# Movie Search Engine Based on Description

Dongjian Chen, Ren Wang, Yumeng Shao
djichen@umich.edu
renw@umich.edu
emilysym@umich.edu
School of Information, University of Michigan
Ann Arbor, Michigan

## ABSTRACT

In this report, we describe our work on description-based movie retrieval. Our search engine is motivated by the idea of helping people retrieve movies when they could only remember fragments of the movies. We adopted a dataset with movie titles, descriptions and keywords as our search database. BM25+ function is adapted after we preprocess the dataset with with tokenizing, stemming, stopword and punctuation removal, and keyword addition. By calculating the harmonic mean of the ranks of the desired answers in our test cases, we are able to evaluate our system and conclude that our search engine has a fairly good performance.

## KEYWORDS

movie, BM25+, information retrieval, search

## 1 INTRODUCTION

Watching movies has become one of the most popular ways of entertainment, and there are hundreds and thousands of new movies being released every year. The information we recieve is explosing and the challenge of remembering the right title for movie plots is becoming larger and larger, since an average person could have watched dozens or hundreds of movies. It is not rare for someone to bump into an annoying situation when they could only remember a rough story or some segments of the movie, but just could not recall its name. In such cases without a specialized movie search engine, existing search websites will return various results that include not only movies and the search results would not be very ideal with queries composed of common terms but has specialized meanings and indications when it comes to movies.

As a result, we we aim at helping people retrieve the film they want when they could only remember a part of the story plot or segments like character names and special props. The search engine we designed and implemented helps with finding the corresponding movies with description or keywords provided by the user that specialize in movies.

## 2 DESIGN DESCRIPTION

The workflow of our design process includes getting data, processing data, selecting model and building interface. And the working process of our final design is shown as following figure,

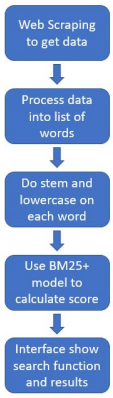


**Figure 1: Working process of our searching system**

### 2.1 Dataset

We obtained our data from two sources:

Kaggle movie dataset[1]

IMDB website[2]

We first downloaded a csv file from Kaggle with data about overviews and imdb link for each movie. We initially tried to use these overviews as documents for queries. However, we found that the performance of searching couldn't fulfill our satisfication so we tried to find more documents. Finally, after trying the summary on wikipedia and the total scene description on IMDB, we chose keywords on IMDB and used python webscraping to get these data.

### 2.2 Basic Data processing

In this project, our retrieval system is built to retrieve movie by its description and our data document for each movie contains its overview and keywords related to it.

We first scraped keyword data from a website and combined it with the overview of each movie from the Kaggle dataset into one string. Then, we tokenize the string into words and do stem and lowercase on each word and get lists of words for each document. Finally we remove stopwords and punctuations, and the document is then ready for query. The above mentioned steps are also applied to query sentences.

### 2.3 Advanced Data processing

We tried two advanced methods on data to see if the performance of search can be enhanced.

First, we thought of adding the synonyms of each words in the query so that those words with similar meanings as the query words in documents would not be missed. For each query we used nltk wordnet to get synonyms for each word in the query and then added them into the query. However, the performance become worse than without synonyms added. We thought that might be due to that adding synonyms has made the query vaguer so that more unrelated results are returned.

Also, we thought of adding weights to different kinds of words in the query. But that will led to another problem of how we should judge the importance of different words. We thought of using part-of-speech tagger to tag the words in the query and assigne a larger weight to adjectives and verbs. However, there is no distince improvement in the performance of our system. We believe that higher performance might be achieved if we can do further research and more experiments to find proper weighting methods on the queries.

## 2.4 Model

We first used BM25 score function, which is used to calculate scores to each query-document pair by the term frequency(TF) of each word in the query and inverse document frequency (IDF). The formula is as follows:

$$S = \sum_{t \in q} \cdot \left[ \frac{(k_1 + 1) \cdot c_t^d}{c_t^d + k_1 \cdot \left(1 - b + b \cdot \frac{l_d}{L}\right)} \right] \cdot \ln\left(\frac{N+1}{N_t}\right)$$

However, the performance of BM25 score function did not perform very well on our data set and we turned to BM25+ function instead. The differences between BM25+ and BM25 are that BM25+ has an additional parameter, and takes the frequency of each words in query into account. Since the performance of BM25+ is better and can meet our needs to a greater extent, we finally chose it as our score function (Inspiration from [3] which is,

$$S = \sum_{t \in q} \frac{(k_3 + 1) \cdot c_t^q}{k_3 + c_t^q} \cdot \left[ \frac{(k_1 + 1) \cdot c_t^d}{c_t^d + k_1 \cdot \left(1 - b + b \cdot \frac{l_d}{L}\right)} + \delta \right] \cdot \ln\left(\frac{N+1}{N_t}\right)$$

This BM25+ function adds two more parameters to change the weight of TF and IDF in the function and it achieves a better performance with our dataset. The performance comparison and analysis will be detailedly discussed later in the Result and Evaluation part.

## 2.5 Interface

We have also built an interface for users to input the description they remember and it will return the retrieved movie name along with a simple overview. The movie with the highest score will be shown directly on the page and users can see more results with lower scores if they scroll down the page as demonstrated on the following figures.

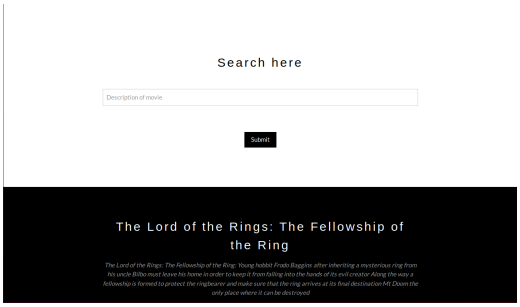


**Figure 2: Interface**

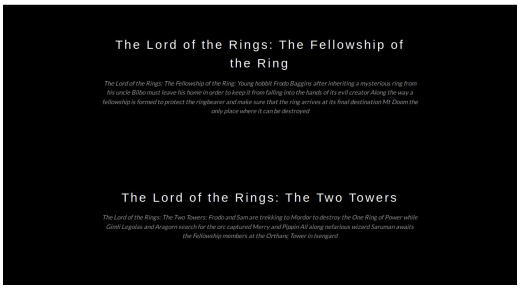


**Figure 3: Interface slide down**

# 3 RESULTS AND EVALUATION

We formulated 35 short descriptions of movies and use our search engine to fetch the result. Some of the queries come from us and some of them are from our users during the testing phase of our development. For each description, we check if the intended result is returned by the search engine.

If the intended result is returned, we check the rank $R$ of the expected movie returned by the search engine. We define $\frac{1}{R}$ as the precision of our search engine at this search, and if the movie is not retrieved within the top 10 movies, we assign R with infinity. Finally, we calculate the harmonic mean of $R$ of all our 35 tests and use it as the final evaluation for the search engine.

Following is part of the testing results from our search engine, the third coloum is the rank of the movie retrieved based on BM25 and the fourth column is for BM25+.

**Table 1: Part of Our Queries Test Result**

| Query | Answer | R by BM25 | R by BM25+ |
|---|---|---|---|
| Jack and rose | Titanic | 2 | 1 |
| Dinosaur | Jurassic World | 4 | 1 |
| France revolution | Les Misérables | 4 | 2 |
| Dream thief | Inception | 6 | 1 |
| Boy magic school | Harry Potter | 9 | 1 |
| Cowboy astronaut | Toy story | 5 | 1 |
| Robot kill human | Terminator | 4 | 1 |

The harmonic mean of ranks of the expected movies can efficiently evaluate the average results the user need to skim through to find the movie they intend to find. Thus, the lower harmonic mean of the rank, the better our search engine is. The harmonic mean of rank result of BM25 and BM25+ is shown as follows.

**Table 2: Comparison of Harmonic Mean of Rank**

| | BM25 | BM25+ |
|---|---|---|
| $n/\sum_{i=1}^{i=n} \frac{1}{R_i}$ | 2.944 | 1.449 |

A sorted distribution of expected movie retrieved ranks also clearly visualize the difference. For the figure concerning BM25, there are a lot of queries that returns failure (with rank greater than 10), while for BM25+, there is no failure for our test cases and the mean of ranks is significantly lower.

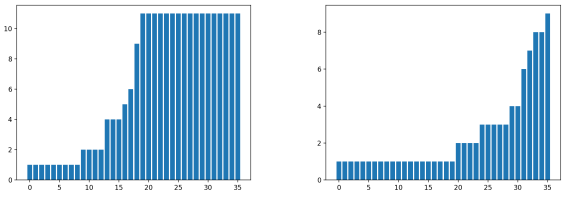

**Figure 4: Expected Movie Retreived Ranks Distribution**

With this comparison and precision shown, we can conclude that our search engine is useful with a BM25+ function, since the user can expect to find the moive they indicated from the first and second result retrieved on average.

### 3.1 Comparison between our search engine and Google

It's hard to quantatively compare our search engine's capability with Google, because for each query, Google returns results of webpages, images and even links to external websites. However, the common behavior of Google is to treat the query "as it is". For example, when the user wants to find the movie "Jurassic World" when he or she only remembers scenes of Dinosaurs, Google will return some sophisticated documentary movies of "Dinosaur", but what the user really want is "a popular movie in which dinosaurs show up". However, with our search engine, the user can directly get it with those keywords. Thus, for accomplishing the job of searching movie based on fragments of description, our search engine can beat Google.

## 4 CONCLUSIONS

The result of this project is generally satisfying with our experiments with it. We succeeded in returning the related results with the queries and has improved its performance with data processing and parameter tuning. Thus, with our search engine, people can get the movie they referred to in most of the cases. However, the search engine might not work well with subjective feelings on movies since the data we used for query are neutral descriptions.

## 5 FUTURE IMPROVEMENTS

During data processing, we have tried part of speech tagging on the sentence and give more weights to certain types. However, it doesn't turn out well as we expected. We believe that digging further into this by analyzing the parse tree to evaluate the part of speeches more precisely can help with giving better weights. Moreover, we believe that keeping a search and view log can also help with improving accuracy. Within a series of similar searches, we can analyze the movies the user has clicked into and find the similarities to feed back into the algorithm and adjust the weights accordingly. Last but not least, we think that taking the reviews of the movies into account might also help with enhancing the performance. We might extract some highly mentioned keywords from the movie reviews so that we can deal with some more subjective viewer feelings on the movies. This might be of importance since our target users have vague memory on the movie and is likely that their memory is affected by subjective feelings.

## ACKNOWLEDGMENTS

Prof. Qiaozhu Mei, Mr. Wei Ai from University of Michigan; Teaching Assistant [Shiyan Yan, Xinyan Zhao] for SI 650 at University of Michigan.

## REFERENCES

[1] Rounak Banik. 2017. *The Movies Dataset,Metadata on over 45,000 movies. 26 million ratings from over 270,000 users.* Retrieved Dec 16, 2019 from https://www.kaggle.com/rounakbanik/the-movies-dataset

[2] IMDB. 2019. *IMDB.* Retrieved Dec 16, 2019 from https://www.imdb.com/?ref_=nv_home

[3] Peilin Yang and Hui Fang. 2016. A reproducibility study of information retrieval models. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval.* ACM, 77–86.