SI650/EECS549 Project Report

Jia Zhu
zjjj@umich.edu

Haoyang Ling
hyfrankl@umich.edu

December 5, 2023

1 Introduction

In the modern entertainment landscape, finding the perfect movie across numerous streaming platforms has become a challenging task. Current recommendation systems often fall short, relying on generic algorithms or user ratings that overlook individual preferences. Our project addresses this challenge by redefining the movie discovery experience. Unlike traditional approaches, our innovative pipeline combines exact matching with semantic understanding and refined re-ranking. This approach not only achieves a notable increase of 0.11 in nDCG@10 over the baseline BM25 but also aligns more closely with user intent and content relevance. Our detailed ablation study further highlights the efficacy of refined re-ranking and feature extraction in this context. We also discuss and address ethical considerations in genre shift and fairness. Additionally, we emphasize a seamless user experience through the design of an intuitive front-end web page. Through rigorous evaluation, our project identifies the strengths and weaknesses of our approach, paving the way for future advancements in movie retrieval systems. Our goal is to make discovering the perfect movie as enjoyable as the viewing experience itself.

2 Data

Table 1 shows the basic information of the dataset we have now. The data sources, preprocessing steps, and relevance data will be explained below.

Number of data	86537
Number of query-doc pair	2759

Table 1: Data Information

2.1 Data Resource

As mentioned above, the IR task of the project is to return a list of movies that are closely related to the user's query and are sorted. Therefore, our collection will be a large number of movie titles and information related to them.

Primary data for now was obtained from MovieLens Latest Datasets on GroupLens, which includes about 86,000 movies[8]. This dataset contains descriptions of the genres of movies(Fig. 1), as well as labels for many users' impressions of the movies(Fig. 2).

In addition, the dataset contains the tmdbID of movies(Fig. 3). So we get the overview for each movie in the dataset through the TMDB API(Fig. 4).

movieId	title	genres
1	Toy Story (1995)	Adventure | Animation | Children | Comedy | Fantasy
2	Jumanji (1995)	Adventure | Children | Fantasy
3	Grumpier Old Men (1995)	Comedy | Romance
4	Waiting to Exhale (1995)	Comedy | Drama | Romance
5	Father of the Bride Part II (1995)	Comedy
6	Heat (1995)	Action | Crime | Thriller
7	Sabrina (1995)	Comedy | Romance
8	Tom and Huck (1995)	Adventure | Children
9	Sudden Death (1995)	Action
10	GoldenEye (1995)	Action | Adventure | Thriller

Figure 1: movie title and genres

userId	movieId	tag	timestamp
10	260	good vs evil	1430666558
10	260	Harrison Ford	1430666505
10	260	sci-fi	1430666538
14	1221	Al Pacino	1311600756
14	1221	mafia	1311600746
14	58559	Atmospheric	1311530439
14	58559	Batman	1311530391
14	58559	comic book	1311530398
14	58559	dark	1311530428

Figure 2: movie tag

2.2 Data Preprocessing

Based on all the data we got, we finally decided to use movieID as the index and merge overview, genres and tags of each movie given by the users as the final dataset. After that, we performed the basic processing of separating words and removing stopwords.

2.3 Relevance Data

For correlation scores, we did not find relevant data. Therefore, we generated about 50 query by ourselves. The types of these queries are very rich. Some describe movie types, such as "Documentaries exploring environmental issues"; some describe movie scenes, such as "Movies with epic space battles"; and there are also keyword types, such as "Vampire, werewolf, and love". In addition, there are also some more complex ones, such as "Science fiction movies with futuristic cityscapes" and "Historical dramas with powerful courtroom scenes" that combine movie types and scenes.

For all queries, we first use BM25 model to sort all possible document in our index. Then manually label the first 50-100 documents obtained by each query with relevance scores. We follow a 5-point relevance scale for labeling, with 1 indicating not at all relevant and 5 indicating very relevant.

3 Related Work

Model information retrieval system aims to deliver customized and precise results to user queries. Its process can be divided into two key stages: retrieval and ranking shown in Fig. 5. [7].

movieId	imdbId	tmdbId
1	114709	862
2	113497	8844
3	113228	15602
4	114885	31357
5	113041	11862
6	113277	949
7	114319	11860
8	112302	45325
9	114576	9091

Figure 3: movie tmdbID

id	tmdbId	overview
1	862	Led by Woody, Andy's toys live happily in his room until Andy's b
2	8844	When siblings Judy and Peter discover an enchanted board game
3	15602	A family wedding reignites the ancient feud between next-door n
4	31357	Cheated on, mistreated and stepped on, the women are holding t
5	11862	Just when George Banks has recovered from his daughter's weddi
6	949	Obsessive master thief Neil McCauley leads a top-notch crew on
7	11860	An ugly duckling having undergone a remarkable change, still ha
8	45325	A mischievous young boy, Tom Sawyer, witnesses a murder by the
9	9091	When a man's daughter is suddenly taken during a championship
10	710	When a powerful satellite system falls into the hands of Alec Tre

Figure 4: movie overview

In the initial phase of document retrieval, the objective centers on pinpointing a preliminary collection of documents that appear relevant to the user's query. This process employs various models such as the vector space model [18], Latent Dirichlet Allocation (LDA) [3], and BERT-based models [6]. Text information can be encoded and represented in two primary ways: sparse and dense representations. Traditional Information Retrieval (IR) models frequently utilize sparse representations, which indicate the presence or absence of terms, exemplified by the "bag-of-words" approach. In contrast, dense representations, known as word embeddings, assign continuous vector spaces to words, as seen in models like Doc2vec and BERT. Research by Google has revealed that sparse retrieval models are particularly adept at identifying precise term overlaps. Conversely, learned dual encoders, employing dense representations, demonstrate a superior ability to capture semantic similarities [12]. Notably, it has been observed that the BM25 model surpasses a dual encoder based on BERT in certain contexts [17]. Consequently, a Google researcher has suggested a hybrid multi-vector encoding model as a promising solution [12].

During the ranking phase, the foremost aim is to reorder the initially retrieved documents according to their relevance. This stage emphasizes the effectiveness of results over efficiency, contrasting with the retrieval stage. While models like BM25 are typically used in initial retrieval for efficient processing of a vast document corpus, ranking models, including RankNet and DRMM, employ a variety of methods such as reinforcement learning, contextual embeddings, and attention mechanisms. These techniques are utilized to optimize document ranking based on user-specific criteria, thereby enhancing result quality [4, 7].

Concerning the movie Information Retrieval (IR) system, several approaches have been explored. Kurihara et al. focused on the retrieval of short sentences from user-generated con-

rel	frequency
1	0.359188
2	0.134469
3	0.158753
4	0.185212
5	0.162378

Table 2: Relevance score distribution

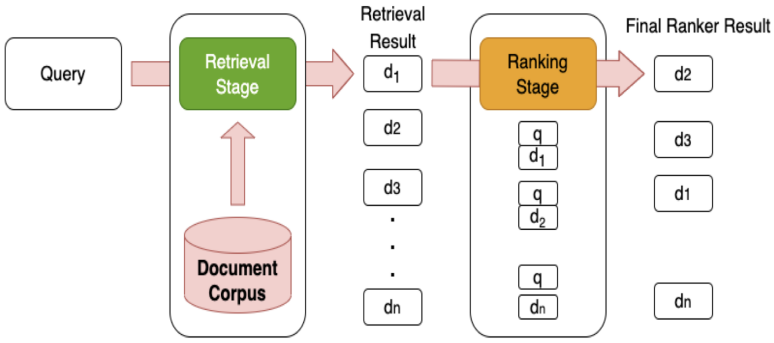

Figure 5: Overview of modern Information Retrieval system [7]

tent, improving the accuracy of Doc2Vec models through the integration of target-topic IDs during training [11]. This approach, however, is primarily concerned with short sentence retrieval and does not directly align with our objective of retrieving scene-based, dialog-centric, or keyword-specific content. Bain et al. introduced the Condensed Movie Dataset (CMD), aiming to understand movie narratives via key scenes, and established a deep network baseline for text-to-video retrieval [2]. Nonetheless, our system concentrates on textual data and does not incorporate visual information. In the realm of recommendation systems, Almuhaimeed et al. presented a hybrid movie recommendation methodology that leverages semantic relations and concealed associations linked to movies, including descriptions and metadata. This approach excels in aligning movie recommendations with user preferences [1]. Xu et al. proposed a generalized neural information retrieval framework that separates signal capturing and combination, highlighting the significance of exact matching, semantic matching, and inference matching in this context. However, it is noted that these descriptions often omit certain details, posing a significant challenge to the task at hand [19]. Chen et al. proposed a novel contrastive learning approach with movie metadata to learn a general-purpose scene representation [5]. They concluded that genre and synopsis were all effective measures in guiding movie scene representation learning, which is valuable for us. However, our project focuses on the genre and synopsis will be the main information used in the retrieval part.

With the rise of Large Language Models, Pradeep et. al. introduced RankVicuna, an open-source LLM designed for high-quality listwise reranking in a zero-shot context [15]. In tests using the TREC 2019 and 2020 Deep Learning Tracks, RankVicuna showed comparable effectiveness to GPT-3.5 in zero-shot reranking. Based on the computing resources limitations, we will try to improve the pipeline with the modern LLM in the future, but we will try to compare our pipeline with the current state-of-the-art LLM models.

The innovation of our approach is distinctly evident in its specialized focus on the chal-

lenge of recalling specific movie titles based on fragmented memory clips or partial information. Divergent from other movie search methodologies that rely on multimedia data, our system is specifically designed to process user queries that are solely text-based, pertaining to brief snippets or impressions of movies. This approach offers a more efficient and user-centric avenue for discovering movies based on incomplete information or memory fragments. Moreover, there is potential for this system to enhance its capabilities through integration with complementary visual models, thus expanding into a multi-model domain.

4 Methodology

4.1 Basic setups

We used the Python library to build up our retrieval system, including:

- PyTerrier [13]: indexing with stopwords and query expansion.
- Sentence transformer [16]: 'msmarco-distilbert-dot-v5' model for semantic similarity.
- lightgbm [10]: LGBMRanker.

4.2 Challenges and Features

In our previous discussions, we highlighted the development of a movie retrieval system designed to process both precise and abstract movie queries. These queries can range from specific genres like 'space battles' or 'vampire and werewolf romances' to more nebulous themes such as 'soldier camaraderie' or 'films that embody the essence of youth'. Our system employs a combination of sparse and dense representations to manage a diverse range of queries, from specific genres to nebulous themes. This approach enhances the system's ability to balance and weigh these representations, thereby improving performance.

A notable challenge we face is working with limited data, such as brief movie synopses, genres, user comments, and tags. Relying solely on exact matches in movie descriptions could lead to unstable performance across various queries. To mitigate this, we incorporate statistical analysis of movie genres and tags, which should enrich our database with more relevant information.

To address these challenges, we've integrated several key features into our retrieval system:

- **Movie Descriptions**: We utilized basic statistical methods like term frequency, TF-IDF, and BM25. Additionally, we transformed these descriptions into vector representations using the 'msmarco-distilbert-dot-v5' model, renowned for its effectiveness in semantic search and popularity on Huggingface [16]. It helped us to calculate the cosine similarity of the descriptions and queries.
- **Movie Titles**: We applied term frequency and TF-IDF analysis. We avoided using BM25 here, as its complexity doesn't suit the typically short nature of movie titles.
- **Categories**: We encoded these in four distinct ways: one-hot encoding, total term frequency, a probability metric indicating category preference, and a maximum category preference probability. This multifaceted approach captured both exact and semantic matches, as well as local and global summary information. We calculated the semantic similarity using the BERT model to convert them into vectors and then calculated the cosine similarity. The choice to use the maximum category preference probability is to introduce some non-linearity with the same idea of max pooling.

- **Tags**: Tags like 'dystopia' or 'powerful ending' often reflect the themes users are searching for. To leverage this, we appended these tags to the movie descriptions. It is especially beneficial for movies with shorter descriptions.

4.3 Pipeline

We create a retrieval pipeline based on BM25. Then, we consider re-ranking the retrieved results with our Learning-to-rank (L2R) with LGBMRanker and DirichletLM. We design the pipeline in this manner because we need to filter out the most 25 relevant movies with exact matching (BM25), then find the top 15 movies with semantic matching (L2R), and finally use the exact matching (DirichletLM) to find the top 10 results shown in Fig. 6. We will do the ablation study on this to see how each part would improve the performance.

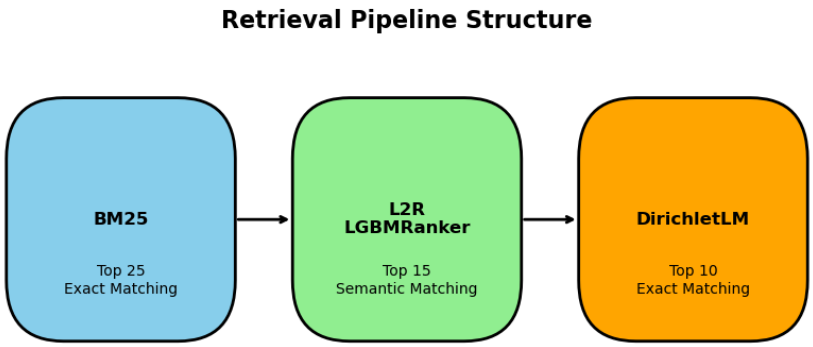

Figure 6: Model pipeline

4.4 Baselines

Through these enhancements, our movie retrieval system is poised to deliver more accurate and relevant results, catering to a wide array of user queries. And, since we create the dataset with diverse queries and pose the questions on exact and semantic searching, we define two baselines for our movie information retrieval system based on the IR task definition given before To provide context for our Movie Retrieval System's performance, we will compare it against two baselines. We denote the first two as the simple baselines and the last two as the challenging baselines.

- **Random Baseline:** This baseline will randomly select movie titles from the dataset that owns some common words with the query and present them as results. It serves as a minimal benchmark for system performance.

- **BM25 Baseline:** The BM25 information retrieval model with default hyper-parameters of BM25 in PyTerrier as the baseline. The movie title and description will be served as the model input. Tbl. 3 shows some sample queries from our dataset. For query 1, the ideal answer is Star Wars. For query 2, the ideal answer is The Hunger Games or Seven Sisters (2017). Cloud Atlas makes some sense because it contains part of the girls' rebellion plots. So, we can observe that BM25 provides some ambiguous answers here. It may be because it only captures the exact matching rather than semantic matching and inference matching.

- **Semantic similarity model (SSM):** Since we use one of the most popular semantic similarity models on the Huggingface "msmarco-distilbert-dot-v5" for Learning-to-Rank pipeline, we will use the same model with direct vector embedding similarity for comparison.
- **LLM:** With the rise of LLM applications in Retrieved-augmented generation (RAG), we are also curious about the capability of LLM in re-ranking the results with its knowledge. We will use the most powerful LLM models: (1) open-sourced: Mistral-7b [9]; (2) close-sourced: GPT-4 [14]. Specificly, we mock the prompt in Pradeep et. al. work on RankVicuna shown below [15].

Query 1: Movies with epic space battles.
Zack Snyder's Justice League (2021) Messenger: The Story of Joan of Arc, The (1999) The Hobbit: The Battle of the Five Armies (2014) Star Wars: Episode IV - A New Hope (1977) Troy (2004)
Query 2: Dystopian future with young girl leading a rebellion.
Giver, The (2014) Cloud Atlas (2012) Tides (2021) Embers (2015) Eden Log (2007)

Table 3: BM25 query examples

LLM promp input: num, query, movie_infos
I will provide you with num movies, each indicated by a numerical identifier []. Rank the movies based on their relevance to the search query: query. Here are the movies that you need to rank: {movie_infos} Please rank the {num} movies above based on their relevance to the search query. All the movies should be included and listed using identifiers, in descending order of relevance. The output format should be [] > [] > ... > [], e.g., [4] > [2] > [3] > [1] > [5] if 5 movies are given. Only respond with the ranking results, do not say any word or explain the reason for the ranking and keep the answer as short as possible.

Table 4: LLM prompt template

5 Evaluation and Results

5.1 Metrics and Evaluation

To evaluate the effectiveness of our Movie Retrieval System, we use the following quantitative evaluation metrics:

- **Mean Average Precision (mAP):** mAP can provide a comprehensive evaluation of the system's ability to rank relevant movie titles higher in the list of results. As we label the movie with a 5-point scale, we select movies with a score of 3 and larger as the relevant movie.
- **Normalized Discounted Cumulative Gain (nDCG):** nDCG will assess the quality of the ranked list of movie titles. It considers the relevance of retrieved movies and their positions in the list, assigning higher scores to more relevant movie titles.

For convenience, we use the mAP(rel=3)@10 and nDCG@10 as the metrics in PyTerrir to evaluate our models. Additionally, to facilitate the evaluation of our Information Retrieval (IR) system using the mentioned metrics, we will generate test cases comprising user queries and their associated expected outcomes. These test cases will be meticulously annotated, taking into account the insights gained from search engine results retrieved from prominent platforms such as Google and Bing.

5.2 Simple Baseline Results

To provide context for our Movie Retrieval System's performance, we compare it against two baselines. We evaluate the two metrics on the test dataset with a cut-off of 10. The results are shown in Fig. 7.

- **Random Baseline:** mAP@10 is 0, and nDCG@10 is around 0.232.
- **BM25 Baseline:** mAP@10 is about 0.216, and nDCG@10 is around 0.688.

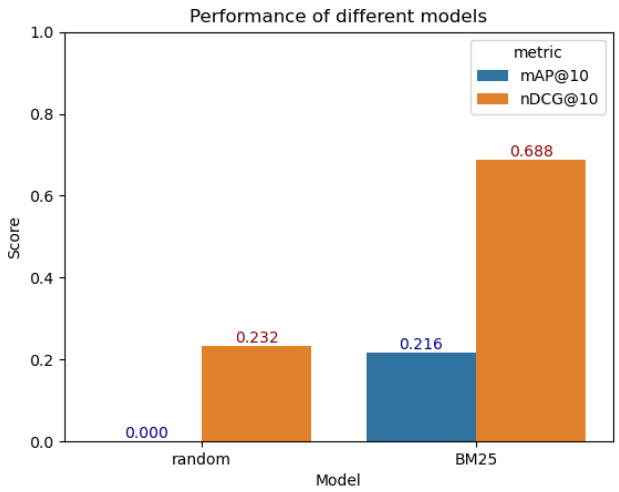

Figure 7: Baseline results on test dataset

Based on the retrieved samples in Tbl. 3, we can see that BM25 fails to capture semantic meaning and leads to a relatively low mAP value.

5.3 Our Model Performance

Based on the proposed pipeline, we test some related models on our test datasets. Detailed configurations are shown in Fig. 5. We also consider whether to extract semantic features like

categories and the probability metric indicating category preference. We will treat the details in the later part of the ablation study. So, based on these configurations, we use the training and validation data with early stop rounds 5 and the metric nDCG@20 to avoid over-fitting. The results are shown in Fig. 8 and Tbl. 6. The best model is LMART_L_FE_DLM. We also include the performance of the semantic similarity model (SSM) "msmarco-distilbert-dot-v5" for reference. From the experiments, it shows that our model is comparable to the LLM-based retrieval system. It is because we inject the specific domain knowledge in it, while Large Language Models try to find a balance in multiple natural language processing tasks. But still, we have less parameters than LLM, so that our model is less competitive in transfer learning.

Model	L2R re-ranking	DirichletLM re-ranking	Semantic features
BM25			
BM25_DLM		✓	
LMART_L	✓		
LMART_L_DLM	✓	✓	
LMART_L_FE	✓		✓
LMART_L_FE_DLM	✓	✓	✓

Table 5: Model configurations

Model	avg mAP@10	avg nDCG@10	std nDCG@10	min nDCG@10	max nDCG@10
BM25	0.216	0.688	0.166	0.426	0.865
DLM	0.165	0.643	0.170	0.403	0.880
BM25_DLM	0.264	0.768	0.128	0.627	0.947
LMART_L	0.305	0.717	0.072	0.604	0.784
LMART_L_DLM	0.294	0.789	0.112	0.690	0.974
LMART_L_FE	0.259	0.732	0.105	0.588	0.870
LMART_L_FE_DLM	0.304	0.796	0.115	0.704	0.987
SSM (DistilBERT)	/	0.251	0.067	0.200	0.375
Mistral 7b	/	0.790	0.113	0.660	0.989
GPT 4	/	0.805	0.105	0.686	0.988

Table 6: Model comparison of nDCG@10 metric on the test dataset (the best model in each column is highlighted in red)

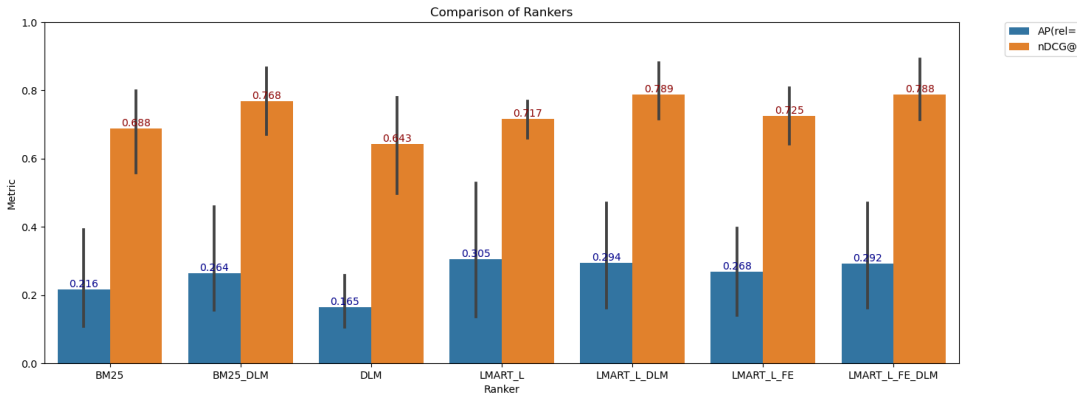

Figure 8: Comparison of model performances on test dataset

5.4 Ablation study

5.4.1 Tag to Movie Description

To validate our hypothesis about the beneficial impact of adding tags to movie descriptions, we employed a variety of statistical models such as BM25, BM25 with Bo1 query expansion, TF-IDF, PL2, CM, and DirichletLM, using the PyTerrier framework. This experiment was conducted on the entire dataset to leverage a broader range of queries and to avoid the need for model training, thereby aiming for a more accurate evaluation.

The results, as detailed in Table 7, indicate a clear trend: incorporating tags into movie descriptions tends to enhance the performance of these models. This improvement can be attributed to the additional context and specificity that tags provide, enabling the models to more effectively match movies with user queries. Tags, by encapsulating key aspects of a movie, enrich the descriptions and make them more informative and searchable.

However, it's important to note that the extent of performance improvement varies across different models. This variability suggests that while tags generally contribute positively, their impact is not uniform and is influenced by the inherent characteristics and mechanisms of each model.

Model	nDCG@10 (w/o tag)	nDCG@10
BM25	0.517	0.533 ↑
BM25-QE	0.540	0.537 ↓
TF-IDF	0.522	0.535 ↑
PL2	0.502	0.502 –
DirichletLM	0.516	0.539 ↑
CM	0.221	0.221 –

Table 7: Tag's influence on basic rankers.

5.4.2 Feature extraction

In the previous section, we discussed the integration of various features into our models, such as query and movie description similarity, category preference probability, and one-hot encoding of categories. These were derived from dense features, topics, or semantic information extracted from movie descriptions. In this part of our discussion, we aim to evaluate the impact of these additional features on our model's performance. Results presented in Fig. 8 and Table 6 demonstrate that incorporating these extra features (such as in LMART_L_FE and LMART_L_FE_DLM models) generally enhances the average performance. However, this improvement is accompanied by increased variability, resulting in more unstable models.

The added complexity brought by these features seems to contribute to the models' inconsistency. While they provide a richer, more nuanced understanding of the data, which can be beneficial in certain contexts, they also introduce greater unpredictability in the model's behavior. This finding suggests a trade-off between the depth of data representation and model stability.

Moreover, Fig. 9 highlights the significant importance of semantic features. The 'dot_score', which represents the similarity between movie descriptions and queries, and 'category_similarity', which reflects the maximum category preference probability, are among the most influential features. The presence of one-hot encoded genres in this context further underscores the value of semantic information in improving average model performance. These observa-

tions collectively affirm that while semantic features indeed enhance model accuracy, they also require careful handling to maintain model stability.

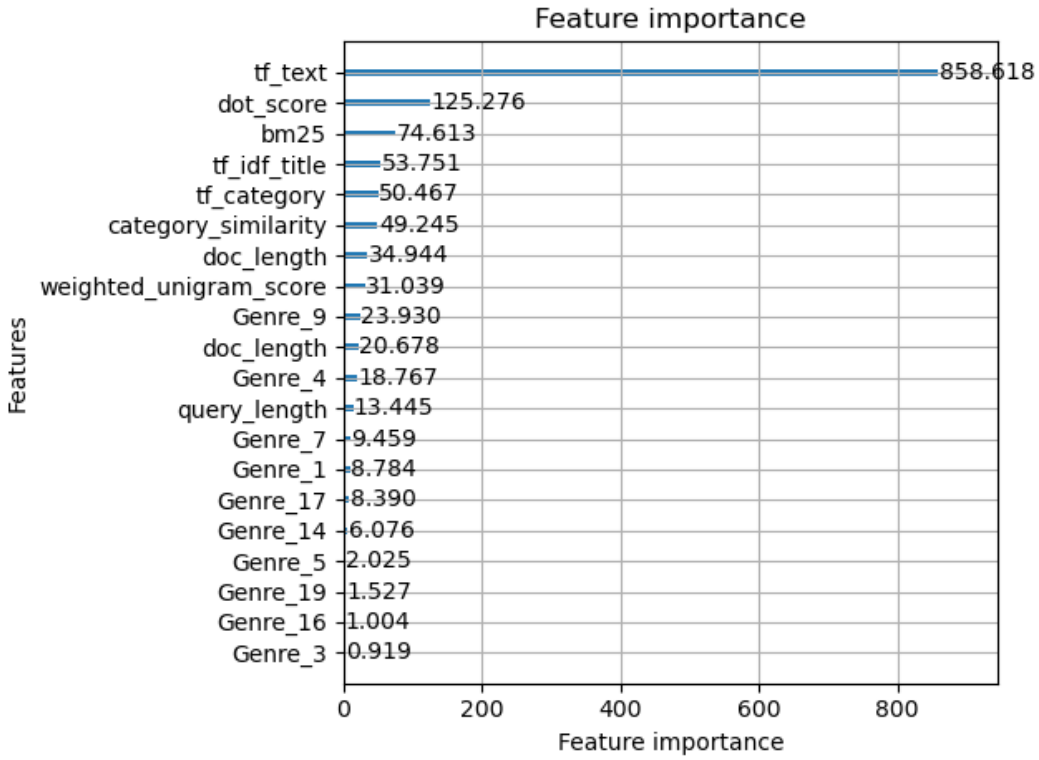

Figure 9: Feature importance of LGBMRanker in LMART_L_FE

5.4.3 DirichletLM re-ranking

DirichletLM, a language model derived from specific word distributions, demonstrates superior performance over basic models, as shown in Table 7. Motivated by these findings, we are exploring the re-ranking of our results using DirichletLM to potentially enhance performance. Evidence from Fig. 8 and Table 6 confirms that DirichletLM consistently betters the baseline models. Although there are instances where DirichletLM's performance is not optimal, its integration in the re-ranking process generally leads to significant improvements. This observation suggests that DirichletLM's strengths are more pronounced in a refined, post-filtering context rather than in initial broader matches.

5.5 Front-end Page

In addition to back-end research, we also successfully designed a front-end web page (Fig. 10. It is an intuitive movie search platform with an eye-catching search box in the center of the page. Users can easily start a search simply by entering their query and clicking the beautifully designed search button. Once the search results are executed, they are displayed elegantly, displaying the movie title and a concise synopsis. Notably, the hyperlinked title transitions seamlessly to the TMDb page for more detailed information, while interactive color changes on hover add a user-friendly touch. Additionally, a continuous scrolling mechanism enhances the browsing experience, automatically loading the next set of movies as the user scrolls to the bottom of the page.

Figure 10: Front-end Page

6 Discussion

We sampled some queries from our dataset and an extra query – "AI future" to observe the performance of the proposed rankers shown in Tbl. 8. For most queries, we got really good top 3 retrieved results. Especially for queries 1 and 3, we can compare them with the retrieved results from BM25 in Tbl. 3. Although it doesn't do well in queries 6 and 7, the better performance convinced us that the proposed models learned something from the dataset.

In Table 9, we presented genre-based queries to investigate whether one genre might overshadow another in our results. For the first query, we observed a well-balanced blend of romance and fantasy in the majority of the listed movies, such as "Stardust," "Princess Bride," "Howl's Moving Castle," and "Harry Potter and the Half-Blood Prince." The second query, primarily focusing on action and fantasy, interestingly showed a subtle shift towards adventure and fantasy. This transition is understandable given the overlapping characteristics between action and adventure genres. The third query, however, displayed a slight inclination towards action over romance, indicating a potential imbalance in genre representation.

As previously shown, the proposed models demonstrate a marked improvement in retrieval accuracy compared to the established baselines based on the metrics mAP@10 and nDCG@10. Specifically, the LMART_L_FE_DLM model exhibited the highest performance with mAP@10 0.304 and nDCG@10 0.796 as indicated by the red highlights in Table 6. This superior performance can be attributed to the model's ability to effectively leverage semantic features and re-rank results using DirichletLM. The enhancement in performance underscores several points:

- **Tag to Movie Description**: the integration of movie tags with descriptions, as demonstrated in Table 7, significantly boosted our model's performance. These tags add crucial context, enriching the descriptions and leading to more precise query matching.

Query 1: Movies with epic space battles. (LMART_L_FE)
Star Wars: Episode IV - A New Hope (1977) Lord of the Rings: The Two Towers, The (2002) Star Wars: Episode I - The Phantom Menace (1999)
Query 2: War films set during World War II. (LMART_L_FE)
Tora! Tora! Tora! (1970) Bridge on the River Kwai, The (1957) Saving Private Ryan (1998)
Query 3: Dystopian future with young girl leading a rebellion. (LMART_L_FE_DLM)
Cloud Atlas (2012) Giver, The (2014) Seven Sisters (2017)
Query 4: Horror movies with haunted house settings. (LMART_L_FE_DLM)
Conjuring, The (2013) Haunting, The (1963) Others, The (2001)
Query 5: AI future. (LMART_L_FE)
A.I. Artificial Intelligence (2001) I Am Mother (2019) Chappie (2015)
Query 6: Films exploring philosophical concepts of time. (LMART_L_FE_DLM)
Animatrix, The (2003) Lemonade (2016) Midnight in Paris (2011)
Query 7: Movies celebrating the enduring power of friendship. (LMART_L_FE_DLM)
28 Hotel Rooms (2012) Oil: A Symphony in Motion (1933) Lucy and Desi (2022)

Table 8: Query examples for the proposed models

It underscores the value of embedding context-specific metadata for enhanced data richness and search accuracy.

- **Feature extraction 1**: incorporating semantic understanding into retrieval systems marks a substantial advancement. It allows the system to interpret not just the literal terms of the queries but also their underlying meanings, bridging the gap between user intent and content relevance to enhance retrieval accuracy and user satisfaction.

- **Feature extraction 2**: our findings also shed light on the intricate balance between feature complexity and model stability. While integrating sophisticated features like category preference probability and one-hot encoding of categories indeed elevated the average performance, it simultaneously introduced an element of unpredictability with a large deviation in the nDCG metric.

- **DirichletLM re-ranking**: the performance of DirichletLM, particularly in re-ranking scenarios, indicates its suitability for more targeted applications with limited context. It excels in the refinement of the initial broader selection. This finding opens up avenues for further research, particularly in optimizing the balance between initial retrieval and subsequent re-ranking. The potential to dynamically adjust the role of DirichletLM

Query 1: romance and fantasy	
Stardust (2007)	Adventure | Comedy | Fantasy | Romance
Princess Bride, The (1987)	Action | Adventure | Comedy | Fantasy | Romance
Howl's Moving Castle (Hauru no ugoku shiro) (2004)	Adventure | Animation | Fantasy | Romance
Harry Potter and the Half-Blood Prince (2009)	Adventure | Fantasy | Mystery | Romance | IMAX
Penelope (2006)	Comedy | Fantasy | Romance
The Hobbit: The Battle of the Five Armies (2014)	Adventure | Fantasy
Legend (1985)	Adventure | Fantasy | Romance
Ladyhawke (1985)	Adventure | Fantasy | Romance
Red Riding Hood (2011)	Fantasy | Horror | Mystery | Thriller
Don Juan DeMarco (1995)	Comedy | Drama | Romance
Query 2: fantasy and action	
Lord of the Rings: The Two Towers, The (2002)	Adventure | Fantasy
Lord of the Rings: The Fellowship of the Ring, The (2001)	Adventure | Fantasy
Pirates of the Caribbean: The Curse of the Black Pearl (2003)	Action | Adventure | Comedy | Fantasy
Hobbit: The Desolation of Smaug, The (2013)	Adventure | Fantasy | IMAX
How to Train Your Dragon (2010)	Adventure | Animation | Children | Fantasy | IMAX
Bright (2017)	Action | Crime | Fantasy
Prince of Persia: The Sands of Time (2010)	Action | Adventure | Fantasy | Romance | IMAX
Clash of the Titans (2010)	Action | Adventure | Drama | Fantasy
Wrath of the Titans (2012)	Action | Adventure | Fantasy | IMAX
Jungle Cruise (2021)	Adventure | Children | Comedy | Fantasy
Query 3: romance and action	
Jack Reacher (2012)	Action | Crime | Thriller
Baby Driver (2017)	Action | Crime | Thriller
Monsters (2010)	Drama | Sci-Fi
Femme Nikita, La (Nikita) (1990)	Action | Crime | Romance | Thriller
Bodyguard, The (1992)	Drama | Romance | Thriller
Red Cliff Part II (Chi Bi Xia: Jue Zhan Tian Xia) (2009)	Action | Drama | War
Three Kingdoms: Resurrection of the Dragon (2008)	Action | Drama | War
Dhoom 2 (2006)	Action | Crime | Drama | Thriller
Marrying the Mafia II (2005)	Action | Comedy | Crime | Romance
My Love Story!! (2015)	Comedy | Romance

Table 9: Genre query examples for the proposed models: (1) red means biased results; (2) cyan identifies genre shift; (3) green represents unbiased ones.

based on the characteristics of the dataset and the specific needs of the retrieval task offers a promising direction for future developments.

- **Potential weakness**: they still fall short in accurately capturing and responding to certain thematic or abstract queries. This limitation is particularly evident in scenarios where the query requires a nuanced understanding of thematic elements or underlying narrative tones, rather than straightforward keyword or genre-based retrieval. Vague or highly abstract queries require a level of interpretation and understanding that the models may not be equipped to handle. It is the limitation of this work.

- **Potential bias and fair genre representation**: ensuring that both genres are fairly represented is crucial for meeting user expectations and providing a diverse range of content. But, we have observed that action films are preferred in the third query of Tbl. 9. If one genre consistently dominates over the other in query results where both genres are specified, it may indicate a bias in the retrieval algorithm. This can lead to a misrepresentation of content and limit the diversity of recommendations.

.

7 Conclusion

In conclusion, our project aimed to revolutionize the movie discovery experience by developing an advanced movie retrieval system that effectively processes both precise and abstract queries. Leveraging a combination of sparse and dense representations, semantic similarity models, and innovative feature engineering, our system demonstrated significant improvements over traditional recommendation approaches. The integration of movie tags, semantic features, and DirichletLM re-ranking played pivotal roles in enhancing the accuracy and relevance of movie recommendations. Our proposed LMART_L_FE_DLM model emerged as the top performer, achieving the highest scores in metrics such as mAP@10 and nDCG@10. While our system excelled in many aspects, there are still limitations in handling nuanced or abstract queries and retrieving unbiased results, representing an area for future refinement.

8 Other Things We Tried

To create an effective movie retrieval system, we faced numerous challenges and tested various methodologies. Our experiments included using the Porter stemmer for indexing, contrasting it with non-stemmed indexing, only to find potential risks of collapsing the Learn-to-Rank (L2R) system. We thoroughly examined different re-ranking setups, selecting the top 25, 50, and 100 results from BM25, ultimately determining that the top 25 yielded the most optimal performance, likely due to an estimated 25 suitable matches per query in our training dataset. Despite exploring DirichletLM, its performance was not as effective as anticipated, suggesting the need for a reasonably effective (but general) first-stage ranker in the L2R system.

Additionally, we ventured into enhancing the system by incorporating features like the mean/summation of category preference probability and the summation of one-hot encoding of categories. This approach aimed to deepen the system's grasp of user preferences in genres. However, this increased complexity also introduced unpredictability and varied performance levels, underscoring the intricacies of feature engineering. Further, we explored the potential of DirichletLM in re-ranking, which, despite its strengths in refining results, showed that its optimal efficacy might be contingent on specific contexts. These comprehensive experiments and explorations significantly contribute to our understanding of the nuanced journey in developing this movie retrieval system.

9 What You Would Have Done Differently or Next

Reflecting on the project, there are several aspects we would consider for future iterations. In envisioning the future trajectory of this project, there are several avenues that future researchers and practitioners could explore:

- Refinement of Semantic Understanding: Future work could focus on refining the semantic understanding of movie descriptions and queries augmenting the dataset with the Wikipedia dataset and cast & crew. Investigating more sophisticated natural language processing models and embeddings could enhance the system's ability to grasp nuanced themes and abstract concepts, ultimately improving performance on diverse queries.
- Dynamic Feature Engineering: The exploration of dynamic feature engineering strategies could be valuable. Researchers might experiment with adaptive feature sets, adjusting the complexity of features based on the characteristics of the queries or the

user's interaction patterns. This could contribute to a more flexible and adaptive recommendation system.

- User Feedback Integration: Integrating user feedback mechanisms into the system could be a promising avenue. Future research might explore incorporating user preferences, ratings, and feedback to iteratively update the recommendation models. This iterative learning process could lead to a more personalized and user-centric movie retrieval experience.
- Real-World Deployment and Evaluation: The deployment of the movie retrieval system in real-world scenarios offers an exciting direction for future exploration. Conducting comprehensive evaluations with real users, gathering feedback, and analyzing user interactions could provide valuable insights into the practical utility and user satisfaction of the system.

By addressing these aspects, future researchers can build upon the foundation laid by this project and contribute to the ongoing evolution of movie retrieval systems.

10 Team Work Distribution

Jia Zhu:

- Led the data collection efforts with a focus on thoroughness and completeness.
- Enriched the project's database through API integration and various tools.
- Implemented the baseline, establishing a solid foundation for the project.
- Demonstrated technical proficiency in developing front-end pages for an intuitive user interface.

Haoyang Ling:

- Collaborated on data collection and baseline implementation, ensuring a cohesive team approach.
- Explored and implemented a diverse array of new models, showcasing adaptability and innovation.
- Conducted a comprehensive comparative analysis of various models, providing valuable insights.
- Played a key role in shaping the project's final approach through meticulous evaluation of model strengths and weaknesses.

References

[1] Abdullah Almuhaimeed and Maria Fasli. "A hybrid semantic method for enhancing movie recommendations". In: *2017 International Conference on the Frontiers and Advances in Data Science (FADS)*. Oct. 2017, pp. 23–28. DOI: 10.1109/FADS.2017.8253188.

[2] Max Bain et al. "Condensed Movies: Story Based Retrieval with Contextual Embeddings". In: *Proceedings of the Asian Conference on Computer Vision (ACCV)*. Nov. 2020.

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. "Latent Dirichlet Allocation". In: *J. Mach. Learn. Res.* 3.null (Mar. 2003), pp. 993–1022. ISSN: 1532-4435.

[4] Christopher Burges. "From ranknet to lambdarank to lambdamart: An overview". In: *Learning* 11 (Jan. 2010).

[5] Shixing Chen et al. "Movies2Scenes: Using movie metadata to learn scene representation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 6535–6544.

[6] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[7] Kailash A. Hambarde and Hugo Proenca. *Information Retrieval: Recent Advances and Beyond*. 2023. arXiv: 2301.08801 [cs.IR].

[8] F. Maxwell Harper and Joseph A. Konstan. "The MovieLens Datasets: History and Context". In: *ACM Trans. Interact. Intell. Syst.* 5.4 (Dec. 2015). ISSN: 2160-6455. DOI: 10.1145/2827872. URL: https://doi.org/10.1145/2827872.

[9] Albert Q Jiang et al. "Mistral 7B". In: *arXiv preprint arXiv:2310.06825* (2023).

[10] Guolin Ke et al. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 3149–3157. ISBN: 9781510860964.

[11] Kosuke Kurihara et al. "Target-Topic Aware Doc2Vec for Short Sentence Retrieval from User Generated Content". In: *Proceedings of the 21st International Conference on Information Integration and Web-Based Applications & Services*. iiWAS2019. Munich, Germany: Association for Computing Machinery, 2020, pp. 463–467. ISBN: 9781450371797. DOI: 10.1145/3366030.3366126. URL: https://doi.org/10.1145/3366030.3366126.

[12] Yi Luan et al. "Sparse, Dense, and Attentional Representations for Text Retrieval". In: *Transactions of the Association for Computational Linguistics* 9 (Apr. 2021), pp. 329–345. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00369. eprint: https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00369/1924040/tacl_a_00369.pdf. URL: https://doi.org/10.1162/tacl%5C_a%5C_00369.

[13] Craig Macdonald and Nicola Tonellotto. "Declarative Experimentation inInformation Retrieval using PyTerrier". In: *Proceedings of ICTIR 2020*. 2020.

[14] OpenAI. "GPT-4 Technical Report". In: *arXiv preprint arXiv:2303.08774* (2023).

[15] Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. "RankVicuna: Zero-shot listwise document reranking with open-source large language models". In: *arXiv preprint arXiv:2309.15088* (2023).

[16] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: http://arxiv.org/abs/1908.10084.

[17] Stephen Robertson, Hugo Zaragoza, et al. "The probabilistic relevance framework: BM25 and beyond". In: *Foundations and Trends® in Information Retrieval* 3.4 (2009), pp. 333–389.

[18] Gerard Salton, Anita Wong, and Chung-Shu Yang. "A vector space model for automatic indexing". In: *Commun. ACM* 18 (1975), pp. 613–620. URL: https://api.semanticscholar.org/CorpusID:6473756.

[19] Shicheng Xu et al. “NIR-Prompt: A Multi-task Generalized Neural Information Retrieval Training Framework”. In: *arXiv preprint arXiv:2212.00229* (2022).