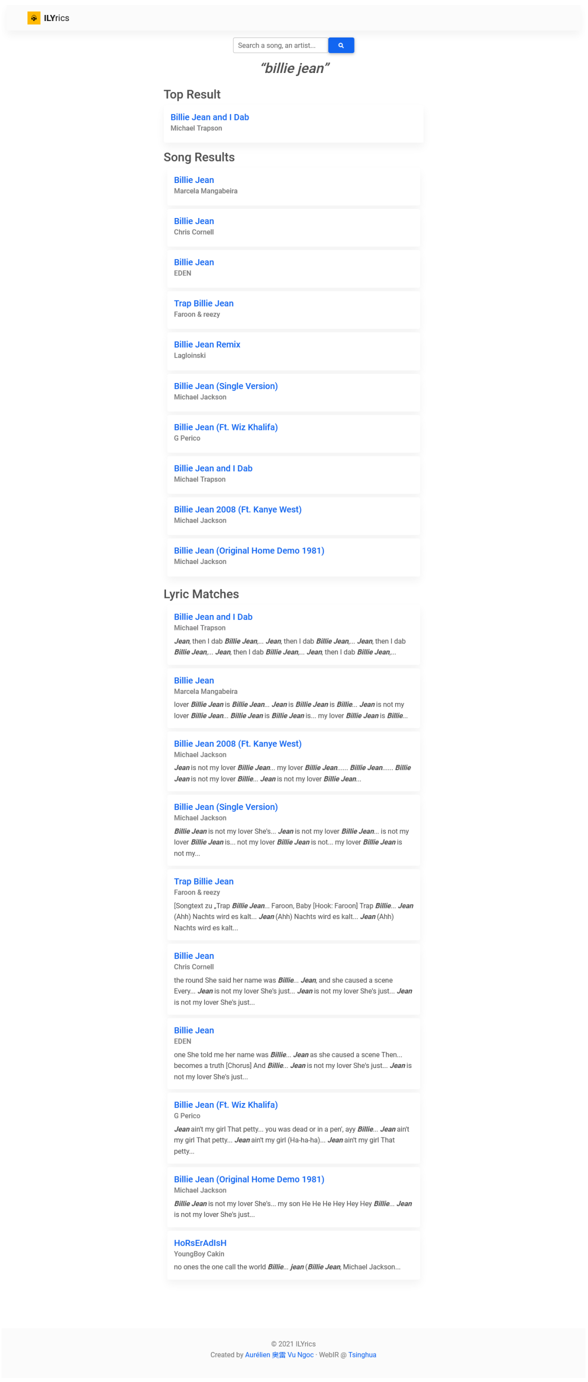
Stephen E. Robertson, Steve Walker, Susan Jones, Micheline M. Hancock-Beaulieu, and Mike Gatford. 1995. Okapi at TREC-3. *Nist Special Publication Sp* 109 (1995), 109. Publisher: NATIONAL INSTIUTE OF STANDARDS & TECHNOLOGY.

Gerard Salton and Chris Buckley. 1987. Term Weighting Approaches in Automatic Text Retrieval. (Nov. 1987). https://ecommons.cornell.edu/handle/1813/6721 Accepted: 2007-04-23T17:23:03Z Publisher: Cornell University.

G. Salton, A. Wong, and C. S. Yang. 1975. A vector space model for automatic indexing. *Commun. ACM* 18, 11 (Nov. 1975), 613–620. https://doi.org/10.1145/361219.361220

Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to BM25 and Language Models Examined. In *Proceedings of the 2014 Australasian Document Computing Symposium (ADCS '14)*. Association for Computing Machinery, New York, NY, USA, 58–65. https://doi.org/10.1145/2682862.2682863

A EXPERIMENTAL RESULTS



ILYrics

Search a song, an artist...

"billie jean"

Top Result

Billie Jean and I Dab
Michael Trapson

Song Results

Billie Jean
Marcela Mangabeira

Billie Jean
Chris Cornell

Billie Jean
EDEN

Trap Billie Jean
Faroon & reezy

Billie Jean Remix
Lagloinski

Billie Jean (Single Version)
Michael Jackson

Billie Jean (Ft. Wiz Khalifa)
G Perico

Billie Jean and I Dab
Michael Trapson

Billie Jean 2008 (Ft. Kanye West)
Michael Jackson

Billie Jean (Original Home Demo 1981)
Michael Jackson

Lyric Matches

Billie Jean and I Dab
Michael Trapson
Jean, then I dab ***Billie Jean***,... ***Jean***, then I dab ***Billie Jean***,... ***Jean***, then I dab ***Billie Jean***,... ***Jean***, then I dab ***Billie Jean***,... ***Jean***, then I dab ***Billie Jean***,...

Billie Jean
Marcela Mangabeira
lover ***Billie Jean*** is ***Billie Jean***... ***Jean*** is ***Billie Jean*** is ***Billie***... ***Jean*** is not my lover ***Billie Jean***... ***Billie Jean*** is ***Billie Jean*** is... my lover ***Billie Jean*** is ***Billie***...

Billie Jean 2008 (Ft. Kanye West)
Michael Jackson
Jean is not my lover ***Billie Jean***... my lover ***Billie Jean***...... ***Billie Jean***...... ***Billie Jean*** is not my lover ***Billie***... ***Jean*** is not my lover ***Billie Jean***...

Billie Jean (Single Version)
Michael Jackson
Billie Jean is not my lover She's... ***Jean*** is not my lover ***Billie Jean***... is not my lover ***Billie Jean*** is... not my lover ***Billie Jean*** is not... my lover ***Billie Jean*** is not my...

Trap Billie Jean
Faroon & reezy
[Songtext zu „Trap ***Billie Jean***... Faroon, Baby [Hook: Faroon] Trap ***Billie***... ***Jean*** (Ahh) Nachts wird es kalt... ***Jean*** (Ahh) Nachts wird es kalt... ***Jean*** (Ahh) Nachts wird es kalt...

Billie Jean
Chris Cornell
the round She said her name was ***Billie***... ***Jean***, and she caused a scene Every... ***Jean*** is not my lover She's just... ***Jean*** is not my lover She's just... ***Jean*** is not my lover She's just...

Billie Jean
EDEN
one She told me her name was ***Billie***... ***Jean*** as she caused a scene Then... becomes a truth [Chorus] And ***Billie***... ***Jean*** is not my lover She's just... ***Jean*** is not my lover She's just...

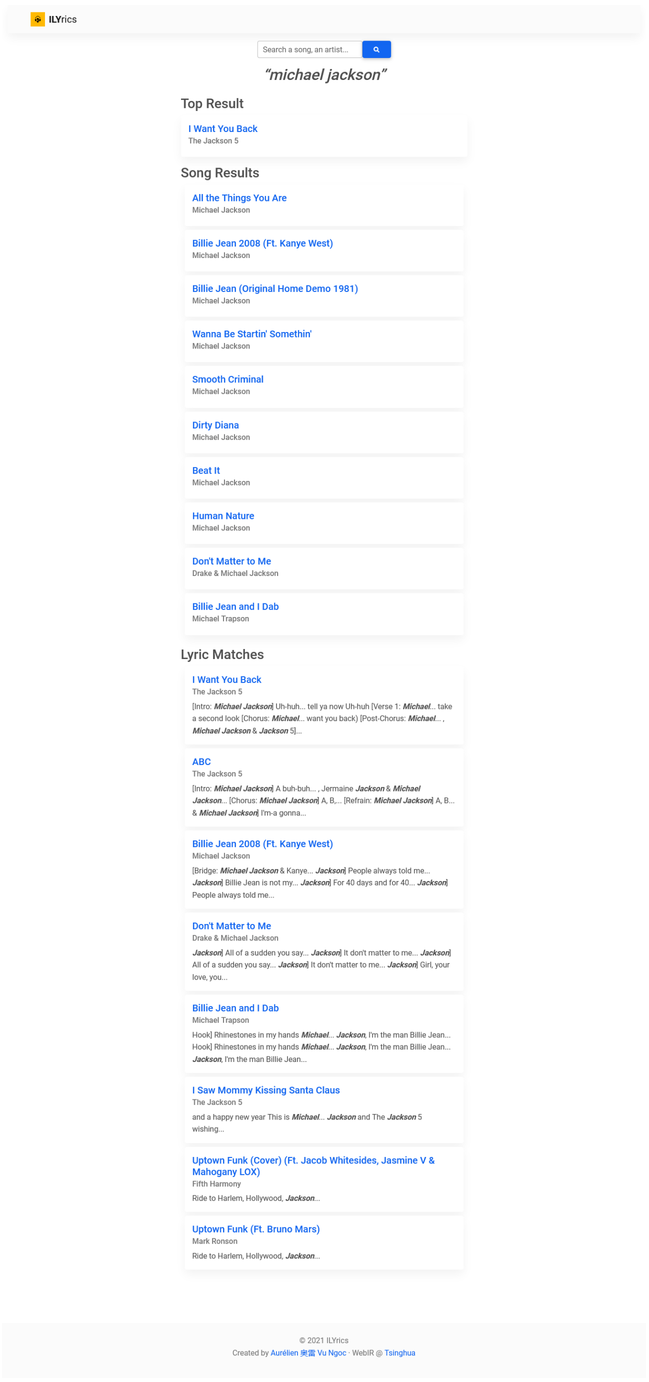
Billie Jean (Ft. Wiz Khalifa)
G Perico
Jean ain't my girl That petty... you was dead or in a pen', ayy ***Billie***... ***Jean*** ain't my girl That petty... ***Jean*** ain't my girl (Ha-ha-ha)... ***Jean*** ain't my girl That petty...

Billie Jean (Original Home Demo 1981)
Michael Jackson
Billie Jean is not my lover She's... my son He He He Hey Hey Hey ***Billie***... ***Jean*** is not my lover She's just...

HoRsErAdIsH
YoungBoy Cakin
no ones the one call the world ***Billie***... ***jean*** (***Billie Jean***, Michael Jackson...

© 2021 ILYrics
Created by Aurélien 奥雷 Vu Ngoc · WebIR @ Tsinghua

Fig. 3. Results using only the song title



ILYrics

Search a song, an artist...

"michael jackson"

Top Result

I Want You Back
The Jackson 5

Song Results

All the Things You Are
Michael Jackson

Billie Jean 2008 (Ft. Kanye West)
Michael Jackson

Billie Jean (Original Home Demo 1981)
Michael Jackson

Wanna Be Startin' Somethin'
Michael Jackson

Smooth Criminal
Michael Jackson

Dirty Diana
Michael Jackson

Beat It
Michael Jackson

Human Nature
Michael Jackson

Don't Matter to Me
Drake & Michael Jackson

Billie Jean and I Dab
Michael Trapson

Lyric Matches

I Want You Back
The Jackson 5
[Intro: ***Michael Jackson***] Uh-huh... tell ya now Uh-huh [Verse 1: ***Michael***... take a second look [Chorus: ***Michael***... want you back) [Post-Chorus: ***Michael***... , ***Michael Jackson*** & ***Jackson*** 5]...

ABC
The Jackson 5
[Intro: ***Michael Jackson***] A buh-buh... , Jermaine ***Jackson*** & ***Michael Jackson***... [Chorus: ***Michael Jackson***] A, B,... [Refrain: ***Michael Jackson***] A, B... & ***Michael Jackson***] I'm-a gonna...

Billie Jean 2008 (Ft. Kanye West)
Michael Jackson
[Bridge: ***Michael Jackson*** & Kanye... ***Jackson***] People always told me... ***Jackson***] Billie Jean is not my... ***Jackson***] For 40 days and for 40... ***Jackson***] People always told me...

Don't Matter to Me
Drake & Michael Jackson
Jackson] All of a sudden you say... ***Jackson***] It don't matter to me... ***Jackson***] All of a sudden you say... ***Jackson***] It don't matter to me... ***Jackson***] Girl, your love, you...

Billie Jean and I Dab
Michael Trapson
Hook] Rhinestones in my hands ***Michael***... ***Jackson***, I'm the man Billie Jean... Hook] Rhinestones in my hands ***Michael***... ***Jackson***, I'm the man Billie Jean... ***Jackson***, I'm the man Billie Jean...

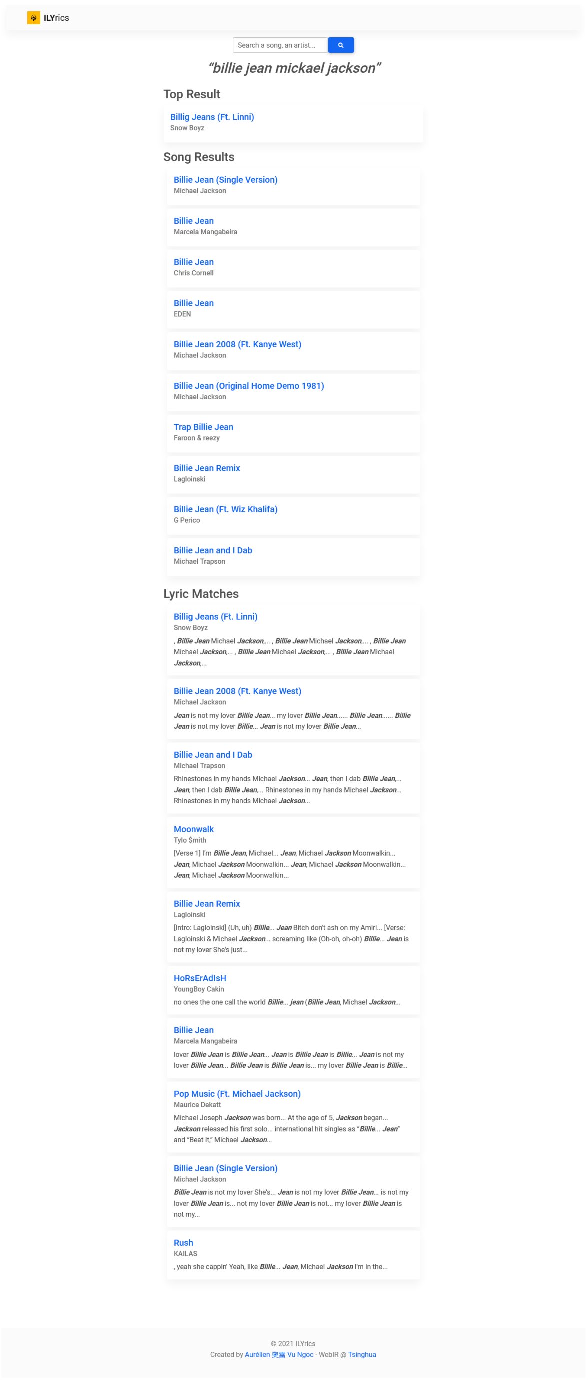
I Saw Mommy Kissing Santa Claus
The Jackson 5
and a happy new year This is ***Michael***... ***Jackson*** and The ***Jackson*** 5 wishing...

Uptown Funk (Cover) (Ft. Jacob Whitesides, Jasmine V & Mahogany LOX)
Fifth Harmony
Ride to Harlem, Hollywood, ***Jackson***...

Uptown Funk (Ft. Bruno Mars)
Mark Ronson
Ride to Harlem, Hollywood, ***Jackson***...

© 2021 ILYrics
Created by Aurélien 奥雷 Vu Ngoc · WebIR @ Tsinghua

Fig. 4. Results using only the artist name


ILYrics

Search a song, an artist...

"billie jean mickael jackson"

Top Result

Billig Jeans (Ft. Linni)
Snow Boyz

Song Results

Billie Jean (Single Version)
Michael Jackson

Billie Jean
Marcela Mangabeira

Billie Jean
Chris Cornell

Billie Jean
EDEN

Billie Jean 2008 (Ft. Kanye West)
Michael Jackson

Billie Jean (Original Home Demo 1981)
Michael Jackson

Trap Billie Jean
Faroon & reezy

Billie Jean Remix
Lagloinski

Billie Jean (Ft. Wiz Khalifa)
G Perico

Billie Jean and I Dab
Michael Trapson

Lyric Matches

Billig Jeans (Ft. Linni)
Snow Boyz
, ***Billie Jean*** Michael ***Jackson***,... , ***Billie Jean*** Michael ***Jackson***,... , ***Billie Jean*** Michael ***Jackson***,... , ***Billie Jean*** Michael ***Jackson***,... , ***Billie Jean*** Michael ***Jackson***,...

Billie Jean 2008 (Ft. Kanye West)
Michael Jackson
Jean is not my lover ***Billie Jean***... my lover ***Billie Jean***...... ***Billie Jean***...... ***Billie Jean*** is not my lover ***Billie***... ***Jean*** is not my lover ***Billie Jean***...

Billie Jean and I Dab
Michael Trapson
Rhinestones in my hands Michael ***Jackson***... ***Jean***, then I dab ***Billie Jean***,... ***Jean***, then I dab ***Billie Jean***,... Rhinestones in my hands Michael ***Jackson***... Rhinestones in my hands Michael ***Jackson***...

Moonwalk
Tylo $mith
[Verse 1] I'm ***Billie Jean***, Michael... ***Jean***, Michael ***Jackson*** Moonwalkin... ***Jean***, Michael ***Jackson*** Moonwalkin... ***Jean***, Michael ***Jackson*** Moonwalkin... ***Jean***, Michael ***Jackson*** Moonwalkin...

Billie Jean Remix
Lagloinski
[Intro: Lagloinski] (Uh, uh) ***Billie***... ***Jean*** Bitch don't ash on my Amiri... [Verse: Lagloinski & Michael ***Jackson***... screaming like (Oh-oh, oh-oh) ***Billie***... ***Jean*** is not my lover She's just...

HoRsErAdIsH
YoungBoy Cakin
no ones the one call the world ***Billie***... ***jean*** (***Billie Jean***, Michael ***Jackson***...

Billie Jean
Marcela Mangabeira
lover ***Billie Jean*** is ***Billie Jean***... ***Jean*** is ***Billie Jean*** is ***Billie***... ***Jean*** is not my lover ***Billie Jean***... ***Billie Jean*** is ***Billie Jean*** is... my lover ***Billie Jean*** is ***Billie***...

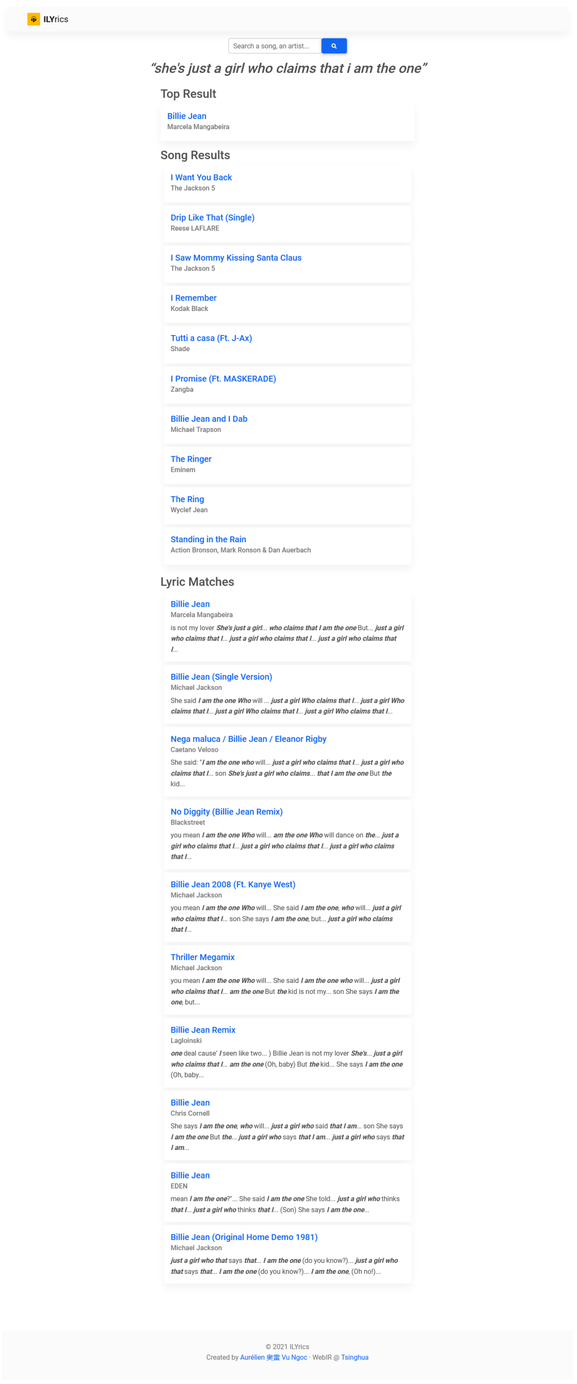
Pop Music (Ft. Michael Jackson)
Maurice Dekatt
Michael Joseph ***Jackson*** was born... At the age of 5, ***Jackson*** began... ***Jackson*** released his first solo... international hit singles as "***Billie***... ***Jean***" and "Beat It," Michael ***Jackson***...

Billie Jean (Single Version)
Michael Jackson
Billie Jean is not my lover She's... ***Jean*** is not my lover ***Billie Jean***... is not my lover ***Billie Jean*** is... not my lover ***Billie Jean*** is not... my lover ***Billie Jean*** is not my...

Rush
KAILAS
, yeah she cappin' Yeah, like ***Billie***... ***Jean***, Michael ***Jackson*** I'm in the...

© 2021 ILYrics
Created by Aurélien 奥雷 Vu Ngoc · WebIR @ Tsinghua

Fig. 5. Results using the song title and the artist name


ILYrics

Search a song, an artist...

"she's just a girl who claims that i am the one"

Top Result

Billie Jean
Marcela Mangabeira

Song Results

I Want You Back
The Jackson 5

Drip Like That (Single)
Reese LAFLARE

I Saw Mommy Kissing Santa Claus
The Jackson 5

I Remember
Kodak Black

Tutti a casa (Ft. J-Ax)
Shade

I Promise (Ft. MASKERADE)
Zangba

Billie Jean and I Dab
Michael Trapson

The Ringer
Eminem

The Ring
Wyclef Jean

Standing in the Rain
Action Bronson, Mark Ronson & Dan Auerbach

Lyric Matches

Billie Jean
Marcela Mangabeira
is not my lover ***She's just a girl***... ***who claims that I am the one*** But... ***just a girl who claims that I***... ***just a girl who claims that I***... ***just a girl who claims that I***...

Billie Jean (Single Version)
Michael Jackson
She said ***I am the one Who*** will ... ***just a girl Who claims that I***... ***just a girl Who claims that I***... ***just a girl Who claims that I***... ***just a girl Who claims that I***...

Nega maluca / Billie Jean / Eleanor Rigby
Caetano Veloso
She said: "***I am the one who*** will... ***just a girl who claims that I***... ***just a girl who claims that I***... son ***She's just a girl who claims***... ***that I am the one*** But ***the*** kid...

No Diggity (Billie Jean Remix)
Blackstreet
you mean ***I am the one Who*** will... ***am the one Who*** will dance on ***the***... ***just a girl who claims that I***... ***just a girl who claims that I***... ***just a girl who claims that I***...

Billie Jean 2008 (Ft. Kanye West)
Michael Jackson
you mean ***I am the one Who*** will... She said ***I am the one, who*** will... ***just a girl who claims that I***... son She says ***I am the one***, but... ***just a girl who claims that I***...

Thriller Megamix
Michael Jackson
you mean ***I am the one Who*** will... She said ***I am the one who*** will... ***just a girl who claims that I***... ***am the one*** But ***the*** kid is not my... son She says ***I am the one***, but...

Billie Jean Remix
Lagloinski
one deal cause' ***I*** seen like two...) Billie Jean is not my lover ***She's***... ***just a girl who claims that I***... ***am the one*** (Oh, baby) But ***the*** kid... She says ***I am the one*** (Oh, baby...

Billie Jean
Chris Cornell
She says ***I am the one, who*** will... ***just a girl who*** said ***that I am***... son She says ***I am the one*** But ***the***... ***just a girl who*** says ***that I am***... ***just a girl who*** says ***that I am***...

Billie Jean
EDEN
mean ***I am the one***?"... She said ***I am the one*** She told... ***just a girl who*** thinks ***that I***... ***just a girl who*** thinks ***that I***... (Son) She says ***I am the one***...

Billie Jean (Original Home Demo 1981)
Michael Jackson
just a girl who that says ***that***... ***I am the one*** (do you know?)... ***just a girl who that*** says ***that***... ***I am the one*** (do you know?)... ***I am the one***, (Oh no!)...

© 2021 ILYrics
Created by Aurélien 奥雷 Vu Ngoc · WebIR @ Tsinghua

Fig. 6. Results using part of lyrics