MOVIE RECOMMENDATION SYSTEM

Sumanth Guduru, Sai Sree Lakshmi Tuta, Anuhya Peddi
Department of Computer Science
Bowling Green State University
Bowling Green, OH 43402, USA

ABSTRACT

It can be challenging for movie enthusiasts to choose the perfect movie, as thousands of movies are produced yearly with different genres. A Personalized recommendation system can be very helpful in this regard, especially when we are unsure which movie to choose. That is exactly where the ”Movie Recommendation System” comes in. Here we present a prototype of a movie recommendation system that caters to the actual needs of movie recommendation by using content-based and collaborative filtering to provide great movie recommendations to the users.

1 INTRODUCTION

In today’s world, it can be challenging for movie enthusiasts to choose the perfect movie to watch, where thousands of movies are produced every year with different genres. This can impact the viewing experience and satisfaction of the users significantly. So, to address this issue, a movie recommendation system has been developed that suggests movies based on the user’s past viewing history and ratings [1]. The system employs collaborative filtering, content-based filtering, and hybrid recommendations, with collaborative filtering using the SVD algorithm and content-based filtering utilizing the TF-IDF vectorizer and cosine similarity. Some users may not have provided enough data, making it challenging to provide accurate recommendations for them. Another challenge is the cold-start problem, which makes it difficult for the system to provide personalized recommendations for new users who have not yet provided enough data. We must also consider scalability, evaluation, diversity, and privacy concerns while creating the recommendation system. The approach considers these challenges and proposes incorporating matrix factorization and content-based recommendations to overcome them. The research provides valuable insights into developing more accurate and relevant movie recommendation systems. It fits into related work in personalized recommendation systems, which aims to provide personalized user recommendations based on their interests and preferences. The basic results and conclusions demonstrate that the hybrid approach performs better than individual approaches, and the incorporation of matrix factorization and content-based recommendations can enhance the accuracy of the system.

2 RELATED WORK

Lund and Ng et al. proposed a movie recommendation system using deep learning approach in their paper published in the 2018 IEEE International Conference on Information Reuse and Integration (IRI). The authors employed a deep learning algorithm to extract high-level features from user ratings, and the resulting features were used to predict user preferences for new movies. The proposed approach outperformed traditional collaborative filtering techniques in terms of prediction accuracy. The authors demonstrated the effectiveness of the proposed method through experiments on a publicly available movie dataset [2].

Wei, Xiao, Zheng and Chen et al. proposed a hybrid movie recommendation approach using social tags. They integrated user-generated social tags with movie content data to generate movie recommendations. The proposed method showed improvement in recommendation accuracy compared to traditional approaches. The study highlights the importance of incorporating user-generated data in recommendation systems to enhance personalization and user satisfaction [3].

Roy and Ding et al. proposed a movie recommendation method that uses YouTube movie trailer data as side information. The method uses a graph-based model to capture the complex relationships between movies and trailers. The authors showed their approach outperformed several baseline methods on a real-world movie dataset. This method could potentially be used in movie recommendation systems to improve the accuracy and relevance of recommendations [4].

Halder, Sarkar and Lee et al. proposed a movie recommendation system based on Movie Swarm that allows users to express their preferences and opinions by voting and commenting on movies. The system uses a novel algorithm to compute the similarity between movies based on user voting and comments and then recommends movies based on the similarity scores. The proposed system outperformed existing recommendation systems in terms of accuracy and user satisfaction. Overall, the study suggests that incorporating user-generated content can improve the quality of movie recommendations [5].

In a typical recommendation system, not all users rate all items, leading to missing values in the user-item rating matrix. SVD can handle these missing values, whereas k-NN cannot. k-NN requires a complete user-item rating matrix to function properly. SVD can also capture latent factors not explicitly represented in the data. SVD can also handle large datasets using its scalable technique.

3 Problem Definition and Algorithm

3.1 Task Definition

The movie recommendation problem involves recommending movies to users based on their preferences and behaviour. It utilises user data such as rating dataset (including user ID, movie title, and ratings given), user profiles, and movie data such as genre and title. The output is a list of recommended movies the user will likely enjoy. This problem is important and interesting, especially in the streaming services industry, where personalized recommendations can increase user satisfaction and retention. However, this task is challenging due to the need for accurate, diverse, and comprehensive recommendations and overcoming the cold start problem and data sparsity issues.

3.2 Algorithm/Methodology Definition

We are using two algorithms, Content-based Filtering and Collaborative Filtering.

3.2.1 Content-Based Filtering(TF-IDF vectorizer and cosine similarity)

Content-based filtering uses item features to recommend other items similar to what the user likes based on their previous actions or explicit feedback [8].

Let's see how TF-IDF works in a movie recommendation system,
Term Frequency (TF) and Inverse Document Frequency (IDF) are commonly used in recommendation systems to determine the similarity between items based on their textual descriptions. In the case of a movie recommendation system, TF-IDF can be used to determine the similarity between movies based on the words used in their plot summaries, reviews, or other textual metadata [10].

The first step in using TF-IDF for movie recommendations is to build a corpus of text data for each movie in the system. This can include plot summaries, reviews, cast and crew information, and other metadata. The corpus for each movie is then tokenized, meaning it is split into individual words. Next, the TF-IDF score for each word in each movie's corpus is calculated. TF-IDF measures how important a word is to a document in a corpus. It considers both the word frequency in the document (TF) and the inverse frequency of the word in the entire corpus (IDF). Once the TF-IDF scores for each word in each movie's corpus have been calculated, a similarity score can be computed between any two movies based on their textual descriptions. One common method for computing similarity is to use the cosine similarity measure, which compares the TF-IDF vectors for each movie. Finally, the system can recommend movies to users based on their past viewing history and ratings and the similarity between those movies and other movies in the system. For example, suppose a user has rated several action movies highly. The system might recommend other action movies with high cosine similarity scores based on their plot summaries or other textual metadata.

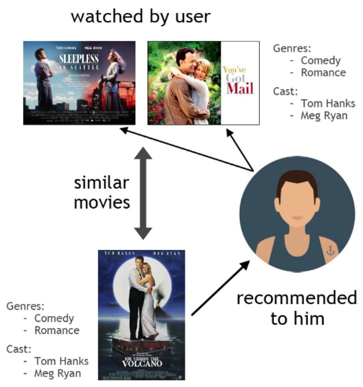


Figure 1: content based filtering

In Figure 1, Bob is a user who has watched two movies, ”Sleepless In Seattle” and ”You’ve Got Mail.” Both movies belong to the comedy and romance genres, featuring actors Tom Hanks and Meg Ryan. When a model is trained with this information, it identifies two important preferences of Bob. Firstly, Bob likes to watch movies that fall under the comedy and romance genres. Secondly, Bob prefers movies that feature Tom Hanks and Meg Ryan. Based on these preferences, the model recommends a movie that includes Tom Hanks and Meg Ryan as actors and falls under the comedy and romance genres. As a result, the system suggests the movie ”Joe Versus The Volcano,” which features both Tom Hanks and Meg Ryan and is a comedy and romance movie.

In content-based filtering, we are also using cosine similarity, where Cosine similarity is a mathematical technique to calculate the similarity between two vectors. In movie recommendation systems, each movie can be represented as a vector of its content features such as genre, director, cast, and plot. To calculate the similarity between two movies using cosine similarity, we take the dot product of their feature vectors and divide it by the product of their vector magnitudes. This gives us a similarity score between 0 and 1, where 1 represents perfect similarity, and 0 represents no similarity.

We can use cosine similarity for movie recommendation systems to recommend movies similar to a user’s past preferences. We can represent a user’s movie preferences as a vector of features and calculate the similarity between this vector and the feature vectors of all the movies in our dataset. The movies with the highest similarity score are recommended to the user.

Cosine similarity is a popular technique for movie recommendation systems because it is computationally efficient and can handle high-dimensional feature vectors. It also does not require prior knowledge of user preferences and can recommend movies based solely on their content features. Additionally, it can be combined with other recommendation techniques such as collaborative filtering to improve the accuracy and diversity of recommendations.

3.2.2 Collaborative Filtering (SVD)

Collaborative filtering uses similarities between users and items simultaneously to provide recommendations.

In the given figure 2, two users, Alice and Bob, have watched the same two movies, ”Sleepless in Seattle” and ”You’ve Got Mail.” Based on this information, the system calculates that Alice and Bob have similar movie tastes. As a result, whenever Alice watches a movie, the system recommends it to Bob, as it assumes that Bob would also enjoy watching the movie due to their similar preferences. This approach is called collaborative filtering, where recommendations are made based on multiple

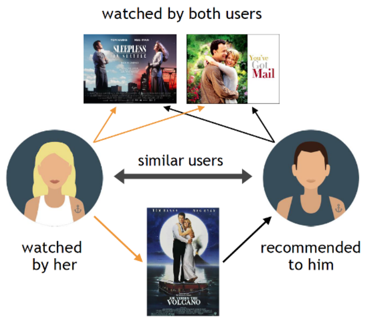


Figure 2: Collaborative filtering

users' shared interests and preferences.
In collaborative-based filtering, we are using the SVD algorithm. Singular Value Decomposition (SVD) is a matrix factorization technique. It decomposes a matrix into three matrices - a left singular matrix, a diagonal matrix, and a right singular matrix. The diagonal matrix contains the singular values of the original matrix, which represent the degree of variance explained by each component. The left singular matrix represents the relationship between the rows of the original matrix, while the right singular matrix represents the relationship between the columns.

SVD is used to identify latent factors that influence a user's movie preferences. It does this by analyzing the user-movie rating matrix, where each row represents a user, each column represents a movie, and each cell contains the rating given by the user to the movie [11].
Using SVD, this matrix is decomposed into three matrices: a user-feature matrix, a feature-weight matrix, and a movie-feature matrix. The feature-weight matrix contains the singular values representing each feature's importance in explaining the users' preferences.
The movie feature and user feature matrices are then used to make recommendations. The movie-feature matrix is used to identify the latent features of each movie, while the user-feature matrix is used to identify the latent features of each user. These matrices are then multiplied to produce a prediction matrix containing all users' predicted ratings for all movies. The recommendation system then recommends the movies with the highest predicted ratings to the user.

The hybrid recommendation function combines the results from the content-based filtering and collaborative filtering algorithms and generates a set of hybrid recommendations for the user [9]. The content-based recommendations are generated based on the similarity of the item's genres to those of the user's input movie. In contrast, the collaborative filtering recommendations are based on the user's past ratings and the estimated ratings for unrated items. The function returns the top 5 items that appear in both sets of recommendations. If fewer than five items are in this intersection, the remaining items are selected from the content-based recommendations in descending order of similarity.

4 Experimental Evaluation

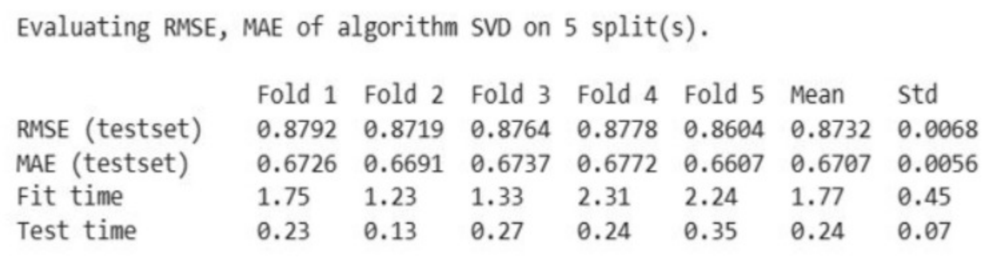
4.1 Experimental Methodology

The Movie recommendation system uses content-based and collaborative filtering techniques to recommend movies to users. We primarily evaluate our method based on accuracy, coverage, and diversity. Our project tests the hypothesis that a hybrid recommendation system combining content-based and collaborative filtering techniques will produce more accurate, diverse, and comprehensive movie recommendations for users than content-based or collaborative filtering techniques alone. To evaluate our method, we used cross-validation techniques such as k-fold cross-validation. We split our dataset into training and test sets and used the training set to train our model and the test set to evaluate its performance.

The experimental methodology involves using the MovieLens dataset, preprocessing the movie data for content-based filtering using TfidfVectorizer and cosine similarity, and training a collaborative filtering algorithm using SVD in the Surprise library. The independent variables in this experiment are the content-based and collaborative filtering techniques, while the dependent variable is the accuracy, coverage, and diversity of the movie recommendations. The training/test data is a subset of the MovieLens dataset, where 80The performance data collected includes accuracy measures such as mean squared error (MSE), root mean squared error (RMSE), and coverage and diversity measures. We want our system to cover as many movies as possible to give users a wide range of choices. We also want our system to recommend movies from different genres and categories to ensure users get exposure to a broad range of movies.

4.2 Results

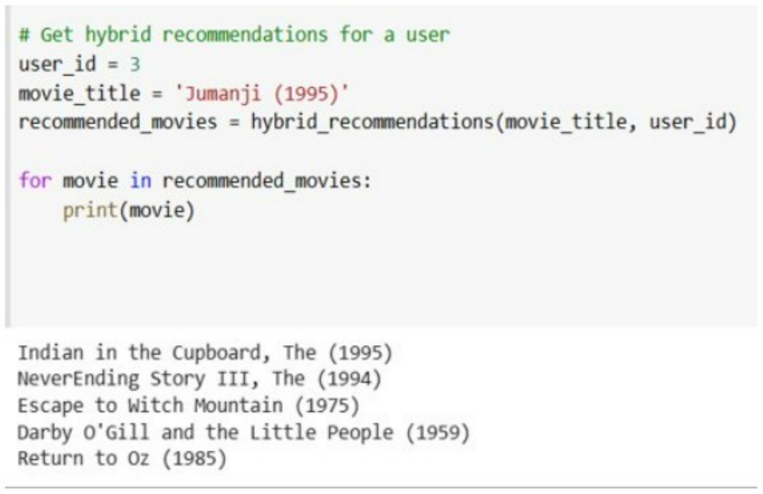
To present the quantitative and qualitative results of the movie recommendation system, we evaluated the performance of the SVD algorithm using the mean squared error (MSE) and root mean squared error (RMSE) metrics. We also tested the hybrid module that combined content-based and collaborative filtering techniques and observed that it provided good user recommendations. However, we did not evaluate the accuracy of the hybrid module. We presented the results in a table to illustrate the performance of the SVD algorithm and the recommendations provided by the hybrid module. The results showed that the SVD algorithm performed well in predicting user ratings and the hybrid module provided good movie recommendations to users.



```
Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.8792  0.8719  0.8764  0.8778  0.8604  0.8732  0.0068
MAE (testset)     0.6726  0.6691  0.6737  0.6772  0.6607  0.6707  0.0056
Fit time          1.75    1.23    1.33    2.31    2.24    1.77    0.45
Test time         0.23    0.13    0.27    0.24    0.35    0.24    0.07
```

Figure 3: Evaluation results of SVD algorithm



```
# Get hybrid recommendations for a user
user_id = 3
movie_title = 'Jumanji (1995)'
recommended_movies = hybrid_recommendations(movie_title, user_id)

for movie in recommended_movies:
    print(movie)
```

```
Indian in the Cupboard, The (1995)
NeverEnding Story III, The (1994)
Escape to Witch Mountain (1975)
Darby O'Gill and the Little People (1959)
Return to Oz (1985)
```

Figure 4: Result of Hybrid recommendation for a user

4.3 Discussion

The study supports the hypothesis that combining content-based filtering and collaborative filtering leads to better recommendation results. The strengths of content-based filtering are in recommending movies based on user preferences, while collaborative filtering is better at finding similarities between users. The SVD algorithm used in collaborative filtering and cross-validation improved the results of the hybrid recommendation system by avoiding overfitting and better generalizing to unseen data. The properties of the algorithms and data can explain the study's results. The MovieLens dataset provides many user ratings, allowing collaborative filtering to identify patterns and similarities between users. Content-based filtering, which uses TF-IDF vectorizer and cosine similarity, identifies similarities between items based on their content. A hybrid recommendation system is an effective approach for movie recommendation systems.

5 Future Work

The current method of the hybrid movie recommendation system that we are using combines content-based filtering and collaborative filtering algorithms. It has some major limitations that need to be addressed. One such limitation is the issue of data sparsity. Here the system may struggle to provide accurate recommendations for movies with few ratings. So, to overcome this limitation, the system can be enhanced with content-based recommendations that focus on the characteristics of the movies, such as genre, cast, or plot. Additionally, matrix factorization techniques can also be used to reduce the data sparsity and to provide more accurate recommendations to the users.

Another limitation is the cold start problem. Here the system struggles with new users or movies with little or no data. To overcome this limitation, the system can be enhanced with content-based recommendations that focus on movie attributes like genre, cast, plot, and reviews. By analyzing all these attributes of a movie, the system can provide relevant recommendations to new users or movies with no data. And also, hybrid approaches with item-based collaborative filtering and content-based filtering can be used to address this cold start problem.

6 Conclusion

Finally, the movie recommendation system uses Content-Based Filtering, Collaborative Filtering (SVD), and hybrid recommendation techniques. Comparatively, the hybrid recommendation system provided better recommendations than the individual methods. Limitations such as data sparsity and the cold start problem can be overcome by incorporating content-based recommendations, matrix factorization, and hybrid approaches. And the machine learning algorithms we used, such as SVD and TF-IDF vectorizer with cosine similarity, can help generate more accurate recommendations to the users. And the enhancements proposed in this report can improve the results of the movies that are being recommended. This research provides insights into the limitations of the current method and proposes enhancements that can be used to develop a more accurate and relevant movie recommendation system.

References

[1] J. Zhang, Y. Wang, Z. Yuan and Q. Jin, ”Personalized real-time movie recommendation system: Practical prototype and evaluation,” in Tsinghua Science and Technology, vol. 25, no. 2, pp. 180-191, April 2020, doi: 10.26599/TST.2018.9010118

[2] J. Lund and Y. Ng, ”Movie Recommendations Using the Deep Learning Approach,” in 2018 IEEE International Conference on Information Reuse and Integration (IRI), Salt Lake City, UT, USA, 2018 pp. 47-54. doi: 10.1109/IRI.2018.00015

[3] S. Wei, L. Xiao, X. Zheng and D. Chen, ”A Hybrid Movie Recommendation Approach via Social Tags,” in 2014 IEEE 11th International Conference on e-Business Engineering (ICEBE), Guangzhou, China, 2014 pp. 280-285. doi: 10.1109/ICEBE.2014.55

[4] D. Roy and C. Ding, ”Movie Recommendation using YouTube Movie Trailer Data as the Side Information,” in 2020 IEEE/ACM International Conference on Advances in Social Net-

works Analysis and Mining (ASONAM), The Hague, Netherlands, 2020 pp. 275-279. doi: 10.1109/ASONAM49781.2020.9381349

[5] S. Halder, A. Sarkar and Young-Koo Lee, ”Movie Recommendation System Based on Movie Swarm,” in 2012 International Conference on Cloud and Green Computing (CGC), Xiangtan, 2012 pp. 804-809. doi: 10.1109/CGC.2012.121

[6] R. Ahuja, A. Solanki and A. Nayyar, ”Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor,” 2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence), Noida, India, 2019, pp. 263-268, doi: 10.1109/CONFLUENCE.2019.8776969.

[7] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Trans. Interact. Intell. Syst. 5, 4, Article 19 (January 2016), 19 pages. https://doi.org/10.1145/2827872

[8] Shanmugapriya, S., Mala, G. S. (2018). Content-based movie recommendation system using genre correlation. 2018 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 1-5. DOI: 10.1109/ICCIC.2018.8722818

[9] Walek, B., Fojtik, V. (2020). A hybrid recommender system for recommending relevant movies using an expert system. Expert Systems with Applications, 158, 113452. https://doi.org/10.1016/j.eswa.2020.113452

[10] Ni, J., Cai, Y., Tang, G., Xie, Y. (2021). Collaborative Filtering Recommendation Algorithm Based on TF-IDF and User Characteristics. Applied Sciences, 11(20), 9554. https://doi.org/10.3390/app11209554.

[11] A. Pal, P. Parhi and M. Aggarwal, ”An improved content based collaborative filtering algorithm for movie recommendations,” 2017 Tenth International Conference on Contemporary Computing (IC3), Noida, India, 2017, pp. 1-3, doi: 10.1109/IC3.2017.8284357.