Personalized movie recommendation system

Group Number: 08

Group Members:

Patrick Eckel
Marcus Gugacs
Martin Tobias Klug
Lukas Leitner

November 10, 2024

Contents

1 Abstract

A platform aims to help users find movies they want to watch by curating a selection based on their preferences. The platform uses a transformer-based architecture to process user queries and rank movie recommendations based on similarity. The system adapts to user feedback, refining recommendations to provide the most relevant content.

2 Idea

In today's world, the sheer volume of content available is overwhelming. It's challenging for many to find movies they genuinely want to watch amidst a sea of irrelevant material. Streaming providers and online distributors possess the potential to curate a concise selection for their users. However, many of them choose not to do so. The reasons behind this decision remain unclear, but it's possible that it's a strategy to keep users locked into their platforms for as long as possible. Instead of offering users a curated selection, they're presented with a flood of mediocre content that's nowhere near what they're truly interested in. Our mission is to value the time of users by making it easier for them to watch the movies they genuinely want to see. To achieve this, we're building a platform where users can explicitly define their movie preferences, whether it's general tastes or specific preferences for the moment. The system then diligently searches through a vast movie database to find an appropriate and concise selection that aligns with the user's criteria. The top results are presented to the user with brief descriptions of the movie's content, genre, actors, release date, and title. If the user isn't satisfied with the provided content, the system adapts and re-searches the database using the modified criteria. Alternatively, if the user finds a movie they enjoy, we strive to minimize the barriers to starting a movie by directly recommending a few services that offer the selected content. With a single click, users can seamlessly switch to their favorite streaming platform, stream or purchase the movie, and enjoy it without any hassle.

2.1 Goal

Our mission is to value human time by simply recommending movies they genuinely want to watch, without forcing them to sift through a catalog of irrelevant content. The demand for such a service will only increase over time as the number of movies and series created continues to grow exponentially. In fact, there are now more TV shows produced in a given period than ever before, and this trend appears to persist.

3 Main Task

Our primary objective is to design an advanced information retrieval system for movie recommendations, employing a transformer-based architecture. To accomplish this, our core component will comprise a fusion of transformer models, including:

- to perform text encoding
- for text summarization
- as a sentence transformer for semantic similarity

Technically, the retrieval pipeline will work approximately as follows:

- Processing the user's query, capturing their preferences.
- Document encoding of the movie preferences retrieved from the dataset.
- Calculating the similarity between the encoded user query and the encoded movie document.
- Ranking the results based on their relevance to the user.

After the initial phase of the retrieval pipeline concludes, the user has the option to assess their satisfaction with the results or to disregard irrelevant recommendations. If a selection of irrelevant content is made, the system will adapt and present a new selection based on the evolving user preferences. This approach aims to deliver the highest quality recommendations while minimizing the user's effort.

4 Dataset and Processing

This section delves into the details of the sources and processing of used data.

4.1 Data Sources

Our movie database is sourced from the "wykonos/movies" dataset, which is loaded through Hugging Face. The dataset comprises various features, including id, title, genre, original language, overview, popularity, production companies, release date, budget/revenue, runtime, status, tagline, average vote, vote count, credits, and keywords.

Additionally, we plan to use a provider like OpenSubtitles to obtain captions and subtitles, allowing us to generate informative summaries or recommendations based on the actual movie content. The provider's data would be fetched through the backend via a REST API. While OpenSubtitles is one option under consideration, we are still evaluating cost-effective or free alternatives to determine the best fit for our needs.

4.2 Data Processing

Before utilizing the dataset, we need to preprocess it through a data "cleaning" step. During this procedure, missing values (NaN) are replaced with empty strings to prevent future issues. Subsequently, each entry in the dataset is converted to strings.

5 Methods and Models

This section delves into the technical details of how our previously defined IR problem will be solved.

5.1 System Architecture

The following system architecture represents a high-level view of our inteded system implementation:

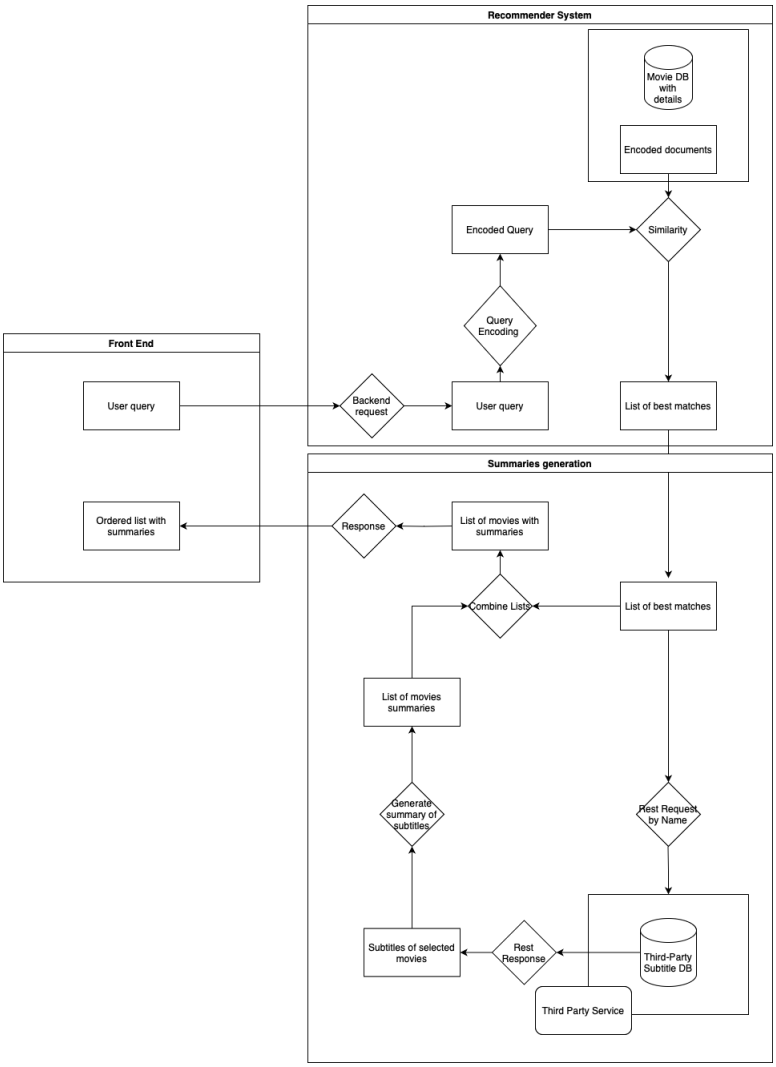

Figure 1: Representation of the planned system architecture

5.2 Implementation

The entire backend of the system will be implemented using Python. There are numerous useful libraries and tools that are beneficial for working on the given problem, such as PyTorch, Pandas, NumPy, and SciPy. The initial step is to perform the necessary input processing (encoding). For this purpose, an optimized transformer model will be utilized. Subsequently, we will use the model generated from the tokenized input to perform mean pooling of the last hidden states for similarity computation later on. In addition to the user input, we will also encode the movie descriptions. This enables us to compute a similarity measurement between the user input and the movie description (for instance, using cosine similarity). To provide the user with only the most relevant recommendations, a sophisticated ranking mechanism based on the similarity scores will be applied. Only the top three results, enriched with various additional movie metadata and a summary/recommendation of the content, will be displayed to the user. To generate the summary, we will first make a request to a caption/subtitle provider and then use the returned data in a summarization pipeline with an appropriate model. The recommendation will then be visualized in the frontend, which will be implemented using a modern web technology stack, including but not limited to NextJS. The recommendation can then

be evaluated by the user, and in the case of dissatisfaction, it can be re-evaluated by the model to update the recommendation using a selection of irrelevant movies suggested by the user.

5.3 Optimization questions

Due to the size of the dataset, various optimization questions arise. These include topics such as memory management strategies, enhanced scalability through batch processing and hardware acceleration (CUDA/MPS).

6 Evaluation

This section delves into the specifics of how we intend to assess the accuracy and utility of our system.

6.1 Metrics and results

To assess the accuracy of our system, we'll initially evaluate the generated summaries by cross-checking them against the hand-written summaries included in another dataset. If the results are promising, we'll conduct a small "user study" where users will report on their satisfaction with the provided recommendations. Our plan involves gathering feedback from five individuals who will perform ten queries to the information retrieval system. Subsequently, we'll analyze the feedback to calculate statistical metrics from the gathered results and evaluate their significance.