Patrick Eckel, Marcus Gugacs, Martin Tobias Klug, Lukas Leitner

# Movie Finder

**Project Report - Group 08**

**Graz University of Technology**

Lecture: Advanced Information Retrieval

Institute of Interactive Systems and Data Science
Lecturer: Assoc.Prof. Dipl.-Ing. Dr.techn. Elisabth Lex

Graz, January 2025

**Abstract**

This project develops a user-centric movie recommendation system to address content discovery challenges in the digital entertainment landscape. With the overwhelming variety of content across streaming platforms, users often struggle to find relevant movies. Existing systems prioritize platform metrics over personalization. Our approach uses a transformer-based architecture to process natural language queries, generating personalized recommendations with minimal effort. The system combines semantic analysis, TF-IDF similarity, and emotion analysis to deliver relevant suggestions, while also considering popularity metrics. Experimental results show high precision (99.08%) and acceptable response times, with room for improvement in recall. This modular system can be easily integrated into existing platforms, enhancing user experience and content discovery.

# Contents



# References

[1] Oscar I. Iglesias R, Carlos E. Pardo B, José Oñate López, and Christian G. Quintero M. Designing a movie recommendation system through a transformer-based embeddings space. In *2024 IEEE Colombian Conference on Communications and Computing (COLCOM)*, pages 1–6, 2024.

# List of Figures

# A Introduction

This section provides an overview of the project, including the motivation, and the research question.

## A.1 Motivation

The digital entertainment landscape has transformed, with unprecedented content production and availability. However, this abundance of choice has become a burden for consumers. Streaming platforms prioritize user retention over satisfaction, resulting in suboptimal viewing experiences. Content discovery is challenging due to cognitive load from navigating extensive catalogs across multiple services. Existing recommendation systems exhibit biases towards platform-specific content and popular titles, wasting user time. A user-centric approach is needed, prioritizing user time and preferences over platform metrics.

## A.2 Research Question

This project addresses a central research question: How can we build a central system using a transformer-based architecture to process natural language queries, to generate personalized movie recommendations and efficetly reduce the users effort of finding matching content?
This question encompasses advanced natural language processing, maintaining recommendation relevance, and system adaptability through user feedback. It directly addresses creating a more efficient and user-centric content discovery platform while acknowledging technical complexity in processing user preferences and ensuring system responsiveness.

# B Related Work

This section provides an overview of the literature and a brief discussion regarding our project. This section starts with a brief overview of the literature, followed by a discussion of our actual project implementation.

## B.1 Literature

There are several approaches to movie recommendation systems. For the literature overview we focus on literature matching the context of the "Advanced Information Retrieval" course. One paper espcially caught our attention, as it uses a transformer-based architecture to generate movie recommendations, but also discusses foundtaional steps like data preprocessing and dimensionality reduction. This is especially of interest to us, as the whole topic is novel to us and we are looking for a good starting point.
One team of researchers [1] proposes a movie recommendation system that uses a transformer-based embeddings space. To achieve this it was necessary to perform dimensionality reduction. The first thing, as in many data science projects, is to preprocess the data. In the case of the discussed project the dataset of 45.466 movies was reduced to just 11.236 [1]. The features of the movies where transformed to a matrix consisting of id, original title, release date, original language, runtime, vote average, vote count, popularity, poster path, production companies, genres, overview, keywords, cast, director and release year [1]. To generate meaningful embeddings of sentences the a variation of the BERT-family of models was used [1]. Through the usage of dimensionality reduction they were able to visualize the movies on a 2D-plane [1].
The mentioned steps already provide good insight into a starting point for such a system. This was a good orientation for our project, but the actual user-interaction is highly different than our approach. In the case of the mentioned project the user is able to select 10 movies and the system will then generate

a recommendation [1]. Afterwards the user is asked to rate the recommendation and the system will generate a new recommendation based on the feedback [1].

## B.2 Project Discussion

After reviewing the literature, we browsed through HuggingFace to find datasets, models and other resources to build our system. This was also the most informative part of our own research as we gathered lots of ideas and insights from the community and through the extensive collection of readmes and whitepapers on the platform. This sophisticated movie recommendation system uses multiple advanced methods to deliver personalized movie recommendations. It aims to use a approach balanced between computational efficiency, accuracy, and user experience. The system evluates recommendations based on five key parts:

- Semantic Analysis (40%): Uses the MPNet-based SentenceTransformer model to capture deep contextual understanding of movie descriptions and user preferences.
- TF-IDF Similarity (40%): Captures specific terminology and phrases.
- Emotion Analysis (15%): Using text-classification as a foundation the emotional accordance with the movie content and the mood choice by the user is evaluated.
- Vote Average Metrics (2,5%): One out of two factors to ensure recommendations remain somewhat connected to mainstream appeal.
- Popularity Metrics (2,5%): One out of two factors to ensure recommendations remain somewhat connected to mainstream appeal.

The system's robustness is enhanced through efficient text preprocessing, caching, batch processing and GPU (CUDA, MPS) acceleration.
Recommendation Generation Process:

- Filtering dataset based on trivial criteria (e.g. language, genres).
- Initial query processing combines user mood preferences with notes.
- Multi-stage similarity computation generates semantic embeddings for the query and movie corpus, then calculates TF-IDF similarity, sentiment alignment, and popularity normalization.
- Final ranking selects top recommendations based on weighted scores.

To further improve the user experience, we integrated subtitle analysis using the facebook/bart-large-cnn model. The subtitles of movies were sourced from OpenSubtitles API[1]., pre-processed and run through the summary model to create an introductory summary of each movie. The summary model only processes the beginning of the subtitles so that it gives users an insight into the movie without spoilers. In addition, KeyBERT was used to extract three key themes from the subtitles and present them to the user.

This system is well-suited for streaming platform recommendation engines, personalized content curation, movie discovery and educational purposes. Due to it's modular design (frontend / backend strictly separated) it can be easily integrated into existing systems. The frontend is optionally deployable as the backend can be integrated into any existing system due to interfacing using a easy-to-understand REST API.

[1]OpenSubtitles. December 2024. url: https://www.opensubtitles.org

# C Experiments and Results

To evaluate the performance, feasibility, and efficiency of our system, a series of experiments were conducted. A standardized questionnaire was employed to assess the system's performance. Each group member independently evaluated the system. This evaluation process was iterated over three days to identify a movie for each evening's viewing. The individual evaluation reports can be found in the questionnaire folder within the project repository. The evaluation results of each group member were averaged to provide a comprehensive overview of the system's performance. In general the the system received positive feedback from all evaluators with a 1.75 (1 = very good, 5 = very bad) on average. The user interface was rated at a 1.02 (1 = very good, 5 = very bad) on average. The recommendation quality was rated at 1.04 (1 = very good, 5 = very bad) on average. The response time was measured at 24.59 seconds on average and the system showed results with a confidence of 65.04% on average. The subjective comments by the evaluators mostly praised the system for the accurarcy of recommendations given their input, but also the added value through the AI-generated introduction. On the other hand, the evaluators also mentioned that the system could show more results, had some issues with some AI-generated introductions and that the system could be faster.
The second part of the evaluation was to objectively and quantifiably evaluate the system through a evaluation script. This script measured different metrics like response time, confidence score and genre distribution. Aftwards a variety of metrics was calculated to evaluate the system. The evaluation script also measured statistical metrics like precision, recall and F1-score. We are going to use the rest of this section to discuss the most relevant metrics and results of the evaluation. The evaluation script was called with the following parameters:

- use randomly sampled 40% of the full dataset for evaluation which yields approximately 583k movies (data rows)
- sample 600 testcases from all possible input combinations and run each testcase 2 times, which results in 1200 test runs

The script was executed on a MacBook Pro 14" with a M3 Pro chip and 18GB of RAM in a Python 3.12.6 environment. In the following figure we can see the distribution of response times of the recommendation system.

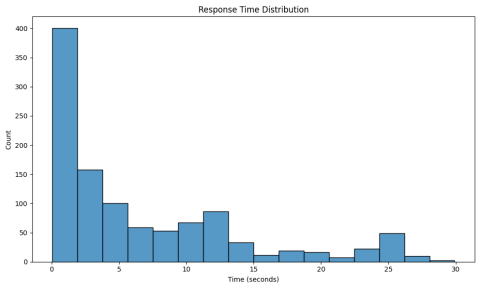


Figure 1: Response time distribution of 1200 test runs on approx. 583k rows (in seconds)

This figure clearly shows that most of the recomendations are processed within at most 10 seconds. This is a reasonable response time given the complexity of the system and the used hardware. Nonetheless there are some outliers which have taken more than 15 or even 25 seconds to provide results. This is also validated when looking at the raw data, where the average response time is 6.69 seconds. The median on the other hand is 3.60 seconds which is a good indicator that the average is skewed by outliers. The high variability in response times can also be validated by looking at the standard deviation of 7.32 seconds.

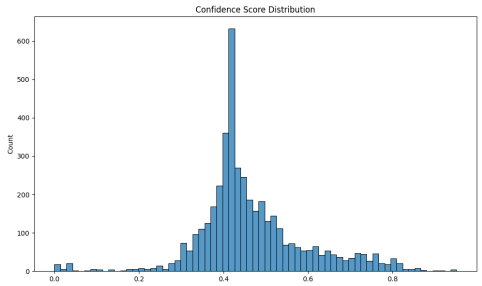


Figure 2: Confidence score distribution of 1200 test runs on approx. 583k rows (in %)

The figure shows that the system has an average confidence score of 46.55% which could be better. Nonetheless, the subjective evaluation through the questionnaire showed that the recommendations at approximately 45% confidence score were still deemed as valuable. In the last figure we can see the distribution of genre distribution of the recommendation system visualized in a histogram.

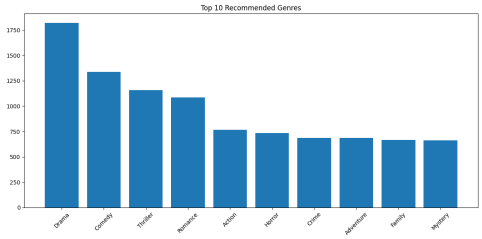


Figure 3: Genre distribution histogram of 1200 test runs on approx. 583k rows (in no. recommendations)

The figure clearly shows that the system is able to recommend movies from a wide variety of genres. Some are recommended more often than others, but the system does not show a clear bias towards a specific genre.
If we take a look at the statistical metrics we can see that the system performs quite well. The precision of the system is 99.08% which indicates that the system is very good at recommending movies that the user likes. The recall of the system is 10.10% which indicates that the system might be missing out on some movies that the user likes. The low recall may be explainable due to us sampling a small number of movies after pre-filtering to add variety in the recommendations. The F1-score of the system is 14.38% which is relatively low, but was expected due to drag-down caused by the low recall.
The mentioned values, complete results as well as other plots can be found in the evaluation folder within the project repository.

# D Conclusion

In the project we managed to produce highly usable and efficient movie recommendations. Nonetheless, there is still a lot of tweaking and fine-tuning to be done. Minor changes can lead to significant improvements in the user experience as we already noticed during development.