Wikipedia Recommender System

Jedrzej Miczke, Michał Redmer

December 2024

1 Introduction

Recommender systems have become an integral part of modern information retrieval and personalization techniques, providing users with suggestions tailored to their preferences or interests. This project focuses on the development of a recommender system for Wikipedia, one of the largest and most frequently accessed online encyclopedias.

The system is designed to recommend relevant Wikipedia articles based on the content of a selected article. By leveraging Wikipedia's extensive hyperlink structure and textual content, the system can identify articles that share thematic or topical similarities with the given page. Such a tool has potential applications in education, research, and general knowledge exploration.

To achieve this, the recommender system utilizes a combination of web scraping, text preprocessing, and machine learning techniques. Articles are scrapped directly from Wikipedia, preprocessed to extract meaningful features, and represented numerically using the Term Frequency-Inverse Document Frequency (TF-IDF) method. Recommendations are generated by calculating cosine similarity between the input article and the dataset of pre-scrapped pages.

This report outlines the methodology used to build the recommender system, highlights the steps involved in creating and processing the dataset, and evaluates its effectiveness in suggesting relevant content. The final implementation scrapes and processes 1,000 articles, ensuring broad coverage and relevance within the system's recommendations.

2 Methodology

The development of the Wikipedia recommender system involves several sequential steps that ensure effective scraping, processing, and ranking of Wikipedia pages. This section outlines the methodology adopted to create and utilize the recommender system.

2.1 Dataset creation

The first step in building the recommender system is the collection of Wikipedia articles. This is achieved using a web scraper tailored for Wikipedia, which begins from a given starting URL. As the starting URL, we chose `https://en.wikipedia.org/wiki/Wikipedia:Popular_pages`, as we believed it would lead to scraping some of the most popular pages on Wikipedia, and therefore, in some way, the most representative ones. The scraper navigates the links within the content section of each page to gather textual data from related articles. In the final version of the system, 1,000 articles are scrapped to form a comprehensive dataset.

Scraper extracts from the HTML of each website relevant textual content; it skips non-content links such as those to lists, categories, or administrative pages, ensuring the relevance of the dataset. The network of articles is traversed with the BFS algorithm.

2.2 Text preprocessing

The raw textual data is subjected to preprocessing to standardize and clean the content using Python's `nltk` package. This includes:

- tokenization: splitting the text into individual words;
- stemming: reducing words to their root forms using the Porter Stemmer;
- filtering: some words are removed - this includes stopwords, which are commonly discarded in recommendation systems, and words that are not English - we decided for this to significantly reduce the size of the generated database.

These steps yield a cleaned and standardized textual representation for each article, which is essential for accurate feature extraction.

2.3 Feature representation using TF-IDF

The processed text is transformed into numerical features using the Term Frequency-Inverse Document Frequency (TF-IDF) method. TF-IDF computes the frequency of every word present in the dictionary(vocabulary from all the documents) in each article and additionally scales the frequency by the importance of the given word - so the negative logarithm of frequency of the given word in all documents. It provides a vectorized representation of each article. The resulting TF-IDF matrix serves as the basis for measuring similarity between articles.

2.4 Generating recommendations

Once the recommender system is created, it can be used to create recommendations for different Wikipedia pages. This process includes the following steps:

1. Calculating cosine similarity between the input page's TF-IDF vector and those of the articles in the dataset.

2. Ranking the articles based on their similarity scores, with higher scores indicating greater relevance.

3. Excluding from the ranking articles for which recommendations were generated - this is the default behavior that can be changed by the user.

3 Analysis of the database

#	URL1	URL2	Cosine Similarity
1	George H. W. Bush	George W. Bush	0.975253
2	Red (Taylor Swift album)	1989 (album)	0.970129
3	Grand Theft Auto IV	Grand Theft Auto V	0.964991
4	25 (Adele album)	21 (Adele album)	0.964192
5	Family of Donald Trump (Barron Trump)	Donald Trump	0.962163
6	The Eminem Show	Recovery (Eminem album)	0.954863
7	2004 Indian Ocean earthquake and tsunami	Tsunami	0.952070
8	Red Dead Redemption	Red Dead Redemption 2	0.942065
9	25 (Adele album)	The Eminem Show	0.941700
10	The Eminem Show	21 (Adele album)	0.941026

Table 1: Most similar articles

The above table lists the most similar articles from our article database, based on their cosine similarity scores. The achieved similarities are very high and the generated pairs of articles seem very reasonable. For example, in the first place are George H. W. Bush and George W. Bush, two presidents of the United States, but even more importantly a father and son (what we learned just now, thanks to the great performance of our recommender system!).

#	URL1	URL2	Cosine Similarity
1	Second impeachment of Donald Trump	Ice Age	0.003140
2	Dick Van Dyke	Undefined	0.003437
3	Arc de Triomphe	Searching	0.003443
4	Spider	XXXX	0.003984
5	Dick Van Dyke	Tsunami	0.004091
6	Jaguar	XXXX	0.004141
7	Judo	Shark	0.004174
8	Daft Punk	Alpha Centauri	0.004183
9	Shark	Ice Age	0.004310
10	Kish (Sumer)	Shark	0.004334

Table 2: Most dissimilar articles

This table, conversely, proves that the devised method, i.e., cosine similarity between preprocessed articles, creates counterintuitive results in general. None of the pairs of articles with the lowest similarity seem to resemble each other, which was desired. Interestingly, the Wikipedia shark page appears several times, which might be caused by the biological domain-specific vocabulary there and even more importantly lack of other animal-connected articles in our database.

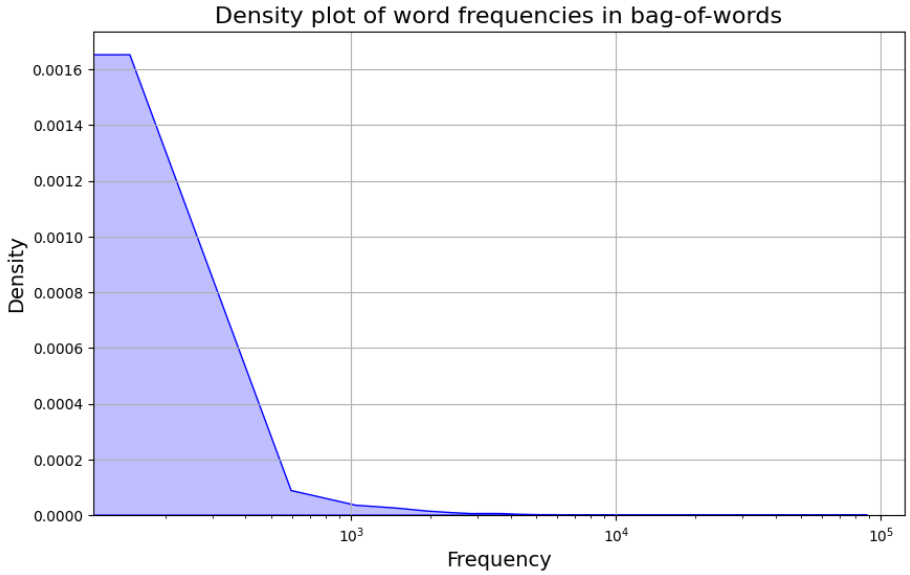

Figure 1: Word frequency density

Here we can see the density distribution of frequencies of word occurrences from our dictionary, built on these 1000 most popular Wikipedia articles that we have scrapped. The words accounted for here are already words from the preprocessed articles, therefore stemmed and without stop words. As we see, the shape of the distribution roughly follows the pattern of the reciprocal function 1/x, which is expected from the Zipf law(the logarithmic scale on the x-axis distorts the real shape, but makes it more readable)

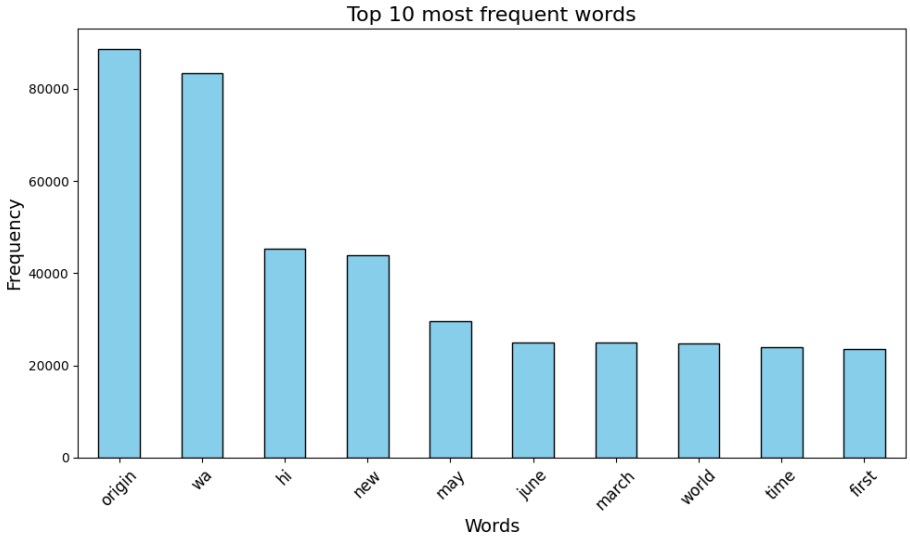

Figure 2: Most frequent words

Here are the 10 most frequent words, not only in the form of curiosity but also allowing us to assess the quality of the stop-word deletion operation that we have undertaken. Seeing a stop word here would mean that the recall of the aforementioned operation was not perfect. Fortunately, there are no stop-words in the dataset, though 'wa' appears as an unusual word. However, upon investigation, we found that it is an abbreviation for 'Washington.' This suggests that our dataset may be particularly rich in references to the American presidency.

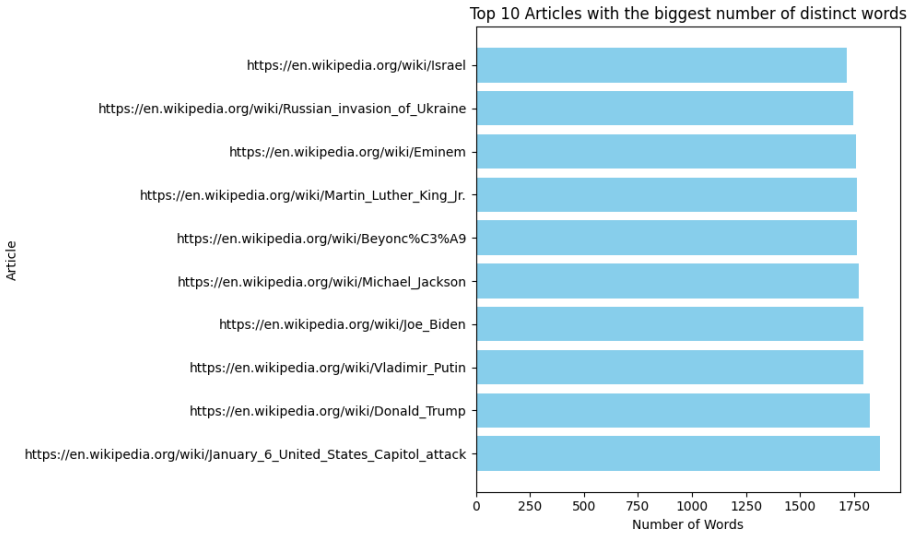

Figure 3: Articlwith the most distinct words after preprocessing

As the last step of the preliminary analysis, we have created the bar chart displaying the 10 articles with the most number of distinct words. In these articles the writers were the most creative, or the articles could have been simply the longest ones. It seems most often these were the biographical pages, but also some pages commemorating important events of recent history.

4 Analysis of generated recommendations

Here we will test our recommender system on the various Wikipedia articles from outside of our database. We will try to find the best matches from our database to the chosen by user article/articles and then we will subjectively gauge the outcome.

4.1 Basic single recommendation

Firstly we will start from what we expect a simple task: prediction for the Wawel Castle. Wawel Castle Why shall it be easy? Our database consists of the most popular Wikipedia articles, thus there are several pages devoted to the famous landmarks. We expect our system to advertise them.

#	URL	Cosine Similarity
1	Windsor Castle	0.488749
2	Church of the Holy Sepulchre	0.447490
3	Poland	0.436008
4	Tower of London	0.409676
5	Neuschwanstein Castle	0.378154

Table 3: Top recommendations for Wawel Castle

As seen in Table 3, the algorithm performed exceptionally well. It not only recommended other landmarks but also landmarks of the same category (two castles) and a similar category (a tower and a church).

Moreover, it suggested the Wikipedia page for Poland, which is a perfect choice. If a foreigner is captivated by the beauty of Wawel Castle but does not know where it is situated, this recommendation would inform them about its location and undoubtedly inspire them to visit our great country.

4.2 More challenging case

Here we will explore a more difficult example. Since our database consists of 1000 most popular articles, there isn't an abundance of domain-specific vocabulary for any field. So let's try to address this and see the recommendation for the term from linear algebra: tensor Wikipedia page Tensor

#	URL	Cosine Similarity
1	Albert Einstein	0.054730
2	Dark energy	0.040743
3	Universe	0.036845
4	Earthquake	0.036000
5	Political spectrum	0.034406

Table 4: Top recommendations for Tensor

As we can see the similarities are very small, even for the top results, however the matches are surprisingly good. The top 3 matches in our opinion are great: Albert Einstein used the tensors, to describe dark energy the tensors are indispensable, not to mention the equations needed to model the workings of the universe(at any scale). The 5th result may be arguable, however, we believe it stems from the lack of better articles in the database, we examined looking at the top 10 most similar articles for this recommendation (not output here).

4.3 A particularly tricky case

Here we delve even deeper into the domain-specific vocabulary to show that we are also aware of the limitations of our model. However great our model performed on the previous inferences, it comes with some kind of performance limitations. Let us consider the prediction for this commutative ring Wikipedia page. Commutative ring

#	URL	Similarity
1	The Lord of the Rings	0.490172
2	Saturn	0.280332
3	Uranus	0.201618
4	Moscow	0.185460
5	The Hobbit	0.162240

Table 5: Top recommendations for Commutative Ring

Wow, such a domain-specific term (what does it even mean?) and articles with such great similarities were found. That's very nice. In fact, not really that nice. Upon the investigation (not a very thorough one was needed) we can see that the proposed articles are not really similar(semantically/topic-wise). But what caused the algorithm to recommend them? Our algorithm as seen above is not resistant to polysemy. The commutative ring and the Lord of the Rings happen to share the term ring. That recurring ring, though in different meanings, could make our algorithm believe the articles were similar. To prevent such an issue, we would have to perform some kind of latent semantic analysis. However, it would be at the expense of the speed of the algorithm.

4.4 Recommendations for more than one article

Now we will explore the work of the recommender in case it recommends more than one article. Let's see the working for such a tuple of articles: Chess and Checkers What we expect is an article connected with games/board games in specific/ leisure time activities in general, let's see!

#	URL	Similarity
1	Tennis	0.164197
2	Fallout 4	0.160449
3	Association football	0.156598
4	Minecraft	0.156029
5	American football	0.139120

Table 6: Top Recommendations for Commutative Ring

The recommendation fulfills our expectations. What chess and checker have in common is that they are both board games, and less specifically they are both leisure time activities (unless Magnus Carlsen would want to use our recommender). The same can be said for the provided recommendations. Tennis, Association football (this is football), and American football are sports - leisure time activities for most of people. Fallout 4 and Minecraft are computer games - also leisure time activities for most of the people.

5 Conclusions

We have thoroughly described all the elements of our system: Beginning with the introduction of the topic, followed by the methodology and data analysis, and finished with the testing and overview of the results. To summarize, We are satisfied with the system we have created, as we have shown the predictions it makes are reasonable both for one article and for more than one. Of course, It's not invulnerable to difficulties stemming from the complexity of the task(as exposed in the example with the polysemy), but most of the time it makes: good, reasonable, and what's most important, useful recommendations.