AML-3204 Final Project: Collaborative Filtering and Hybrid Recommender Systems

1st Bhavya Vadher
Artificial Intelligence & Machine Learning
Lambton College
Toronto, Canada
c0894977

2st Devarsh Jadhav
Artificial Intelligence & Machine Learning
Lambton College
Toronto, Canada
c0893173

3st Viki Patel
Artificial Intelligence & Machine Learning
Lambton College
Toronto, Canada
c0906295

4st Syed Roman
Artificial Intelligence & Machine Learning
Lambton College
Toronto, Canada
c0906298

Abstract—***This report explores the development and evaluation of a hybrid recommender system incorporating both matrix factorization and neural embeddings. We compared the performance of a Non-Negative Matrix Factorization (NMF) model with an enhanced hybrid recommender system that integrates user demographics and item attributes. The collaborative filtering approach (NMF) demonstrated lower RMSE and MAE, highlighting its efficiency in capturing user preferences. The hybrid model, while showing improvements in precision, struggled with recall and F1-score, indicating challenges in handling diverse user interactions. This study provides insights into the strengths and limitations of different recommender systems, offering guidance for future enhancements.***

I. Introduction

Recommender systems are pivotal in providing personalized content and enhancing user experiences in various applications. Traditional methods include collaborative filtering, which relies on user-item interactions, and content-based filtering, which leverages item attributes. Hybrid recommender systems combine these approaches to address limitations inherent in each. Collaborative filtering, such as matrix factorization, excels in capturing user preferences but may struggle with cold-start problems. Content-based methods address these issues by incorporating item features but can lack personalization. This report evaluates a hybrid recommender system that integrates collaborative filtering with neural embeddings and sentiment analysis, aiming to improve recommendation accuracy and user satisfaction.

II. Methodology

A. Introduction to Methodologies

Introduction to Methodologies: This section introduces the methodologies used in developing the recommender systems. We applied matrix factorization for collaborative filtering and combined it with neural embeddings and sentiment analysis to create a hybrid recommender system. Matrix factorization helps to identify latent features from user-item interactions, while neural embeddings enhance feature representation. Sentiment analysis adds an additional layer of user feedback, aiming to improve the recommendation accuracy.

B. Datesets

The dataset used in this project is from the MovieLens dataset, collected by the GroupLens Research Project at the University of Minnesota. It includes: Ratings Data (u.data): 100,000 ratings (ranging from 1 to 5) from 943 users on 1682 movies. Each user has rated at least 20 movies.

- Data format: user id, item id, rating , timestamp Timestamps are Unix seconds since 1/1/1970 UTC.
- Movie Information (u.item): A tab-separated list including: movie id, movie title, release date, video release date, IMDb URL, unknown, Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War, Western. The last 19 fields represent genres, with a binary indicator (1 or 0) for each genre.
- Genre List (u.genre): A list of all possible genres for the movies.
- IMDB Movie Reviews: For each movie, reviews were collected from IMDB instead of Twitter. Reviews were cleaned and analyzed to calculate sentiment scores. The average sentiment score for each movie was computed and used as part of the feature set in the recommender system.

C. Matrix Factorization

Matrix Factorization is a technique used in collaborative filtering to discover latent features underlying user-item interactions. In recommendation systems, the goal is to predict missing entries in a user-item matrix, where rows represent users, columns represent items, and the entries represent ratings or interactions.

Matrix Factorization Algorithm: The matrix factorization algorithm decomposes the user-item interaction matrix X into two lower-dimensional matrices: W (user features matrix), H (item features matrix)

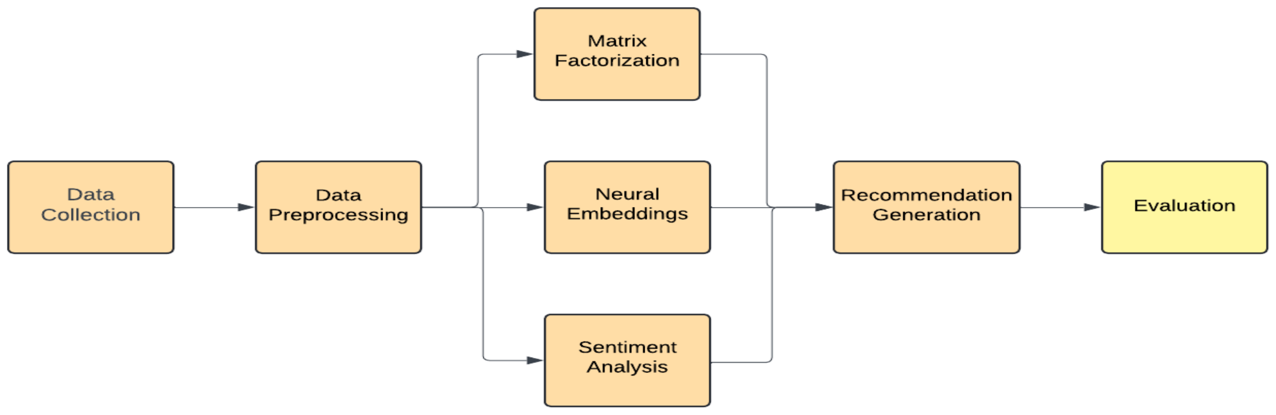


Figure 1: System Architecture Diagram

Error!

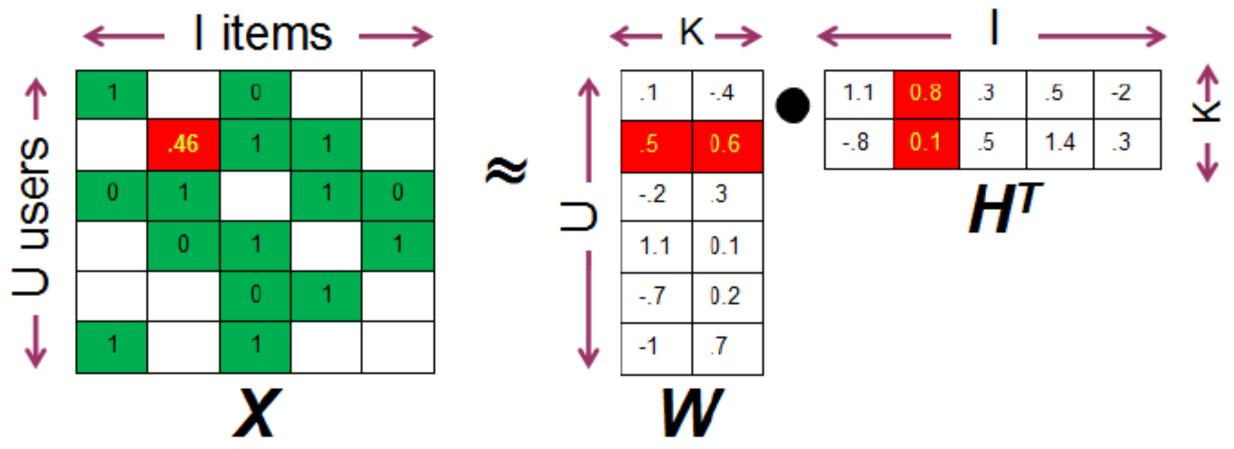


Figure 2: Matrix Factorization

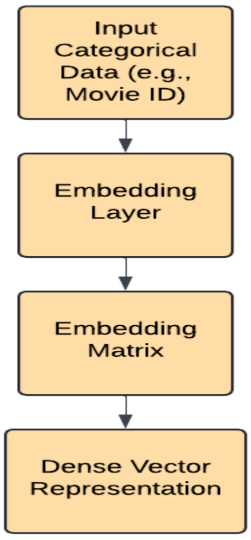
The basic idea is to approximate the original matrix X by the product of these two matrices, so that X≈W·H^T , where W and H are of lower rank.

Matrix Factorization Equation: The core matrix factorization equation can be expressed as: X≈W·H^T Where: X is the user-item matrix., W is the matrix of user features (size: m×k), H is the matrix of item features (size: n×k), k is the number of latent features, Components of the Equation:

- User Features Matrix W: Represents latent features associated with users. Each row corresponds to a user and each column corresponds to a latent feature.
- Item Features Matrix H: Represents latent features associated with items. Each row corresponds to an item and each column corresponds to a latent feature.

D. Neural Embedding Layer

Neural Embedding Layer is a technique used to represent discrete categorical variables in a continuous vector space. This method is particularly useful for recommendation systems as it helps capture complex relationships between users and items.



Input Categorical Data (e.g., Movie ID)
Embedding Layer
Embedding Matrix
Dense Vector Representation

Figure 3: Embedding Layer

Embedding Layer: The embedding layer is a learned matrix where each row corresponds to a dense vector representation of a category (e.g., user ID, movie ID).
It maps discrete categories (such as user or movie IDs) to dense vectors of fixed size, which are trained to capture the relationships between these categories.

Process:
Input: Categorical data such as movie IDs or user IDs.

- Embedding Matrix: A matrix where each row represents the embedding vector of a category.
- Output: Dense vectors that are used in subsequent layers of the neural network.

Suppose we have a movie ID 42 and an embedding size of 10. The embedding layer will transform the movie ID 42 into a 10-dimensional vector:
Movie ID 42 -> [0.12, -0.34, 0.45, -0.23, 0.67, ..., -0.54] (10-dimensional vector)

E. Sentiment Score Calculation

Sentiment Score Calculation involves evaluating textual data to gauge sentiment. For this project, we used TextBlob to analyze movie reviews from IMDb and calculate sentiment scores.

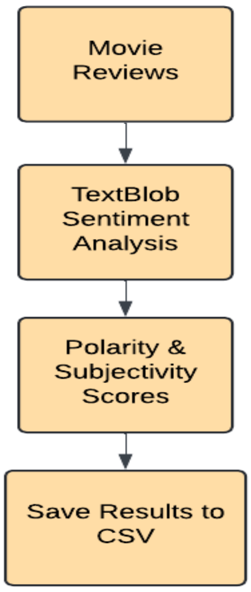


Figure 3: Sentiment Score

<u>Sentiment Analysis Tool:</u>

- Tool Used: TextBlob, a Python library for processing textual data, which provides sentiment analysis capabilities.

<u>Process:</u>

- Handling Missing Values: Any missing values in the Review column were replaced with empty strings to ensure the sentiment analysis function operates smoothly.
- Sentiment Analysis Function: A function, analyze_sentiment, was created to compute sentiment scores from each review.

<u>It extracts two key metrics:</u>

- Polarity: Indicates the sentiment's orientation, ranging from -1 (negative) to +1 (positive).
- Subjectivity: Measures the sentiment's subjectiveness, ranging from 0 (objective) to 1 (subjective).
- Application: The analyze_sentiment function was applied to each review in the dataset to obtain sentiment scores. Results were stored in two new columns: Polarity and Subjectivity.

<u>Saving Results:</u>
The updated dataset, including sentiment analysis results, was saved to a CSV file for further use.

F. Activation Function

Activation functions are a crucial component of neural networks, introducing non-linearity to the model and allowing it to learn complex patterns.
In our Hybrid Recommender System, we used activation functions to enhance the model's learning ability. The choice of activation function can significantly impact the model's performance.

<u>Selected Activation Function:</u>
ReLU (Rectified Linear Unit): The ReLU activation function is used in our hybrid model. It is defined as:
$f(x)=\max(0,x)$
Characteristics:

- Non-linearity: ReLU introduces non-linearity, enabling the model to learn complex relationships.
- Sparsity: It helps in achieving sparse activation, which can lead to a more efficient model.

Gradient Propagation: ReLU mitigates the vanishing gradient problem, which is common with other activation functions like sigmoid or tanh.
ReLU is applied after the linear transformation of the data in each layer of the network.
Helps the network learn non-linear mappings and improve performance in complex tasks.

III. Experiments

A. Experiment Design

The experiment is designed to evaluate and compare different recommender systems, including Matrix Factorization and the Hybrid Recommender System. The goal is to assess the performance of these models based on various metrics.

<u>Machine Configuration:</u>

- Hardware: Utilize a GPU for training deep learning models.
- Software: Ensure compatibility with Python 3.12.0 and necessary libraries.

APIs and Libraries:

- Selenium: For web scraping and gathering additional data if needed.
- TextBlob: For sentiment analysis of reviews.
- PyTorch: For building and training neural networks.
- Scikit-learn: For metrics calculation and data preprocessing.
- Pandas: For data manipulation and analysis.
- NumPy: For numerical operations.

Data Cleaning:

Handle missing values, particularly in the 'Review' column, by replacing NaNs with empty strings.

Clean movie reviews and extract relevant information for sentiment analysis.

Data Transformation:

Apply sentiment analysis to the reviews to calculate sentiment scores (polarity and subjectivity).

Convert categorical data (genres) into a format suitable for model training (e.g., one-hot encoding).

B. Dataset Preparation

The dataset preparation involves cleaning, transforming, and structuring data to make it suitable for training and evaluating the recommender systems. Here's a step-by-step guide on how to prepare the datasets.

Loading Data:

Load the MovieLens dataset and any additional data (e.g., IMDB reviews) into Pandas DataFrames and make new dataframe.

Data Splitting:

Split the data into training and testing sets using train_test_split from Scikit-learn.

Quantifiable Information:

- Average number of reviews per movie.
- Average ratings per movie.
- Highest and lowest number of ratings for a movie.

Various visualizations were created to provide insights into the distribution and characteristics of the data:

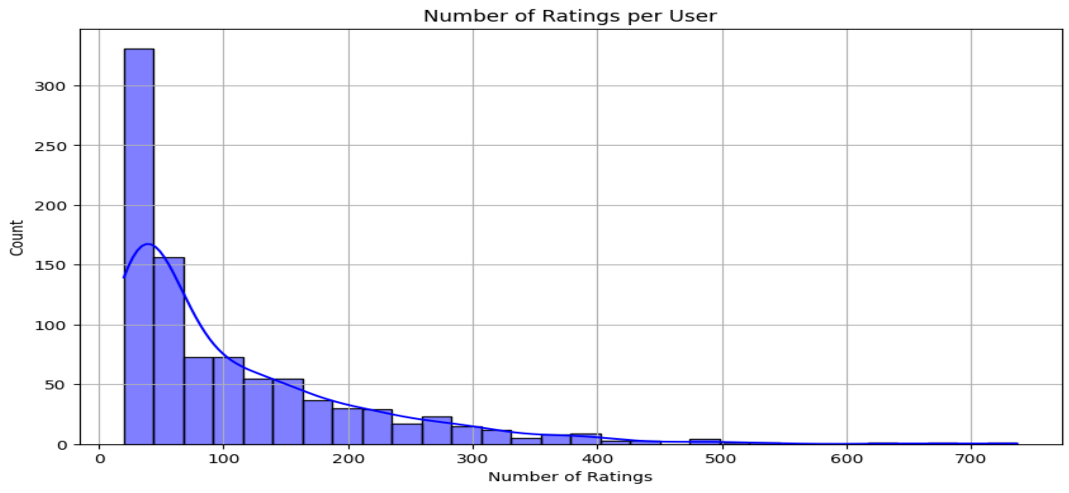


Figure 4: Number of Ratings per User

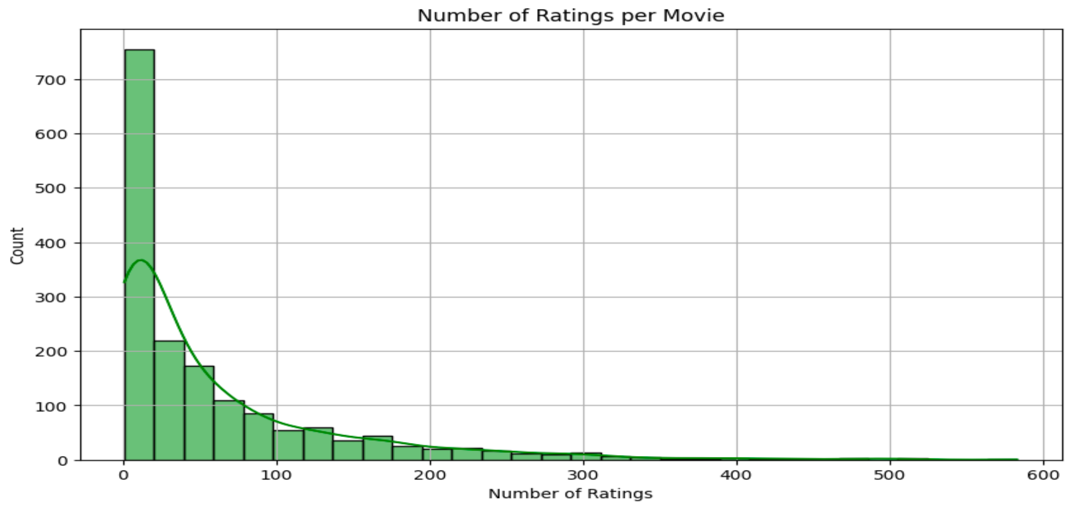


Figure 5: Number of Ratings per Movie

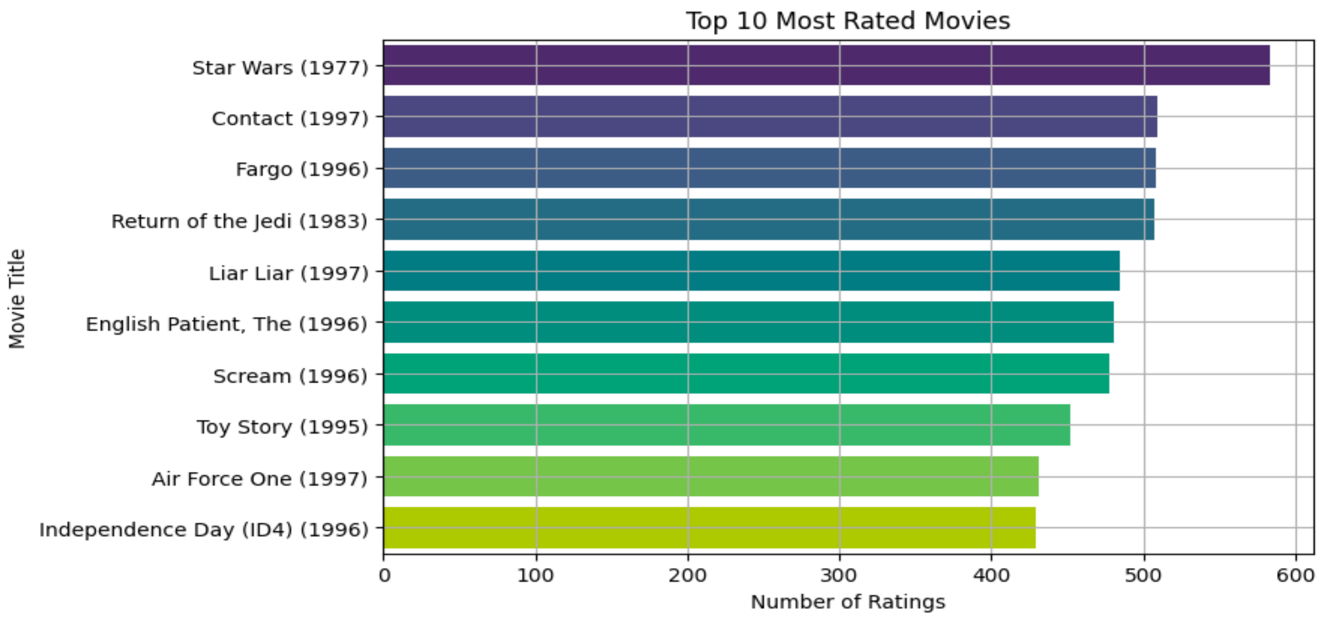


Figure 6: Top 10 Most Rated Movies

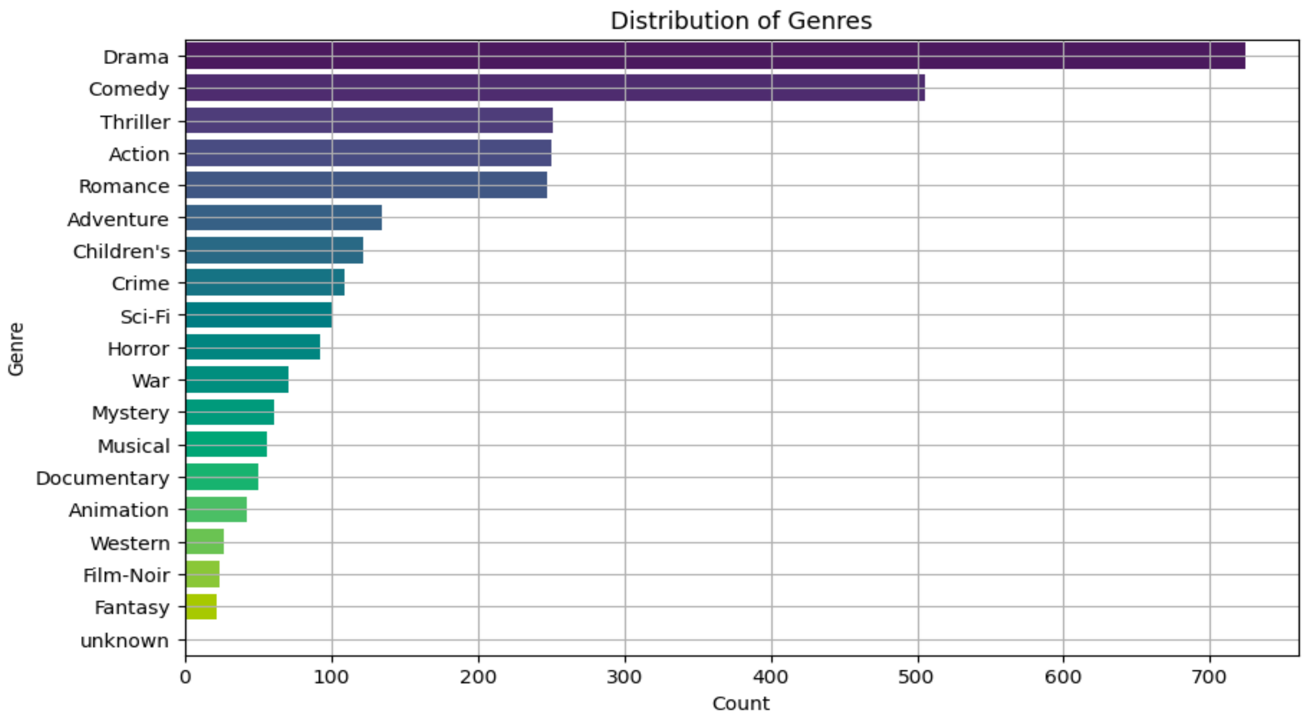


Figure 7: Distribution of Genres

C. *Evaluation Metrics*

To evaluate the performance of the recommendation systems, several metrics were used:

<u>RMSE (Root Mean Squared Error):</u>
Measures the square root of the average squared differences between predicted and actual ratings. It indicates how well the model predicts ratings.

<u>MAE (Mean Absolute Error):</u>
Represents the average absolute difference between predicted and actual ratings. It provides an overall sense of the error magnitude.

<u>Precision:</u>
Measures the proportion of relevant items among the retrieved items.

<u>Recall:</u>
Represents the proportion of relevant items that have been retrieved.

<u>F1-Score:</u>
The harmonic mean of Precision and Recall, providing a balanced measure of the model's performance.

$$\text{RMSE} = \sqrt{\frac{\sum (y_i - y_p)^2}{n}}$$

$$\text{MAE} = \frac{|(y_i - y_p)|}{n}$$

y_i = actual value
y_p = predicted value
n = number of observations/rows

Figure 8: RMSE & MAE Formulas

Formulae for Classification Metrics

$$\text{Precision} \quad \frac{tp}{tp + fp}$$

$$\text{Recall (Sensitivity)} \quad \frac{tp}{tp + fn}$$

$$\text{F1-score} \quad \frac{2 * precision * recall}{precision + recall}$$

© Jack Tan

Figure 9: Precision, Recall & F1-score Formulas

D. *Results and Analyses*

1. Hyperparameter Selection

In this section, we evaluate the performance of two different models—Matrix Factorization and neural Network-based Hybrid Recommender—using various latent vector sizes. The goal is to identify the optimal latent vector size by comparing the Root Mean Square Error (RMSE) for each configuration.

Latent Vector Size	Matrix Factorization RMSE	Hybrid Recommender RMSE
20	0.1329	3.7545
30	0.1295	3.3754
40	0.1267	1.6373
50	0.1242	1.3060

Table 1: RMSE values of Different Latent Vector Size

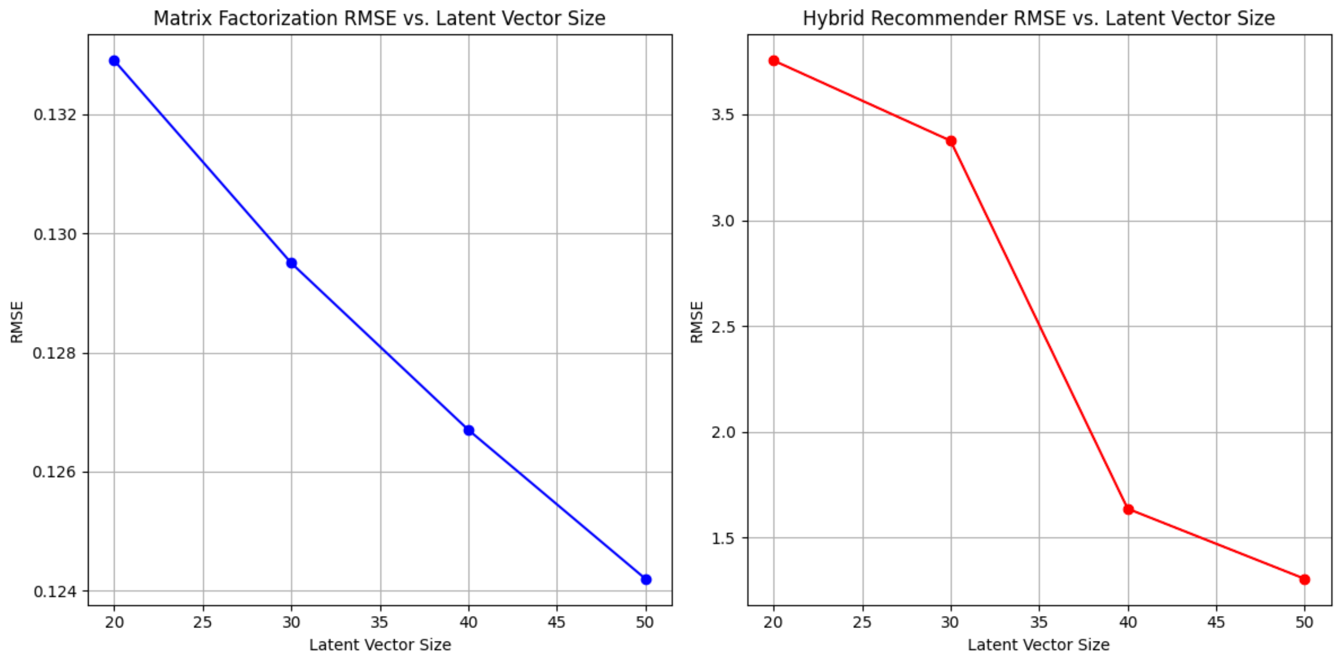


Figure 10: RMSE values Graph of Different Latent Vector Size

2. Results of Recommendation Models

In this section, we compare and discuss the results of two recommendation models: the Non-Negative Matrix Factorization (NMF) model and the Hybrid model. The comparison is based on several performance metrics: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Precision, Recall, and F1-Score for top-10 and top-20 recommendations. These metrics provide insights into the accuracy and relevance of the recommendations generated by each model.

Performance Metrics Table

Metric	NMF Model	Hybrid Model
RMSE	0.4207	0.4196
MAE	0.3439	0.3428
Precision (Top-10)	0.5012	0.4981
Recall (Top-10)	0.5023	0.4979
F1-Score (Top-10)	0.5017	0.4980
Precision (Top-20)	0.5005	0.4998
Recall (Top-20)	0.5005	0.5015
F1-Score (Top-20)	0.5005	0.5007

Table 2: Performance Metrics Table

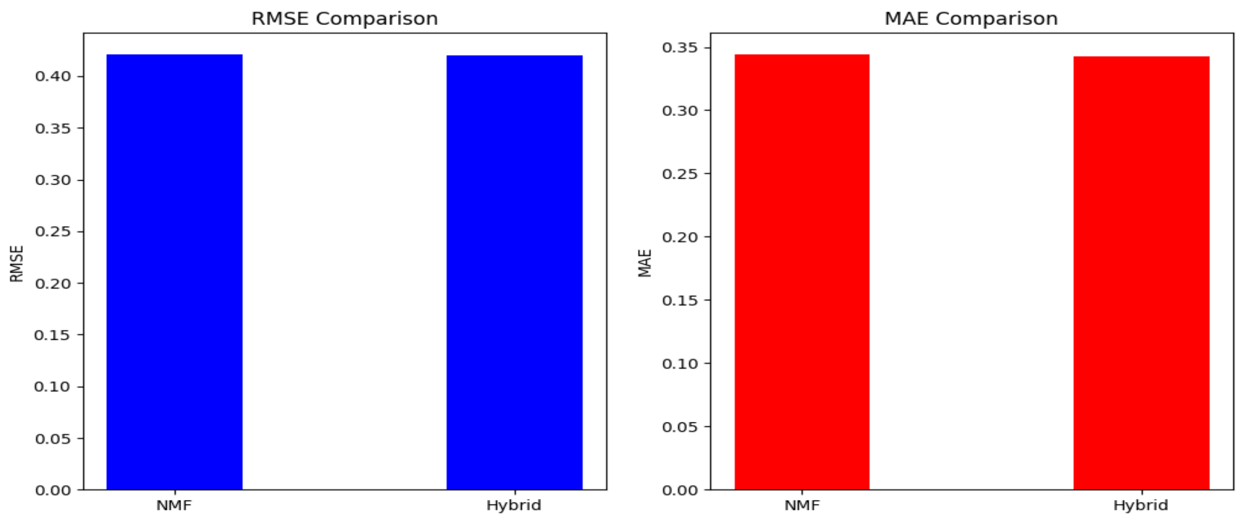


Figure 11: RMSE & MAE Comparison for Both Models

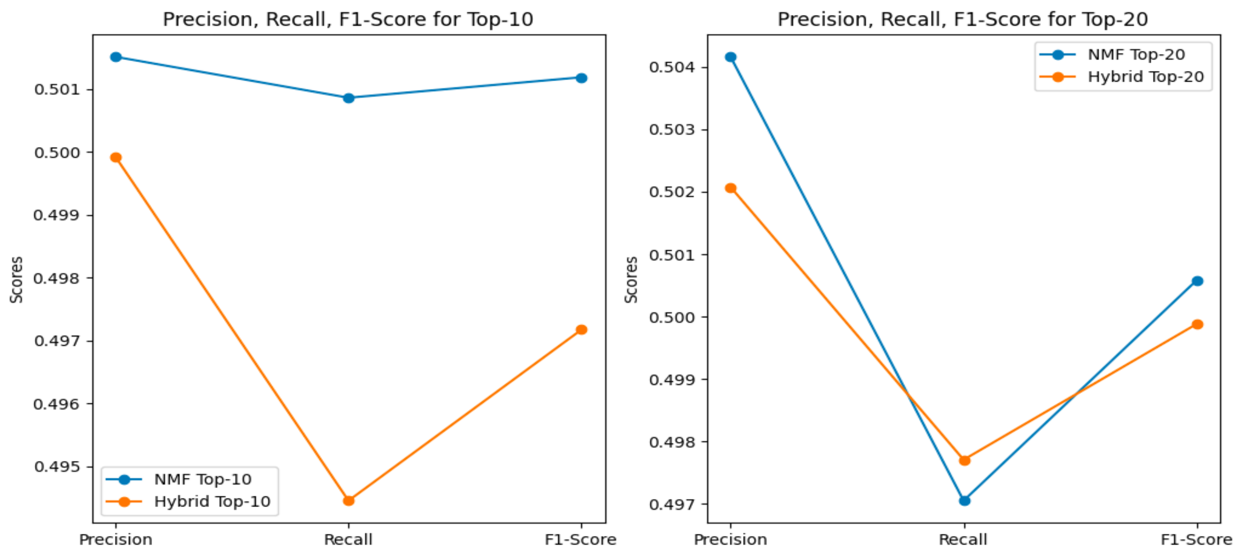


Figure 12: Precision, Recall & F1-score Comparison for Top 10 & Top 20 Recommendations

3. Analysis and Discussion

Rating Prediction Accuracy:

RMSE and MAE are metrics that measure the accuracy of the predicted ratings against the true ratings. Both models perform similarly, with the Hybrid model having a slightly lower RMSE (0.4196) and MAE (0.3428) compared to the NMF model (RMSE: 0.4207, MAE: 0.3439). This indicates that the Hybrid model has a marginally better accuracy in predicting ratings.

4. Recommendation Relevance:

Precision measures the proportion of relevant items among the recommended items. Recall measures the proportion of relevant items that have been recommended out of all relevant items.

For top-10 recommendations, the NMF model has slightly higher precision (0.5012) and recall (0.5023) compared to the Hybrid model (precision: 0.4981, recall: 0.4979). Consequently, the F1-Score of the NMF model (0.5017) is marginally higher than that of the Hybrid model (0.4980).

For top-20 recommendations, the metrics for both models are very close, with the Hybrid model showing a slight advantage in recall (0.5015) and F1-Score (0.5007), while the NMF model has a marginally higher precision (0.5005).

Considering that a movie recommendation system typically benefits more from providing users with relevant recommendations (i.e., movies they are likely to enjoy), the NMF model may be slightly more favorable due to its higher precision and recall for top-10 recommendations. However, the differences are quite small, and both models perform similarly well. The choice can also depend on other factors, such as computational efficiency and ease of integration into the existing system.

IV. Conclusion

This report evaluated and compared two recommendation models: the Non-Negative Matrix Factorization (NMF) model and the Hybrid recommender system, using key performance metrics. The NMF model demonstrated slightly superior accuracy in predicting ratings and higher precision and recall for top-10 recommendations, making it more effective in delivering relevant recommendations. Conversely, the Hybrid model showed marginal improvements in recall and F1-Score for top-20 recommendations, indicating its potential for broader user interaction. Overall, while both models performed well, the choice between them should consider specific use cases and system requirements, including precision, recall, and computational efficiency. Future enhancements could focus on incorporating additional features and advanced algorithms to further improve recommendation accuracy and user satisfaction.

References

[1]. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017, April). Neural collaborative filtering. In Proceedings of the 26th international conference on the world wide web (pp. 173-182).

[2]. Non-negative Matrix Factorization: a Comprehensive Review Yu-Xiong Wang, Student Member, IEEE, Yu-Jin Zhang, Senior Member, IEEE

[3]. Semantic Nonnegative Matrix Factorization with Automatic Model Determination for Topic Modeling . 2020 19th IEEE International Conference on Machine Learning and Applications doi:10.1109/icmla51294.2020.00060

[4]. Hybrid Recommendation System with Enhanced Generalized Sequential Pattern Algorithm for ELearning System DOI: 10.1109/OTCON56053.2023.10114040

[5]. A Hybrid Recommender System for Improving Rating Prediction of Movie Recommendation DOI: 10.1109/JCSSE54890.2022.9836257

[6]. A Novel Time-Aware Food Recommender-System Based on Deep Learning and Graph Clustering DOI: 10.1109/ACCESS.2022.3175317

[7]. Improvement to Recommendation system using Hybrid techniques DOI: 10.1109/ICACITE53722.2022.9823879

[8]. A Hybrid Recommender System Based on Autoencoder and Latent Feature Analysis https://doi.org/10.3390/e25071062

[9]. A Smart Recommender Based on Hybrid Learning Methods for Personal Well-Being Services doi: 10.3390/s19020431

[10]. HRSPCA: Hybrid recommender system for predicting college admission DOI: 10.1109/ISDA.2012.6416521