Movie Recommendation Search Engine Design

Daniel Elston and Group Members
School of Electronic Engineering and Computer Science - MSc Data Science and AI
Queen Mary University of London

June 11, 2022

Contents

1 Overview

The search engines will take user input queries, analyse metadata regarding movie descriptions and output a relevant movie recommendation. The documents and queries will be indexed using tokenization, stop word removal, lemmatization and stemming. The documents will be vectorized, then inverse document frequency (IDF) will be applied. Finally, the BM25 frequency conversion model is used with parameters $k_1 = 1.2$ and $b = 0.8$. A full visual of the architecture can be seen in figure 1. This provides each relevant document with a score. An output is then generated in the form of retrieved documents. Objectives of the search engine include recommending movies based on user input query, and improving overall customer satisfaction.

2 Technical Aspect

2.1 IR Techniques

- Tokenization
- Stop word removal
- Lemmatization
- Stemming
- Vectorization
- BM25

2.2 Technology

- Language: Python 3.6.9
- Libraries: pandas, numpy, sklearn, nlkt, spacy, seaborn
- Installs: nltk
- Packages: nltk.omw-1.4, nltk.wordnet, spaCy

2.3 Raw Data Reference

(Harshit Shankhdhar) (February 2021) (IMDB Movies Dataset) (Retrieved 08.03.2022)

https://www.kaggle.com/harshitshankhdhar/imdb-dataset-of-top-1000-movies-and-tv-shows/activity

3 Problem Formulation

With the rise of digital streaming services such as Netflix and Amazon Prime, the use of movie and TV based information retrieval has become a part of our everyday lives. Users of these services generally search for the names of movies and shows directly and are returned a selection of relevant titles. Our aim is to go beyond this and create a search engine that allows users to input more general queries, for example plot points, to retrieve their desired results.

During our research paper analysis, we discovered the many challenges faced in information retrieval within the context of social media, from indexing and query formulation to the evaluation stage. In reference to this, the incorporation of relevant data pre-processing of the queries and documents are essential components within the information retrieval model. Additionally, a focus on the ability to retrieve relevant documents based on limited query material was highlighted. As a result, the aim of this project is to create a search engine that returns relevant movies and TV shows based on queries given by the user. This could further be used as a form of recommendation. Moreover, we hope to explore novel information retrieval techniques and ways to optimise our model at every section of its architecture.

4 Dataset Introduction

The dataset used to evaluate this search engine is the IMBD movies dataset. It has been used to analyse many film industry relationships such as movie rating and gross profits. The dataset provides a wealth of information regarding movies including the 'overview' feature which gives an excellent summation of each movies in the form of an individual document. The Features of the dataset include:

- Poster Link - URL link to the movie poster.
- Series Title - Movie title.
- Genre - The genres in which the movie has been categorised.
- Certificate - Suitability for audience measure.
- Runtime - Duration of the movie.
- IMDB Rating - User rating of the movie on a 1-10 scale.
- Overview - Summary of the movie plot.
- Director - Director of the movies name.
- Star 1, Star 2, Star 3, Star 4 - Main actors/actresses in the movie.
- No of Votes - Number of rating votes on IMBD website.
- Gross - The gross profit the movie made.

The query type will be searched based on the movie/TV show overview. An example of basic query would be 'A love story' or 'war'. The model will return the relevant movies/TV shows based on statistical data retrieved from the overview feature. Unfortunately, the dataset provides no ground truths in relation to movie overview and for this particular search engine, ground truths are difficult to define.

5 Architecture

The search engine consists of 5 major stages: Query processing, Indexing, Retrieval Framework, Results and Evaluation. The full architecture can be seen in figure 1. The user will input a query in the form of text, then the search engine will process the query and assign it to relevant documents. This produces an output as text movie recommendations. The components of each of these stages are briefly described below.

5.1 Query

The Query Processor is responsible for executing database queries, it receives input in the form of text and parses so that query text is converted from high level queries to low-level expression and is then passed on to the retrieval framework.

5.2 Indexing

Indexing is used to reduce tokens down to their simplest form so that the engine can more accurately identify relationships. The Indexing components include:

- Tokenization of documents: The breaking down sequence of strings into words, keywords, phrases, symbols and numbers, called tokens. Once broken-down parts of the tokens are filtered out such as punctuation, symbols and numbers. In our search engine we have decided to exclude numbers from being filtered out during the tokenization stage due to the fact reviews may include the year of the movie/tv-series.
- Stop Word Removal: Words that occur frequently such as 'and', 'a', 'or', are removed.
- Lemmatization: Conversion of words in the document into dictionary form to reduce inflectional or variations of words/verbs to its original base form.
- Stemming: Removal of the last few characters of a word to reduce the word, to the word's stem.

The Indexing Framework will encapsulate links to semantic relations between words for indexing, this will be carried out using the following:

- WordNet API: A lexical database of semantic relations between words that can be utilised to get information for a given word or phrase such as synonyms, hyponyms and meronyms of the given word or phrase.
- spaCY (Spacy.io, n.d.): A more object orientated library that is beneficial for information extraction tasks wherein which one would need to know what the text is about, what the context of the word is and identify which texts are similar to each other.
- NLTK is a string processing library that takes the string as the input and returns strings or list or strings as an output.

WordNet, spaCY and NLTK will be used for the stemming, tokenization and lemmatization steps during Query processing and Indexing.

5.3 Retrieval Framework

Using the output from query processing and indexing, the retrieval framework will be implemented. This framework will comprise of the BM25 (Best Match 25) model. The BM25 model is one of the most widely used information retrieval functions because of its consistently high retrieval accuracy [1].

The BM25 term-weighting and document-scoring function will retrieve the list of documents that ranks a set of documents based on the query terms appearing in each document [2]. For a given query, the retrieval status value is the sum of individual term scores,

$$rsv_q = \sum_{t \in q} \log\left(\frac{N}{df_t}\right) \frac{tf_{td}(k1+1)}{tf_{td} + k1\left[(1-b+b\left(\frac{L}{L_{avg}}\right)\right]} \quad (1)$$

where N is the number of documents in the collections, df_t is the number of documents containing the term (document frequency), tf_{td} is the number of times the term (t) occurs in document (d), L is the length of the document and L_{avg} is the mean of the document lengths. There are two tuning parameters, b and $k1$. This is a simplified version of the BM25 formula. The first term is the formula for inverse document frequency (IDF). The BM25 will rank matching documents according to their relevance to the given query after being given the list of documents and the specified query. The highest scoring documents will be suggested first.

6 Evaluation Techniques

The first step in our evaluation is to create a test collection consisting of three things: documents, information needs, and a list of relevance judgements. The test collection will be expressed as a list of query-document pairs and a binary relevance score that is 1 for relevant documents and 0 for non-relevant documents. An example of this could be ("War film", "1917", "1") which corresponds to (query, document, relevance). We will use our domain knowledge of movies and TV, information from our dataset and data obtained by scraping techniques, to score the relevance of a document with respect to the information need.

It is estimated, that the test collection must have at least 50 information needs to cover in test collection [3]. We will then calculate our two basic evaluation metrics: Precision and Recall. The engines recommendation precision is given by,

$$P = \frac{N_{rel}}{N_{rec}} \quad (2)$$

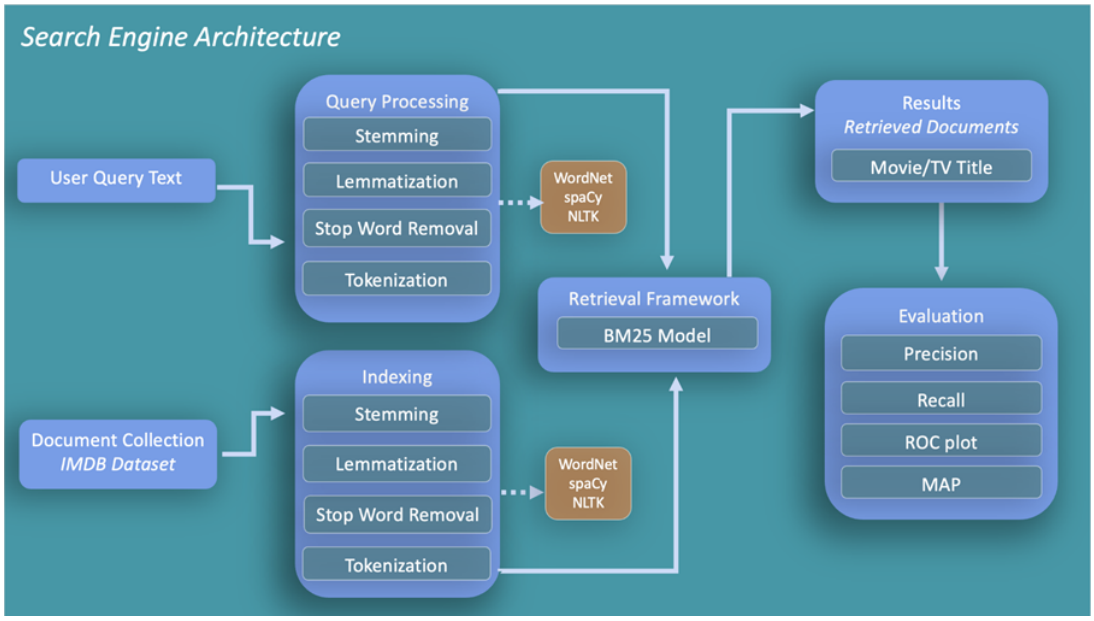

Figure 1: Pipeline of a search engine.

where N_{rel} is the number of relevant recommendations the engines makes and N_{rec} is the number of movies recommended. Precision measures the proportion of retrieved documents that are relevant. The engines recommendation recall is given by,

$$R = \frac{N_{rel}}{N_{tot}} \tag{3}$$

where N_{tot} is the number of all possible relevant movies. Recall is the proportion of relevant documents that have been retrieved. We will also use further evaluation metrics such as MAP (Mean Average Precision) and ROC plots to gain a more comprehensive view of the evaluation conducted on the search engine.

7 Optimisation Techniques

7.1 Stochastic Optimisation

Stochastic optimisation is used as an alternative to deterministic decisions in an effort to improve the likelihood of locating the global optima or a better local optima [4]. Stochastic algorithms help the model produce reliable results when noise or randomness is present in the data. Due to the fact the overview feature was written by humans, there will be an element of noise or randomness. Similarly, each query input will have randomness in it, with many queries likely showing a significant amount of noise. By implementing this algorithm, noise and randomness can be reduced, improving precision and recall values for the engine.

7.2 Industry Specific Optimisation

The search engine is clearly successful if the results yielded are compared to a ground truth and the end output is deemed a match. However, in the advancement of technology and its availability over the last decade, a simple query-document match is not always the best case. Netflix customers do not want the same films suggested over and over. Therefore, a further consideration for the engine could be industry specific optimisation, whereby the engine is optimised to suit its industrial needs. In the case of Netflix, perhaps a simple reshuffling of the top 20 output documents suggested is all that is needed to improve customer satisfaction.

7.3 Further Considerations

Thinking in terms of queries and not the under-lying information can lead to over optimizing the engine, resulting in dramatic variance in system effectiveness [5]. It has been documented that the implementation of a multi-stage retrieval system could yield more accurate and reliable results. Namely the BM25 model would be coupled with the monoBERT model [6]. This would ultimately make use of two models to feed the indexed documents through. Finally, some form of feature mapping could yield excellent results. The obvious choice would be the mapping of the 'genre' and 'overview' features. By either combing the two features or running both through the model and combining results, more relevant documents and therefore movie recommendations could be generated.

8 Conclusion

This paper has outlined a search engine design used to recommend movies from user input queries. The BM25 model will be used as the retrieval framework for the engine with standard indexing techniques outlined. A large

dataset consisting of 11 features and 1000 observations is used to produce raw documents of brief but summarising movie overviews. Evaluation techniques have been discussed to assess the reliability in the results produced by the engine. Finally, ideas for optimisation are given with the aim of improving the models architecture and results further.

References

[1] Krysta M Svore and Christopher JC Burges. A machine learning approach for improved bm25 retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1811–1814, 2009.

[2] Ling Liu and M Tamer Özsu. *Encyclopedia of database systems*, volume 6. Springer, 2009.

[3] Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103, 2010.

[4] Jason Brownlee. *Better deep learning: train faster, reduce overfitting, and make better predictions*. Machine Learning Mastery, 2018.

[5] J Shane Culpepper. Single query optimisation is the root of all evil. In *DESIRES*, page 100, 2018.

[6] Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. Multi-stage document ranking with bert. *arXiv preprint arXiv:1910.14424*, 2019.