Hybrid Recommendations: Integrating Neural Collaborative Filtering and Content-Based Methods with Interactive Chatbots

Kal Pandit
Rutgers University
kal.pandit@rutgers.edu

Nile Kolenovic
Rutgers University
nkk41@scarletmail.rutgers.edu

Aaditya Rayadurgam
Rutgers University
ar1873@scarletmail.rutgers.edu

ABSTRACT

Recommender systems have become a key component of digital services, providing hyper-personalized insights across domains such as e-commerce and media platforms. This study presents a hybrid movie recommender system that combines collaborative filtering with content-based filtering, addressing challenges like the cold-start problem and dataset sparsity. The proposed method leverages neural collaborative filtering and cosine similarity on movie genres and is evaluated on the MovieLens 100K dataset using RMSE, MAE, Precision, and Recall. Additionally, we propose a chatbot user interface to provide users with an interactive endpoint with which they can view recommendations (and inverse recommendations) based on our model.

KEYWORDS

Recommender Systems, Collaborative Filtering, Content-Based Filtering, Hybrid Models, MovieLens Dataset

1 INTRODUCTION

Recommender systems have emerged as a key component of digital services, providing users with hyper-personalized and valuable insights across various domains, like in e-commerce and media platforms. These systems have proven particularly valuable when users are faced with navigating through extensive sets of content or items, where traditional browsing methods may become cumbersome or tedious for a user [1]. One impactful application is in the area of movie recommendation, where these systems assist users in navigating vast catalogs of films by predicting their preferences. There is a large array of movies that viewers can watch, along with a wide range of habits and tastes that viewers can have. With this, recommender algorithms can leverage massive sets of user interaction and rating data, as well as data about movies themselves (like genres), to personalize user experiences and provide them with realistic recommendations.

However, recommending movies to users is not without its challenges. New users who are just starting on a platform can be affected by the cold start problem, where there is sparse data on a specific user to the point where a system may not be able to produce effective recommendations. Moreover, movies that are completely new or unrated may not be recommended to users if popularity is a dependency. Traditional approaches such as collaborative filtering and content-based filtering have demonstrated their use, but they often face limitations such as the cold start problem and sparsity in user-item interaction data. To address these challenges, hybrid models have been proposed, integrating multiple sources of information to fill in gaps from user or movie data [7].

We present a hybrid approach to building a movie recommender system that integrates user ratings and movie genres to provide recommendations. Our proposed method leverages cosine similarity on movie genres and neural, user-based collaborative filtering based on ratings, to address the limitations of traditional approaches. We evaluate and build the system on the MovieLens 100K dataset [4] and use RMSE, MAE, Precision and other metrics against a test set to demonstrate effectiveness.

Additionally, we combine this approach with an interactive chatbot that allows users to see movies they would like or dislike, as well as similar movies they should watch given a movie that they have just watched.

Next, we will review related work, formalize the problem, describe our intervention, and discuss results and future directions.

2 PROBLEM FORMALIZATION

In this section, we mathematically formalize the problem of movie recommendation. Given a set of users $U = \{u_1, u_2, \ldots, u_m\}$ and a set of movies $I = \{i_1, i_2, \ldots, i_n\}$, the objective is to predict a user u's preference for a movie i using their historical interactions and the movie attributes. The observed ratings matrix $R \in \mathbb{R}^{m \times n}$ is sparse, and we leverage both collaborative filtering and content-based approaches to address this sparsity.

Given a set of users $U = \{u_1, u_2, \ldots, u_m\}$ and a set of movies $I = \{i_1, i_2, \ldots, i_n\}$, the goal is to predict the preference of a user u for a movie i based on historical interactions. Let $R \in \mathbb{R}^{m \times n}$ represent the ratings matrix, which is sparse. We employ Bayesian normalization to handle sparsity:

$$\text{Bayesian Rating} = \frac{v_i R_i + m R_{\text{global}}}{v_i + m},$$

where v_i is the count of ratings for movie i, R_i is the mean rating, R_{global} is the global mean, and m is a smoothing parameter (set to the median count of ratings).

To handle variability in user rating scales, we normalize Bayesian ratings using Z-scores against standard deviation of ratings, as follows:

$$\text{normalized rating} = \frac{\text{rating} - \text{bayesian rating}}{\sigma_{\text{rating}} + 10^{-8}}$$

Conference'17, Washington, DC, USA
© 2016 ACM. 978-x-xxxx-xxxx-x/YY/MM...$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

We applied these transformations to ensure consistent scaling against all ratings and balance any biases in user-specific rating distributions.

For content-based recommendations, each movie i is represented as a binary feature vector $x_i \in \mathbb{R}^d$, encoding its genres. The similarity between two movies i and j is computed using cosine similarity:

$$\text{Sim}(x_i, x_j) = \frac{x_i \cdot x_j}{\|x_i\| \|x_j\|}$$

The resulting similarity matrix $S \in \mathbb{R}^{n \times n}$, where $S_{i,j}$ represents the similarity between movies i and j, provides the basis for content-based recommendations.

Collaborative filtering is modeled using Neural Collaborative Filtering (NCF). Each user u and movie i is mapped to some embeddings $e_u, e_i \in \mathbb{R}^k$. The predicted rating is computed via a neural network f_θ that combines these embeddings:

$$\hat{R}_{u,i} = f_\theta(e_u, e_i)$$

Here, f_θ is a function that is trained to minimize the mean squared error (MSE) between the predicted ratings $\hat{R}_{u,i}$ and normalized observed ratings $\text{NormalizedRating}_{u,i}$.

To integrate collaborative and content-based approaches, we define a hybrid scoring function that takes into account user preferences and movie genres:

$$H(u, i) = \alpha \cdot \hat{R}_{u,i} + (1 - \alpha) \cdot S_{i,j}$$

where $\hat{R}_{u,i}$ is the NCF-predicted rating, $S_{i,j}$ is the average similarity of i to movies rated by u, and $\alpha \in [0, 1]$ balances the contributions of both methods.

We discuss approaches used in similar work in the following section.

3 RELATED WORK

Hybrid recommender systems help address the limitations of isolated methods like collaborative filtering and content-based filtering. While collaborative filtering relies on patterns of user interactions (e.g., ratings or purchases), it often suffers from data sparsity and struggles with the cold-start problem, where limited information about new users or items is available due to new releases or onboardings, or little interactions. Li et al. [6] proposed a hybrid content and collaborative filtering algorithm in which K-means clustering was used alongside a user-feature matrix to address a sparse movies recommendations dataset. Compared with other algorithms, the content-collaborative mixing in this algorithm showed improvements in the mean absolute error (MAE) validity metric compared to other methods. Furthermore, this hybrid approach significantly alleviated the cold-start problem, particularly when applied to large datasets, demonstrating its scalability and robustness.

Other studies have also highlighted the benefits of incorporating metadata to enhance collaborative filtering techniques. Jafri et al. [5] used a hybrid method that uses a weighted prediction score combining item-based similarity and user popularity within a deep learning system, yielding higher precision and performance over baseline algorithms.

Normalization approaches are also promising in the context of providing strong recommendations in sparse, biased data. Cheng et al. used quantile normalization by "mapping a feature value x to its cumulative distribution function , divided into n_q quantiles", resulting in increased accuracy [2].

Prior literature has established hybrid systems as a strong solution for improving recommendation quality in sparse or limited datasets.

3.1 Explainability

Building off the discussion of hybrid systems, recent research has taken an emphasis on enhancing quality of recommendations and trust with users. As Zhang and Chen (2020) highlight, explainable recommender systems intend to provide explanations that drive trust and transparency by providing clear [9] propose employing sentence-based templates in order to identify items that could perform well and poorly and present them to users.

Explainable methods have been used in conjunction with content-based models — both to allow users to determine whether or not a recommendation should be used as well as to provide justifications. de Campos, Luna and Huete (2024) conducted a user study on academic journal recommenders with explanations, and showed word clouds, similar papers, and highlighting to explain which journals to publish to [3]. The explainer serves to justify recommendations given information from the content-based model, and we adopt this approach in our work.

4 PROPOSED METHOD

In the collaborative filtering component, we use a Neural Collaborative Filtering (NCF) model to predict the ratings $\hat{r}_{u,m}$ for a user u and a movie m. Let R denote the user-movie rating matrix of size $|U| \times |M|$, where $|U|$ is the total number of users, and $|M|$ is the total number of movies. The majority of entries in R are unknown, and the goal is to estimate the missing values.

4.1 Collaborative Filtering with NCF

Instead of traditional nearest-neighbor approaches, we model $\hat{r}_{u,m}$ using learned latent representations (embeddings) of users and movies. We say that $\mathbf{e}_u \in \mathbb{R}^d$ and $\mathbf{e}_m \in \mathbb{R}^d$ represent the embeddings for user u and movie m, respectively. It follows that these embeddings are learned as part of a neural network architecture. The predicted rating is defined as:

$$\hat{r}_{u,m} = f(\mathbf{e}_u, \mathbf{e}_m; \Theta),$$

where f is a multi-layer neural network parameterized by Θ. In the implementation, the embeddings $\mathbf{e}_u$ for users and $\mathbf{e}_m$ movies are concatenated and passed through dense layers, which is defined as follows:

$$\mathbf{z}_1 = \text{ReLU}(\mathbf{W}_1[\mathbf{e}_u; \mathbf{e}_m]),$$

$$\mathbf{z}_2 = \text{ReLU}(\mathbf{W}_2\mathbf{z}_1),$$

$$\hat{r}_{u,m} = \mathbf{W}_3\mathbf{z}_2,$$

where $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3$ and $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ are weight matrices and biases, and $[\mathbf{e}_u; \mathbf{e}_m]$ represents the embeddings being concatenated. The model is trained to minimize the Mean Squared Error (MSE) loss, taking into account the space of observed user-movie ratings in the training data.

4.2 Content-Based Filtering

To incorporate content-based filtering, we use the genre information associated with each movie. Let $\mathbf{g}_m \in \{0,1\}^k$ represent the binary genre vector for movie m, where k is the total number of unique genres. The similarity between two movies m_1 and m_2 is calculated using cosine similarity:

$$\text{sim}_{\text{content}}(m_1, m_2) = \frac{\mathbf{g}_{m_1} \cdot \mathbf{g}_{m_2}}{\|\mathbf{g}_{m_1}\| \|\mathbf{g}_{m_2}\|}.$$

For a user u, we compute the content-based score for an unrated movie m by aggregating the similarity scores between m and the movies rated by u. Let I_u be the set of movies rated by u. The content-based score is given as:

$$\text{Content Score}_{u,m} = \frac{1}{|I_u|} \sum_{m' \in I_u} \text{sim}_{\text{content}}(m, m').$$

4.3 Normalization and Hybrid Scoring

To ensure consistent ratings, Bayesian normalization is applied to the raw ratings before training the model. For a movie m, the Bayesian-adjusted rating is defined as:

$$\text{Bayesian Rating}_m = \frac{v_m R_m + m R_{\text{global}}}{v_m + m},$$

where R_m is the mean rating for movie m, v_m is the number of ratings for m, R_{global} is the global mean rating across all movies, and m is a smoothing parameter. The Bayesian ratings are further transformed into a normalized distribution using a Z-score against standard deviation.

The final hybrid recommendation score for a user u and an unrated movie m combines the NCF prediction and content-based similarity:

$$s_{u,m} = \alpha \cdot \hat{r}_{u,m} + (1 - \alpha) \cdot \text{ContentScore}_{u,m},$$

where $\alpha \in [0,1]$ is a hyperparameter that controls the trade-off between the two components.

4.4 Evaluation Metrics

The performance of the system is evaluated using standard metrics. Precision, Recall, and F-Measure assess the relevance of recommendations, while Normalized Discounted Cumulative Gain (NDCG) evaluates the ranking quality.

Precision measures the fraction of recommended movies that are relevant:

$$\text{Precision} = \frac{|\hat{R}_u \cap R_u|}{|\hat{R}_u|},$$

where $\hat{R}_u$ is the set of recommended movies for user u, and R_u is the set of relevant movies.

Recall measures the fraction of relevant movies that are recommended:

$$\text{Recall} = \frac{|\hat{R}_u \cap R_u|}{|R_u|}.$$

The **F-Measure** is the harmonic mean of Precision and Recall:

$$\text{F-Measure} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

Finally, **NDCG** evaluates the quality of rankings by assigning higher importance to relevant movies appearing at higher ranks:

$$\text{NDCG} = \frac{\sum_{i=1}^{K} \frac{1}{\log_2(i+1)} \cdot \mathbb{I}(m_i \in R_u)}{\sum_{i=1}^{K} \frac{1}{\log_2(i+1)} \cdot \mathbb{I}(m_i \in R_u^{\text{ideal}})},$$

where m_i is the movie at rank i, and R_u^{ideal} is the ideal ranking of relevant movies.

5 EXPERIMENTS

We conducted a series of experiments on the MovieLens 100K dataset to evaluate the impact of different configurations and algorithms on recommendation metrics such as Precision, Recall, F-Measure, Normalized Discounted Cumulative Gain (NDCG), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE). The experiments included comparisons between the base collaborative filtering model, a binary collaborative filtering process, and the normalized, hybrid approach: one with L2 regularization, one without.

5.1 Processing

The dataset was split into training and testing sets, with an 80/20 split. For each experiment, recommendations were evaluated against the testing set to determine metrics like precision, which measures the number of items actually watched against those recommended.

The collaborative filtering model was trained using TensorFlow, with embedding layers representing users and movies. These embeddings were followed by dense layers with 128, 64, and 32 neurons, respectively. ReLU activation functions were used for all layers, and the output layer produced a single predicted rating. During training, the Mean Squared Error (MSE) loss function and the Adam optimizer were used with a learning rate of 10^{-3}. Training was conducted over 10 epochs with a batch size of 256. User and movie embeddings were represented in a 50-dimensional space.

A genre-based similarity matrix was precomputed using cosine similarity for the content-based filtering component. During the training phase, these similarity scores were not directly used by the model but were combined with the collaborative predictions during the hybrid scoring step.

5.2 Results

5.2.1 Base Collaborative Filtering Model. Without normalization, the collaborative filtering model yielded relatively modest performance, with a precision metric below 0.01. The RMSE was 0.9501. The raw ratings were not suitable for this data, as most ratings were skewed toward higher values (e.g., 4.0 and above), limiting the model's ability to distinguish relevant recommendations.

5.2.2 Binary Collaborative Filtering. To address the limitations of raw ratings, we employed a binary classification model by marking ratings of 3 or above as positive. We did not need to normalize results, and we used binary cross-entropy as a loss metric. We saw the following results.

- **MAE:** 0.7081
- **RMSE:** 0.9367
- **Precision:** 0.0077
- **Recall:** 0.0018
- **NDCG:** 0.0044
- **F-Measure:** 0.0030

Moreover, the binary collaborative model had a binary accuracy of over 75% across training epochs, as measured during the training process.

5.2.3 Hybrid Model: No Normalization. The hybrid system combined Bayesian ratings, user-based collaborative filtering, and content-based scoring using cosine similarity. To balance the contributions of the two components, collaborative predictions were weighted at 70% and content-based scores at 30%. The final score was computed as:

$$s_{u,m} = \alpha \cdot \hat{r}_{u,m} + (1 - \alpha) \cdot \text{ContentScore}_{u,m},$$

where $\alpha = 0.7$. This hybrid approach yielded improved results across all metrics, but with modest impact, as follows:

- **MAE:** 0.0136
- **RMSE** 0.0412
- **Precision:** 0.0633
- **Recall:** 0.0346
- **NDCG:** 0.0704
- **F-Measure:** 0.0351

For MAE and RMSE, the ratings used as labels for training were normalized to ensure consistency.

5.2.4 Hybrid Model: With Normalization. The hybrid system combined Bayesian ratings, user-based collaborative filtering, and content-based scoring using cosine similarity. To balance the contributions of the two components, collaborative predictions were weighted at 70% and content-based scores at 30%. The final score was computed as shown above.

With L2 regularization on the model itself combined with our own normalized scores, we saw the below results:

- **MAE:** 0.6266
- **RMSE:** 0.8269
- **Precision:** 0.0043
- **Recall:** 0.0013
- **NDCG:** 0.0046
- **F-Measure:** 0.0014

Without it, we saw an increase in precision to 1%, but a significant increase in validation loss indicating the model could not generalize.

In general, we see significant, viable decreases in MAE and RMSE, but the model suffers with precision, recall, and other hit measures.

5.3 Observations

From these experiments, we observe the following:

- Collaborative filtering without normalization was modest for data with skewed ratings distributions.
- Binary collaborative filtering improves performance but is still limited compared to the hybrid approach.
- The hybrid system addresses the limitations of both collaborative and content-based filtering with reduced error but lower precision and hit measures.

We discuss the limitations of these results in the Challenges section, and describe the focus toward explainability that the hybrid-based model has.

6 USER-FACING OUTPUTS

We incorporate two template-based methods of interfacing with our algorithm. More formally, given either a user title or a movie name, a list of movies is displayed to the user along with reasoning. This serves as an endpoint for users, acting as preliminary work toward future development of a recommendation system.

We chose the hybrid model so we could provide more concrete scores, while also providing qualitative information like genres and similarities.

6.1 Cosine Similarity: Because You Watched

We employ cosine similarity to implement a "Because You Watched" feature. Given a movie, this feature provides a list of similar movies based on genre similarity rather than ratings. Let $M = \{m_1, m_2, \ldots, m_n\}$ denote the set of movies in the dataset. Each movie m_i is represented by a binary genre vector $\mathbf{g}_{m_i} \in \{0, 1\}^k$, where k is the total number of unique genres. The entry $g_{i,j}$ indicates whether a movie m_i belongs to genre j.

The similarity between two movies m_i and m_j is calculated using cosine similarity:

$$\text{sim}(m_i, m_j) = \frac{\mathbf{g}_{m_i} \cdot \mathbf{g}_{m_j}}{\|\mathbf{g}_{m_i}\| \|\mathbf{g}_{m_j}\|},$$

where $\mathbf{g}_{m_i} \cdot \mathbf{g}_{m_j}$ represents the dot product of the two genre vectors, and $\|\mathbf{g}_{m_i}\| = \sqrt{\sum_{l=1}^{k} g_{i,l}^2}$ is the Euclidean norm of $\mathbf{g}_{m_i}$.

The algorithm retrieves an index, computes similarities, ranks movies in descending order of similarity scores, and selects the top 10 recommendations. This feature provides a user-facing interface for exploring similar movies, though it is not a core part of the hybrid recommendation model.

6.2 Template-Based Recommendations

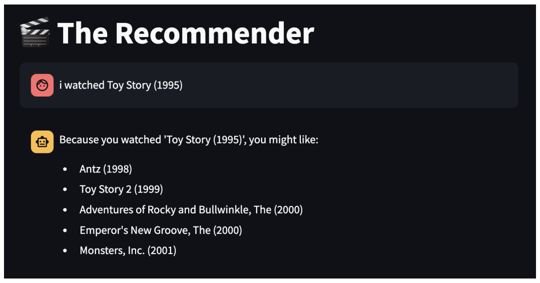
In our hybrid approach, we recommend movies to a single user by focusing on genres for user-facing explanations. This algorithm provides transparent reasoning about why a movie is recommended (or not recommended).

We identify movies rated as 4 or above by the user for positive recommendations and those rated below 3 for disrecommendations. The hybrid scoring approach described in Section 4 is used to compute weighted scores. Explanations are generated by comparing the recommended movies to the user's preferred genres. The "User Interface" section shall describe this approach further.

6.3 User Interface

A user interface was developed using Streamlit to allow users to interact with the recommendation system.

This interface works similarly to the backend script but provides interactive options for users to type prompts and interact with a



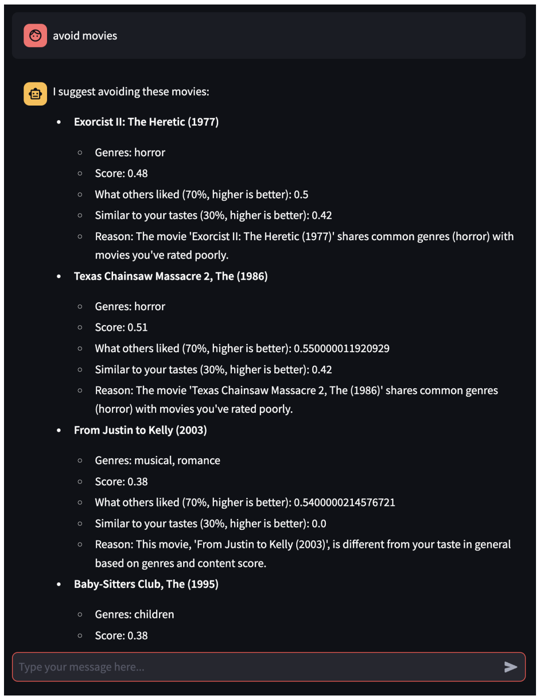
The Recommender

i watched Toy Story (1995)

Because you watched 'Toy Story (1995)', you might like:

- Antz (1998)
- Toy Story 2 (1999)
- Adventures of Rocky and Bullwinkle, The (2000)
- Emperor's New Groove, The (2000)
- Monsters, Inc. (2001)

Figure 1: Movies recommended as a result of similar genres to a pre-watched movie.



avoid movies

I suggest avoiding these movies:

- **Exorcist II: The Heretic (1977)**
  - Genres: horror
  - Score: 0.48
  - What others liked (70%, higher is better): 0.5
  - Similar to your tastes (30%, higher is better): 0.42
  - Reason: The movie 'Exorcist II: The Heretic (1977)' shares common genres (horror) with movies you've rated poorly.
- **Texas Chainsaw Massacre 2, The (1986)**
  - Genres: horror
  - Score: 0.51
  - What others liked (70%, higher is better): 0.550000011920929
  - Similar to your tastes (30%, higher is better): 0.42
  - Reason: The movie 'Texas Chainsaw Massacre 2, The (1986)' shares common genres (horror) with movies you've rated poorly.
- **From Justin to Kelly (2003)**
  - Genres: musical, romance
  - Score: 0.38
  - What others liked (70%, higher is better): 0.5400000214576721
  - Similar to your tastes (30%, higher is better): 0.0
  - Reason: This movie, 'From Justin to Kelly (2003)', is different from your taste in general based on genres and content score.
- **Baby-Sitters Club, The (1995)**
  - Genres: children
  - Score: 0.38

Figure 2: Movies being disrecommended.

chatbot-like feature. The interface enhances personalization and provides an operational endpoint for the recommendation system, allowing us to use a foundation for further work.

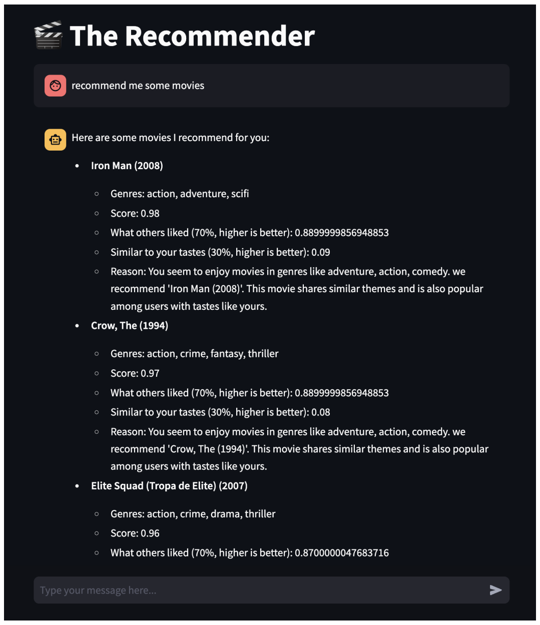
The user can prompt the chatbot and it will respond based on keyword inclusion: ”i watched” will provide movies the user should watch based on genre, given the movie they just watched; a statement containing ”avoid” provides anti-recommendations (movies someone should not watch).

A statement containing the affirmative ”would recommend” would result in recommendations being displayed with reasoning.

For efficiency, we use the training data as a prototype.

7 CHALLENGES

We attribute the low results in this dataset and model to a limited, sparse dataset used for testing. There are movies that have no ratings, as well as users with little to none, and this low density



The Recommender

recommend me some movies

Here are some movies I recommend for you:

- **Iron Man (2008)**
  - Genres: action, adventure, scifi
  - Score: 0.98
  - What others liked (70%, higher is better): 0.8899999856948853
  - Similar to your tastes (30%, higher is better): 0.09
  - Reason: You seem to enjoy movies in genres like adventure, action, comedy. we recommend 'Iron Man (2008)'. This movie shares similar themes and is also popular among users with tastes like yours.
- **Crow, The (1994)**
  - Genres: action, crime, fantasy, thriller
  - Score: 0.97
  - What others liked (70%, higher is better): 0.8899999856948853
  - Similar to your tastes (30%, higher is better): 0.08
  - Reason: You seem to enjoy movies in genres like adventure, action, comedy. we recommend 'Crow, The (1994)'. This movie shares similar themes and is also popular among users with tastes like yours.
- **Elite Squad (Tropa de Elite) (2007)**
  - Genres: action, crime, drama, thriller
  - Score: 0.96
  - What others liked (70%, higher is better): 0.8700000047683716

Figure 3: Movies being recommended.

makes it difficult to come up with recommendations in those cases. Additionally, our content score is simplistic: it only groups together similar movies based on genres. While we found this viable for explanations (as discussed below), it needs further study.

Lastly, we used a heuristic to tune the weight parameter that was largely based on results from experiments. The chosen balance of 70% collaborative to 30% content may not be ideal for all users or scenarios, and needs refinement.

7.1 Biases and Review-Bombing

We used a smaller dataset ($n = 100,000$ movies, 650 users) with a naive algorithm to split the training and testing data. This approach made it challenging to account for biases in the algorithm, both within the movies themselves and in the users' content preferences. This algorithm faces similar pitfalls to those noted by Jafri et al. (2023) [5], where factors like location, gender, and demographics are not included in the data. Our data suffered from these limitations, as some movie titles were in languages other than English (ex: French). This is critical for foreign-language movies, as a movie solely in French may not be reviewed or received well by English-speaking viewers in the United States.

Certain movies with a large number of ratings can become outliers, particularly when they are popular, highly-anticipated blockbusters. Additionally, adversarial behavior such as review-bombing—where users provide negative reviews on media as a reactionary measure to external circumstances or ideological positions—can further distort the recommendation system. Indicators of trolling often include negative words and expressions of anger [8]. It is plausible for trolling to be conveyed in reviews, and it should be interpreted as an attack on the movie. The MovieLens data is neither sufficient nor intended for this purpose.

Our algorithm currently mitigates biases by taking the Bayesian average of reviews for a movie. This ensures that movies with fewer ratings are not penalized for sparsity or not being a blockbuster. However, future work is needed to analyze reviews for trolling and address their effects on recommendations.

7.2 Dataset Sparsity and Cold-Start

The dataset's sparsity, where many users and movies have few ratings, made it difficult to test the algorithm and generate accurate predictions, especially against the testing set. Moreover, ratings were skewed towards 4-5 stars, so we could not necessarily work with items users did not like and this can lead to predictions being clustered in higher ratings.

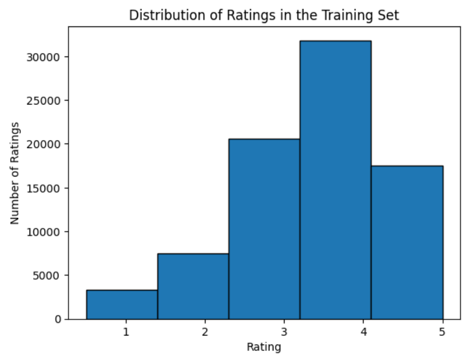


Figure 4: Ratings are skewed towards positive stars.

7.3 Overfitting

The recommendation system struggled markedly with overfitting, where the model performed well on the training data but failed to generalize to unseen data. The training loss separated from the validation loss, such that the validation loss increased. Future work should focus on more robust validation to mitigate overfitting issues, as well as potentially working with embedding sizes to find the most optimal.

We mitigated this using L2 regularization.

7.4 Discovery

A significant limitation of this algorithm, and recommender systems at large, is their focus on finding *relevant* items rather than enabling users to discover items outside their narrow range of preferences. This confirmatory behavior prevents users from exploring new genres or discovering movies that may not align with their immediate tastes. Future iterations of the algorithm should incorporate mechanisms to encourage discovery and novelty.

7.5 Explainability

The current implementation of our chatbot uses relatively trivial parsing to output sentence-based templates. As users can enter any plausible prompt (ex: "I just went to the movies and saw X" vs. "i watched X"), the chatbot needs to be able to recognize a range of prompts and understand user intents, while also providing more specific recommendations. Our code does not handle input parsing beyond key phrases.

8 CONCLUSION AND FURTHER DIRECTIONS

This study leverages NCF to address sparsity and cold-start issues while providing user-facing endpoints for system interaction. By combining collaborative filtering and content-based recommendations in a chatbot, we provide user-focused ratings on a sparse dataset with a framework for further interaction.

The recommendations are enhanced through a normalization pipeline and weighted scoring to produce interpretable predictions. We further integrate a chatbot-like interface, enabling users to interact with the recommendation system. This interface balances personalization and operationalizes the system so that users can more concretely interact with this system. While the current implementation uses template-based messages, future work could expand on this to create abstract explanations and potentially deeper conversational interfaces.

The most pressing and limitations still include addressing biases and cold-start issues. While content-based weighting alleviates some aspects, reliance on genre similarity may not generalize to niche or uncommon genres. Additionally, the skewed distribution of ratings, which heavily favors positive reviews, suggests that users often review movies they enjoy rather than those they dislike and can lead to discrepancies in overall metrics.

To expand on this work, future directions include conducting sentiment analysis and topic modeling of reviews to better understand user tastes beyond genres. Encouraging discovery of new genres or tastes is another critical area, aiming to reduce the "echo chamber" effect where recommendations reinforce existing preferences. Furthermore, understanding the context behind certain movies—such as fanbases or susceptibility to trolling—could enhance recommendation quality.

This system serves as a foundation for transparent, user-centered interactions that account for content details and user behaviors. Its further directions entail improving prediction accuracy, inclusivity in recommendations, and the transparency of reasoning provided to users.

REFERENCES

[1] R. Burke, A. Felfernig, and M. H. Göker. 2011. Recommender Systems: An overview. *AI Magazine* 32, 3 (2011), 13–18. DOI: http://dx.doi.org/10.1609/aimag.v32i3.2361

[2] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, and others. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems.* DOI: http://dx.doi.org/10.1145/2988450.2988454

[3] L. M. de Campos, J. M. Fernández-Luna, and J. F. Huete. 2024. An explainable content-based approach for recommender systems: A case study in journal recommendation for paper submission. *User Modeling and User-Adapted Interaction* 34, 4 (2024), 1431–1465. DOI: http://dx.doi.org/10.1007/s11257-024-09400-6

[4] GroupLens. 1998. MovieLens 100K Dataset. (1998). http://grouplens.org/datasets/movielens/100k/

[5] S. I. Jafri, R. Ghazali, I. Javid, Y. Mazwin Mohmad Hassim, and M. Hayat Khan. 2023. A hybrid solution for the cold start problem in recommendation. *Comput. J.* 67, 5 (2023), 1637–1644. DOI: http://dx.doi.org/10.1093/comjnl/bxad088

[6] L. Li, Z. Zhang, and S. Zhang. 2021. Hybrid algorithm based on content and collaborative filtering in recommendation system optimization and simulation. *Scientific Programming* 2021 (2021), 1–11. DOI: http://dx.doi.org/10.1155/2021/7427409

[7] A. Panteli and B. Boutsinas. 2023. Addressing the cold-start problem in recommender systems based on frequent patterns. *Algorithms* 16, 4 (2023), 182. DOI: http://dx.doi.org/10.3390/a16040182

[8] D. Schuff, S. Mudambi, and M.-X. Wang. 2024. Understanding the review bombing phenomenon in movies and television. In *Proceedings of the Annual Hawaii International Conference on System Sciences*. DOI: http://dx.doi.org/10.24251/hicss.2023.036

[9] Y. Zhang and X. Chen. 2020. Explainable recommendation: A survey and new perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101. DOI: http://dx.doi.org/10.1561/1500000066