Exploring Content-Based Movie Recommendation Systems: A Comparative Study

Lin Cui
Department of Statistical and Actuarial Sciences
Western University
London, Canada
lcui28@uwo.ca

Yaoxin Liu
Department of Statistical and Actuarial Sciences
Western University
London, Canada
yliu4685@uwo.ca

***Abstract*—The increasing number of available movies grows the demand and popularity for personalized movie recommendations, based on the contents of the movies in users' rating records. This project presents an in-depth examination of content-based movie recommendation systems, with a specific focus on similarity-based methods and Behavior Sequence Transformer (BST) models, a more sophisticated Transformer-based model, capable of capturing intricate temporal dynamics and context from users' behaviour sequences.**

The results showed that the similarity-based approach demonstrated simplicity and interpretability, offering a robust solution when the availability of sequential user interaction data was limited. However, it suffered from issues of data sparsity and the lack of users' preferences. On the other hand, the BST model provided superior personalization by leveraging temporal users' sequential behaviours, though at the cost of increased complexity and potential overfitting.

Our study reveals that the choice of recommendation strategy should be context-dependent, considering each method has its trade-offs. It encourages future work to explore hybrid models that combine the strengths of both methods, thereby achieving a balance between simplicity, interpretability, and personalization. This project offers insights into the necessity of understanding the trade-offs in designing recommendation systems.

***Index Terms*—recommendation system, recommender, neural network, transformer, text embeddings, similarity, artificial intelligence.**

I. Introduction

The recommendation system has become increasingly important for online applications, and one of the key challenges is to improve the accuracy and efficiency of the system. Content-based recommendation systems are widely used and have become an important research area for the past few years. In this paper, we aim to study and evaluate the state-of-the-art algorithms for content-based movie recommendation systems and develop an AI-based movie recommender.

The Full MovieLens Dataset [1], [2] is a collection of data that includes metadata for 45,000 movies released on or before July 2017, as well as ratings for those movies from 270,000 users, with a total of 26 million ratings. The dataset is available in several files, including movies_metadata.csv, keywords.csv, credits.csv, links.csv, links_small.csv, ratings_small.csv, and ratings.csv. The movies_metadata.csv file contains information on the movies, including posters, backdrops, budget, revenue, release dates, languages, production countries, and companies. The keywords.csv file contains plot keywords for the movies, and the credits.csv file consists of cast and crew information. The links.csv file contains the Movie Database (TMDB) and Amazon IMDB IDs of all the movies, while the links_small.csv file contains this information for a subset of 9,000 movies. The ratings_small.csv file contains a subset of 100,000 ratings from 700 users on 9,000 movies, and the ratings.csv file contains the entire set of 26 million user ratings. Columns of interest in the ratings_small.csv and movies_metadata.csv files include *userIdv*, *movieId*, *rating*, *timestamp* in the ratings_small.csv, along with *genres*, *id* (of the movie), *original_language*, *overview*, *production_companies*, *production_countries*, *tagline*, *vote_average*, *vote_count*, and *title* from the movies_metadata.csv. The data was collected from TMDB and GroupLens, with the movie details, credits, and keywords collected from the TMDB Open API, and the movie links and ratings obtained from the Official GroupLens website. The algorithms used in this research can be classified into similarity-based recommendation and transformer-based recommendation. We utilized the text data of the movie descriptions as the content base of our system and try different text representations and similarity metrics. Our aim is to rank the similarities between each pair of movies and recommend movies in the order of decreasing similarities to the movies currently being watched by the user.

We attempted different text representations, including TF-IDF, GloVe, and BERT, and various similarity metrics, including Cosine similarity and BM25. In addition, we investigated the transformer-based recommendation algorithms, more specifically Behavior Sequence Transformer (BST) model and apply the pattern to our movie recommender. In this study, the BST model learned users' rating histories and movie features to get the taste of each user and predict the ratings of movies, allowing us to see whether a movie is worth being recommended.

The main objectives of this research are to verify the implementation feasibility of different algorithms, evaluate and compare different approaches, and make suggestions about recommendation algorithms in different applications. The report specifically analyzes similarity-based models, demonstrating that while these models are straightforward and easy to interpret, they are limited by data sparsity and a lack of user feedback. In contrast, the BST model offers superior personalization capabilities, but is more complex and susceptible to overfitting. Therefore, the research emphasizes the significance of selecting a recommendation approach that balances simplicity, interpretability, and personalization, and suggests exploring hybrid models that integrate the strengths of both techniques.

This paper is organized as follows: the next section reviews the previous work in the field and literature analysis. We will then describe the data preprocessing methods and algorithms used in this research. After that, we will present the results obtained from the comparison research, including the accuracy and efficiency of the algorithms. Finally, we will discuss the novelty and progress made in this research and the impact of the results on theory and practice.

II. Background and Related Work

The development of Recommendation Systems has a long history, and the idea of using computers to recommend the best item for the user dates back to the beginning of computing. The first implementation of the Recommendation System concept appeared in 1979 with Grundy [3], a computer-based librarian that provided book suggestions. The early 1990s saw the launch of Tapestry [4], the first commercial Recommendation System, and the GroupLens Recommender System [5], a system for helping people find their preferred news articles. Collaborative filtering-based Recommendation Systems, such as Amazon's Collaborative Filtering [6], became popular in the late 1990s and are still widely used today. The success of Amazon led to the development of hybrid approaches, which combine multiple approaches.

In 2006, Netflix launched the Netflix Prize, a competition in Recommendation Systems research, which offered 1 million US dollars to the winner of the competition who provided the best Recommendation System movie recommendation. The winner was announced in 2009 [7]–[9]. YouTube implemented a Recommendation System on its website in 2010 [10].

From the perspective of recommendation models, collaborative filtering technologies dominated the Recommender System applications and studies before 2005, and matrix factorization models [7]–[9] were extensively studied from 2006 to 2009. The first ACM Recommender Systems Conference was held in 2007 [11], and the research has been continuously improved from different aspects, including optimizing methods [12], [13], automatic feature engineering [14], and field-aware factorization machines [15], [16].

Since 2016, deep neural network-based recommendation models have received rapidly growing attention and dominated the development of industry Recommender Systems. In the past five years, deep-learning-based methods have been widely used in industrial recommender systems, e.g., Google [17], [18], and Netflix [19]. Inspired by the great success of the Transformer for machine translation tasks in natural language processing (NLP) [20], Behavior Sequence Transformer was developed by Alibaba which applies the self-attention mechanism to learn a better representation for each item in a user's behaviour sequence by considering the sequential information in the embedding stage, and then feeding them into MLPs to predict users' response to candidate items. The key advantage of the Transformer is that it can better capture the dependency among words in sentences by the self-attention mechanism, and intuitively speaking, the "dependency" among items in users' behaviour sequence can also be extracted by the Transformer. Neural recommendation models are more flexible for advanced feature conjunction and have been proposed to address the issue of insufficient learning of rare feature pairs.

Recent studies have focused on causal inference-inspired recommendations [21]–[24] to address biases in Recommendation Systems. There is still a need for future research to develop recommender system architectures that can address bias while providing high accuracy and good interpretability.

III. Methodology

This section presents a comprehensive outline of the research objectives, methodologies, and strategic implementation approaches utilized in developing AI content-based movie recommendation models. The research objectives are as follows:

1) To explore the current state-of-the-art algorithms for content-based movie recommendation systems through an extensive review of the literature.
2) To determine the feasibility of implementing various algorithms and identify potential implementation challenges early on in the research process.
3) To assess and compare different approaches to recommending relevant movies based on the overview of a movie to identify the most effective and efficient algorithms.
4) To examine potential applications of the developed movie recommendation system and make recommendations for their use.
5) To design an AI content-based movie recommendation system that suggests movies based on the current movie.

Each of these objectives serves a significant purpose for both research and practice. Objective 1 helps to establish a solid foundation for the research by identifying the existing approaches and gaps in the field. Objective 2 allows for the identification of potential implementation issues and the mitigation of risks associated with unsuccessful or inefficient implementations. Objective 3 aims to determine the most effective and efficient algorithms for the content-based movie recommender, and Objective 4 explores potential applications of the developed movie recommendation system. Objective 5 is significant in providing a practical solution to the problem of movie recommendation, which benefits movie watchers and potentially leads to commercial applications.

To achieve these objectives, the proposed methodologies include data pre-processing, algorithm research, and model evaluation.

A. Data Collection

The MovieLens Dataset, which includes metadata for 45000 films released up to July 2017, is used in this study. This dataset provides a comprehensive resource for both researchers and industry professionals. It includes a wide range of data points such as cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts, and vote averages. It also includes 26 million ratings from 6000 users on a 0-5 scale. The data files are organized into various files:

- *users.csv*: This file contains information about users which includes user id, gender, age, occupation, and zip-code.
- *movies_metadata.csv*: This principal file encompasses information on 45000 movies, such as featuring posters, languages, and titles.
- *keywords.csv*: This file contains plot keywords for the movies, presented in JSON format.
- *credits.csv*: This file supplies cast and crew information for the movies, also in JSON format.
- *links.csv*: This file furnishes TMDB and IMDB ids for all movies within the dataset.
- *links_small.csv*: This file is a smaller subset of 9000 movies.
- *ratings.csv*: This file contains ratings for each movie from each user.
- *ratings_small.csv*: This file contains a subset of 100000 ratings from 700 users on 9000 movies.

The *movies_metadata.csv* file is used in the similarity-based approach and the Transformer-based method, along with subsets of the *ratings.csv* and *users.csv* files, given the limited computational power. The data statistics are shown in Table. I.

TABLE I
STATISTICS OF DATA

Number of Users	Number of Movies	Number of Samples
6040	3900	1000209

B. Data Preprocessing

The first step in data pre-processing was to handle missing values and non-English movies. To achieve this, the relevant columns were inspected and checked for missing values, which were removed in the following steps. The non-English movies were also removed from the dataset. Due to the resource limit, 3900 movies were randomly drawn from the cleaned dataset for the following steps.

Then, to preprocess the training data for the similarity-based models, the next step was to combine the columns of movie name and overview into one column called *OverallInfo*. The new column was the major data basis on which our similarity-based models make recommendations.

For the transformer-based algorithm, we applied some different procedures to preprocess the data. in addition to the steps above, the columns of movies metadata dataset other than *genres*, (movie) *id*, *original_language*, *overview*, *production_companies*, *production_countries*, *tagline*, *vote_average*, *vote_count* and *title* were removed. Then the genres, production companies and countries were cleaned up to contain only the list of key information strings, and the different genre categories were made into flags corresponding to each category. For the data to be more effectively fed into the neural network architecture, all id data were converted to strings and the ratings were converted to float numbers. Then the rating records corresponding to the sampled movies were grouped by users and sorted as a sequence in chronological order, and were then broken into fixed-length sequences of movies, with the user groups and chronological order remaining the same. The preprocessed dataset was split into a training dataset with 85% of the data and a validation set with 15% of the data.

C. Similarity-Based Recommendation

In our similarity-based algorithms, we rely primarily on movie descriptions to develop recommendation system models. To accomplish this, we tokenize the movie description texts into smaller pieces and convert them into vectorized text representations. These representations are then fed into models to identify trends and relationships in the text. We evaluate various combinations of text representations and similarity metrics, such as TF-IDF, GloVe, BERT, Cosine similarity, and BM25, to rank the similarities between all pairs of movies. Based on these rankings, we recommend the most similar movies to the one currently being watched. Specifically, the models recommend movies in the order of decreasing similarities to the current movie.

Term Frequency-Inverse Document Frequency (TF-IDF) [25] is a statistical method used to assess the importance of a word to a document in a corpus. It assigns a score to each word in the document based on its frequency and rarity in the corpus, with higher scores given to words that frequently appear in the document and are rare in the corpus, and lower scores assigned to common words that frequently appear in both the document and the corpus. Global Vectors for Word Representation (GloVe) [26], on the other hand, is an unsupervised learning algorithm that generates word embeddings by analyzing word co-occurrence in a corpus. GloVe produces word vectors that capture the semantic relationships between words, including synonyms and antonyms, and is widely used for generating pre-trained word embeddings for natural language processing. Moreover, Bidirectional Encoder Representations from Transformers (BERT) [27] is a deep learning-based technique for natural language processing that creates word embeddings with context from a transformer. BERT is trained on large amounts of text data using a masked language modelling task, where the model predicts a missing word in a sentence. BERT has achieved state-of-the-art results in various NLP tasks such as text classification, question answering, and text generation.

The choices of similarity metrics also affect the recommendations made by the system. Cosine similarity [28] is a measure of similarity between two non-zero vectors that calculates the cosine of their intersection angle. In natural language processing, it is commonly used to assess the similarity between two text representations, such as word embeddings or TF-IDF vectors. The score ranges from -1 to 1, where a score of 1 indicates identical vectors, and a score of -1 indicates completely dissimilar vectors. In addition, BM25, which is part of the Best Matching models family [29], is a ranking function used in information retrieval to evaluate the relevance of a document to a query. BM25 considers various factors, including document length, term frequency, and term rarity in the corpus, to compute a relevance score. It is a popular ranking function used in search engines and recommendation systems.

D. *Transformer-Based Recommendation*

In addition to the similarity-based approach, we propose to investigate a novel algorithm inspired by the transformer model–Behavior Sequence Transformer (BST). This model was developed by Taobao, China's largest Consumer-to-Consumer e-commerce platform owned by Alibaba, in 2019 [30]. As a sequential deep learning model, BST has exhibited substantial benefits in the recommendation industry, generating more dynamic and relevant recommendations based on user behavioural patterns. The BST leverages the Transformer model's prowess in capturing sequential signals that underscore user behaviour, specifically in Alibaba's match-and-rank recommendation pipeline.

Employing the BST model, we modelled the recommendation as a regression problem in which the inputs are user features, movie features, and the actual rating of each movie, and the outputs are predicted ratings. By predicting the ratings of movies, we will be able to tell whether the user likes one particular movie or not, then determine if the movie is the one to recommend.

Fig. 1 illustrates the BST model architecture, which consists of three critical components: the embedding layer, the transformer layer, and a multi-layer perceptron (MLP). The transformer's encoders, utilizing self-attention, effectively integrate signals from past user interactions. This mechanism is able to capture recent shifts in user interests while concurrently preserving long-term context.

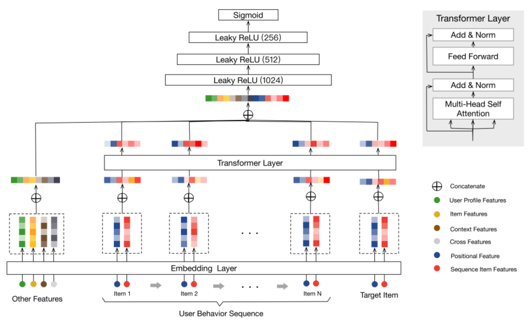

Fig. 1. BST Architecture [30]

1) Input Embedding: This layer reshapes various features from interaction, user, and item data and adds them together to create a final input vector which would be input to the transformer's encoder layers.

Theoretically, the embedding layer integrates various features from interaction, user, and item data, creating a final input vector for the transformer's encoder layers. We represent a movie using "Sequence Item Features" (in red) and "Positional Features" (in dark blue), where the former comprises *movie_id* and *genres*. The model also accounts for a combination of user profile features, movie features, context features, and cross features, collectively denoted as "Other Features." Since this work is focused on modelling the behaviour sequence with a transformer, all these features are denoted as "Other Features" in this model. For computational efficiency, only the *movie_id* and *genres* are included to represent a movie in a behaviour sequence. The "Positional Features" corresponds to the following "positional embedding". Then for each movie, we concatenate Sequence Item Features and Positional Features, and create an embedding matrix

$$\mathbf{W}_V \in \mathbb{R}^{|V| \times d_V},$$

where d_V denotes the dimension size of the embedding, and $|V|$ is the number of movies. We use $\mathbf{e}_i \in \mathbb{R}^{|V| \times d_V}$ to represent the embedding for the i-th movie in a given behaviour sequence.

The interesting part of this model is the Positional Features. The positional embedding was introduced in the original Transformer model to capture the order information in sentences [20]. Similarly, the order exists in users' behaviour sequences. Thus, adding the "position" as an input feature of each movie in the bottom layer before it is projected as a low-dimensional vector is useful in our context. The position value of item v_i is computed as $pos(\nu_i) = t(\nu_t) - t(\nu_i)$, where $t(\nu_t)$ is the recommending time and $t(\nu_i)$ the timestamp when user actually watch that movie ν_i.

2) Transformer Layer: The transformer layer learns a deeper representation of each movie, by learning the relations among other movies within the behaviour sequences. It uses self-attention operations and Point-wise Feed-Forward Networks (FFN) to convert the movie embeddings into three matrices, which feed into an attention layer. To avoid overfitting, ensure non-linearity, and learn meaningful hierarchical features, dropout and LeakyReLU are employed in both self-attention and FFN. After the first self-attention block, it combines all the previous movies' embeddings. Besides, we also stack up the self-building blocks in order to further model the complex relations among movie sequences.

3) MLP Layers and Loss Function: The BST model concatenates the Other Features embedding and the Transformer layer output for the target movie. Subsequent processing involves three fully connected layers to further study the interactions among the dense features.

Predicting the target movie's rating (ν_t) is modelled as a regression problem with continuous output values. So, the model is trained using the Mean Squared Error (MSE) loss:

$$\text{MSE} = \frac{1}{n}\sum(y-\hat{y})^2,$$

where n represents all the samples, $y \in [0,5]$ is the actual movie rating, and $\hat{y} \in [0,5]$ is the predicted rating.

E. Model Evaluation

The similarity-based approach is mainly ranking movies based on similarity scores. Therefore accuracy is not the primary goal of this project. The results we care about and would like to explore are the differences coming from different embedding models and similarity calculations. We will simply pick a movie and compare the recommended ones in the following section.

On the other hand, for the BST model, we use Mean Absolute Error (MAE) to evaluate the performance. The mathematical expression is:

$$\text{MAE} = \frac{\sum_{i=1}^{n}|\hat{y}-y|}{n},$$

where n represents all the samples, $y \in [0,5]$ is the actual movie rating, and $\hat{y} \in [0,5]$ is the predicted rating.

IV. Results

We use a well-known movie which also exists in the dataset, ”Catch Me If You Can”, as an example to show the difference among recommendations given by different similarity metrics. Table. II. III. IV show the results of each combination of each similarity metric and word embedding method.

TABLE II
Recommendations from TF-IDF

Cosine Similarity	BM25
Manhunter	Wolf
Marilyn Hotchkiss' Ballroom Dancing & Charm School	5 Card Stud
The Tarnished Angel	Made for Each Other
Punisher: War Zone	Angel
Wrestling Ernest Hemingway	Arch of Triumph
Bad Ass 2: Bad Asses	Sunday Too Far Away
Barely Legal	Ride, Vaquero!
Pan	Bad Boys
Staten Island Summer	Cry of the Hunted
Wind River	Max

TABLE III
Recommendations from GloVe

Cosine Similarity	BM25
Just Cause	Clash by Night
Young and Innocent	Wolf Creek
Oleanna	Waist Deep
Unstoppable	Stormbreaker
He Was a Quiet Man	Deck the Halls
One for the Money	Confessions of a Nazi Spy
Something big	Busses Roar
Who Was That Lady	Thunder Birds
Bullet to Beijing	Moonshine County Express
2001: A Space Travesty	I Aim at the Stars

TABLE IV
Recommendations from BERT

Cosine Similarity	BM25
Medium Cool	The Abominable Dr. Phibes
Rogue Trader	Weeds
San Quentin	Stormbreaker
The Tarnished Angels	Deck the Halls
Perrier's Bounty	Confessions of a Nazi Spy
One for the Money	Busses Roar
Bad Ass 2: Bad Asses	Thunder Birds
That Man Bolt	Witch Hunt
Sealed Lips	Moonshine County Express
The Argyle Secrets	I Aim at the Stars

From the results, we observe some degree of variance in the movie recommendations given by different embedding methods. We believe that these differences arise from the distinct characteristics of each method. TF-IDF, a baseline method, relies on word frequency statistics within a document and across documents. Its recommendations such as ”Manhunter”, ”Marilyn Hotchkiss' Ballroom Dancing & Charm School”, and ”The Tarnished Angel” suggest that it may have picked up on themes of crime and pursuit in the movie description of ”Catch Me If You Can”. These keywords could pertain to the movie's genre, plot elements, or key themes. As TF-IDF gives importance to terms that are more frequent in a document but less common in the corpus, these recommendations might align with ”Catch Me If You Can” on less commonly discussed but defining aspects.

On the other hand, the GloVe and BERT embeddings, owing to their ability to understand context and semantic meanings, recommend movies that might not only share keywords with ”Catch Me If You Can,” but also align with it in terms of broader themes or narrative structures. For example, if ”Catch Me If You Can” involves themes of deception and pursuit, GloVe and BERT would recommend other movies that share these themes, even if they don't use the same keywords. This is what happened. The *OverallInfo* field of the movies ”Just Cause” and ”Catch Me If You Can” do not share exact keywords but partially common themes.

BM25, like TF-IDF, gives importance to term frequency and inverse document frequency. However, it also incorporates a document length normalization factor, which can give higher importance to matching terms in shorter documents. Therefore, the movies recommended including ”Wolf”, ”Wolf Creek”, and ”5 Card Stud” might not share important matching terms with ”Catch Me If You Can”, but the match might be considered more significant due to the overall length of the *OverallInfo*.

The BST model demonstrated a robust performance, achieving an MAE of 0.75 which implies that, on average, the model's predictions were approximately 0.75 points off from the actual ratings given by users. Considering the scale of movie ratings typically ranges from 0 to 5, this is a relatively low error rate, suggesting a strong ability of the model to accurately predict users' ratings. The configuration of the model is shown in Table. V.

TABLE V
CONFIGURATION OF BST

embedding size	4 ∼ 64	batch size	256
head number	3	dropout	0.1
learning rate	0.01	number of epochs	4

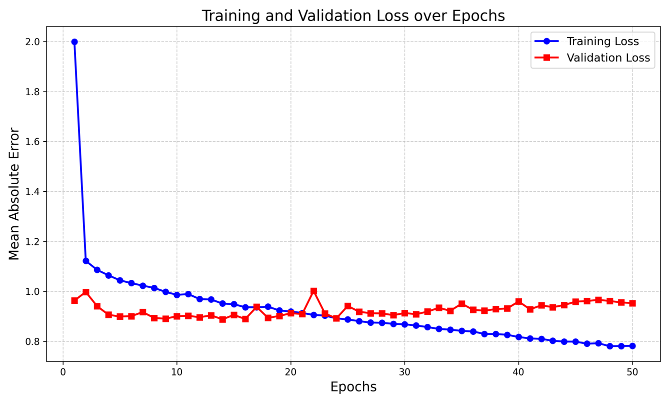

Fig. 2. Training and Validation Loss

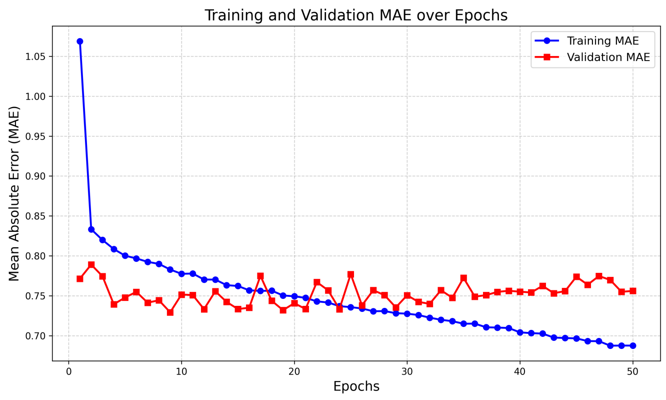

Fig. 3. Training and Validation MAE

Fig. 2 and Fig. 3 represent the loss and MAE over different numbers of epochs. In the plot about training and validation loss, the blue line represents the training loss, and the red line represents the validation loss. Ideally, both lines should decrease over time, indicating that the model is effectively learning and improving its ability to predict the rating of the target movie. If the validation loss stops decreasing or starts increasing, it is a sign of overfitting, which means the model is learning the training data too well and performing poorly on unseen data. The training loss is constantly decreasing, however, validation loss remained at a relatively stable level before 20 epochs and slowly increased to around 1, which is a sign of overfitting. Training and validation MAE shows the same pattern. The second epoch brought a significant improvement to the performance on the training dataset, but the performance on the testing set only fluctuated slightly. From the logical perspective, 4 epochs would be the most optimal option for the sake of saving computational power and training efficiency.

V. CONCLUSIONS AND FUTURE WORK

In this project, we present two different approaches to recommend movies based on movies' or users' content. By vectoring movies' essential information and computing similarity scores with various similarity metrics, we show the variance of recommendations from each combination. The recommendations could be totally different, which is good and bad at the same time. The good is TF-IDF vectorization finds similar contents strictly based on keywords allowing recommendations to be closely related to the original movie, while GloVe and BERT embedding methods consider semantics and relations between words which expands the searching scope. The recommendations may be movies that strict keyword-matching could never find. However, this kind of system can struggle with large and sparse data. If there are many movies and few user interactions, it can be quite expensive to find similar items, and this recommendation strategy did not consider users' preferences at all. Two different users could have opposite reviews on the same movie. On the other hand, the BST model can capture the temporal dynamics of user behaviour, which can be important in certain recommendation scenarios. It also considers the context of previous user behaviour, leading to more personalized and accurate recommendations. And the results are still good even when the data is sparse. As with many deep learning models, lack transparency in their decision-making processes, making it hard to explain why a certain recommendation was made, and overfitting tends to happen when dealing with sparse data. In real life, the choice between a similarity-based recommendation system and a deep-learning-based recommendation system should depend on the specific requirements of the use case. For an e-commerce platform like Alibaba, knowing users' sequential behaviours are beneficial to the business on many levels which makes similarity-based recommendation barely usable.

Despite the results and progress achieved in this project, limitations still exist. Due to the resource limit, only a small subset from the full dataset was used to draw the conclusions. Since a larger set of training data can increase the validity and reliability of models, this study can be further developed with the full MovieLens dataset. Additionally, the similarity-based models were not yet evaluated, due to the lack of quantitive evaluation metrics. Therefore, following the implementation of the similarity-based movie recommender algorithms, potential next steps involve conducting user research and tests to evaluate the effectiveness of these algorithms. The research should involve selecting a diverse group of users, to provide feedback on the system's accuracy and ease of use. Additionally, A/B testing can be conducted to compare the performance of our algorithms of recommender systems. Finally, the results of user research and testing can be analyzed to identify areas for improvement and make necessary adjustments to optimize the performance of our system.

References

[1] "Movielens," Dec 2021. [Online]. Available: https://grouplens.org/datasets/movielens/

[2] R. Banik, "The movies dataset," 2017. [Online]. Available: https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset

[3] E. Rich, "User modeling via stereotypes," *Cognitive Science*, vol. 3, no. 4, pp. 329–354, 1979. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0364021379800129

[4] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Commun. ACM*, vol. 35, no. 12, p. 61–70, Dec 1992. [Online]. Available: https://doi.org/10.1145/138859.138867

[5] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews," MIT Center for Coordination Science, Working Paper Series 165, Mar. 1994. [Online]. Available: https://ideas.repec.org/p/wop/mitccs/165.html

[6] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

[7] Y. Koren, Aug 2009. [Online]. Available: https://www.asc.ohio-state.edu/statistics/dmsl/GrandPrize2009_BPC_BellKor.pdf

[8] A. Töscher and M. Jahrer, Sep 2009. [Online]. Available: https://www.asc.ohio-state.edu/statistics/statgen/joul_aut2009/BigChaos.pdf

[9] M. Piotte and M. Chabbert, Aug 2009. [Online]. Available: https://www.asc.ohio-state.edu/statistics/statgen/joul_aut2009/PragmaticTheory.pdf

[10] J. Davidson, B. Liebald, J. Liu, P. Nandy, and T. V. Vleet, Sep 2010. [Online]. Available: https://www.inf.unibz.it/~ricci/ISR/papers/p293-davidson.pdf

[11] *RecSys '07: Proceedings of the 2007 ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, Oct 2007.

[12] T. Graepel, J. Q. n. Candela, T. Borchert, and R. Herbrich, "Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. Madison, WI, USA: Omnipress, Jun 2010, p. 13–20.

[13] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, S. Chikkerur, D. Liu, M. Wattenberg, A. M. Hrafnkelsson, T. Boulos, and J. Kubica, "Ad click prediction: A view from the trenches," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '13. New York, NY, USA: Association for Computing Machinery, Aug 2013, p. 1222–1230. [Online]. Available: https://doi.org/10.1145/2487575.2488200

[14] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, R. Herbrich, S. Bowers, and J. Q. n. Candela, "Practical lessons from predicting clicks on ads at facebook," in *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, ser. ADKDD'14. New York, NY, USA: Association for Computing Machinery, Aug 2014, p. 1–9. [Online]. Available: https://doi.org/10.1145/2648584.2648589

[15] Y. Juan, Y. Zhuang, W.-S. Chin, and C.-J. Lin, "Field-aware factorization machines for ctr prediction," in *Proceedings of the 10th ACM Conference on Recommender Systems*, ser. RecSys '16. New York, NY, USA: Association for Computing Machinery, Sep 2016, p. 43–50. [Online]. Available: https://doi.org/10.1145/2959100.2959134

[16] S. Rendle, "Factorization machines," in *2010 IEEE International Conference on Data Mining*, Dec 2010, pp. 995–1000.

[17] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah, "Wide & deep learning for recommender systems," 2016.

[18] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM Conference on Recommender Systems*, ser. RecSys '16. New York, NY, USA: Association for Computing Machinery, Sep 2016, p. 191–198. [Online]. Available: https://doi.org/10.1145/2959100.2959190

[19] H. Steck, L. Baltrunas, E. Elahi, D. Liang, Y. Raimond, and J. Basilico, "Deep learning for recommender systems: A netflix case study," *AI Magazine*, vol. 42, no. 3, pp. 7–18, Nov. 2021. [Online]. Available: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/18140

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[21] J. Pearl, *Causality*, 2nd ed. Cambridge University Press, 2009.

[22] D. Liu, P. Cheng, H. Zhu, Z. Dong, X. He, W. Pan, and Z. Ming, "Mitigating confounding bias in recommendation via information bottleneck," in *Proceedings of the 15th ACM Conference on Recommender Systems*, ser. RecSys '21. New York, NY, USA: Association for Computing Machinery, Sep 2021, p. 351–360. [Online]. Available: https://doi.org/10.1145/3460231.3474263

[23] W. Wang, F. Feng, X. He, H. Zhang, and T.-S. Chua, "Clicks can be cheating: Counterfactual recommendation for mitigating clickbait issue," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '21. New York, NY, USA: Association for Computing Machinery, Jul 2021, p. 1288–1297. [Online]. Available: https://doi.org/10.1145/3404835.3462962

[24] Y. Zhang, F. Feng, X. He, T. Wei, C. Song, G. Ling, and Y. Zhang, "Causal intervention for leveraging popularity bias in recommendation," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Jul 2021. [Online]. Available: https://doi.org/10.1145%2F3404835.3462875

[25] K. Sparck Jones, *A Statistical Interpretation of Term Specificity and Its Application in Retrieval*. GBR: Taylor Graham Publishing, Dec 1988, p. 132–142.

[26] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct 2014, pp. 1532–1543.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[28] A. Bhattacharyya, "On a measure of divergence between two multinomial populations," *Sankhyā: The Indian Journal of Statistics (1933-1960)*, vol. 7, no. 4, pp. 401–406, 1946. [Online]. Available: http://www.jstor.org/stable/25047882

[29] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. USA: Cambridge University Press, Jul 2008.

[30] Q. Chen, H. Zhao, W. Li, P. Huang, and W. Ou, "Behavior sequence transformer for e-commerce recommendation in alibaba," 2019.