

Final Report - MapMyMovies

Team 23: Emily Mehigan, Nathan Popper, Hannah Quintal, Anand Subudhi, Nikhita Vasan
{emehigan3, npopper3, hquintal6, asubudhi6, nvasan7}@gatech.edu

1 Introduction

Like most products of the Internet era, modern streaming services can induce information overload. With countless options at our fingertips, users often struggle to identify content that aligns most with their interests. Recommendation algorithms have become essential in addressing this problem, improving user experience by curating content they are most likely to enjoy.

However, modern recommendation systems often operate as black boxes, providing limited transparency or user control. Our project aims to create a more transparent movie recommendation system - one that not only generates accurate and personalized suggestions but also incorporates an interactive visualization to help users explore and understand their recommendations. To achieve this, our system will feature an interactive graphical model [11], allowing for actions such as node dragging and event-triggered visuals. By developing a novel visualization that illustrates how user preferences, related content, and similar users contribute to generating recommendations, we aim to build trust and enhance the overall user experience.

2 Problem Definition

More formally, the problem we aim to address is twofold:

1. How can we generate accurate and personalized movie recommendations?
2. How can we use interactive visualizations to present recommendations in a way that is intuitive and transparent, enhancing user trust, engagement, and understanding?

3 Literature Survey

3.1 Explaining Recommendations

Recent research has examined ways to use interactive visuals to enhance the explainability of recommendation systems. He et al. [8] survey a variety of interactive recommender systems, exploring how different platforms improve explainability through transparency, recommendation justification, and user controllability. Another framework [4] identifies key components of successful interactive explanations: goal, scope, style, and format.

An effective explanation of recommendations should follow principles outlined by Herlocker et al. [10], matching the user's cognitive understanding of how recommendations are generated. First, users should understand how their personal data and preferences influence the recommendations they receive. Services like MovieMine [12]

and DecCM [18], which use both search queries and user data, could improve trust by clarifying how this information is utilized. Second, users should be able to inspect their neighborhood such that they can identify why they are similar to other users. Finally, the system should clearly present the data that justifies each recommendation, enhancing transparency and confidence. These principles will guide our visual approach, ensuring the final product is both informative and intuitive.

3.2 Recommender System Methods

Ko et al. [13] discuss the evolution of recommender systems in the streaming services space, tracing the progression from content-based filtering, to collaborative filtering, and most recently, to hybrid models that integrate deep learning-based architectures. Ahuja and Gong et al. further explore the advantages of combining user-based collaborative filtering, item-based collaborative filtering, and content-based filtering into a unified system [2, 5]. The efficiency and interpretability of such hybrid approaches provide significant benefits over traditional collaborative filtering methods.

Koren introduces matrix factorization, proposing the idea that the sparse user-item matrix generated in collaborative filtering can be decomposed into smaller, denser matrices that capture latent features [14]. However, this approach relies on dot products to model user-item interactions, limiting its expressivity and scalability. Building on this foundation, He et. al. develop neural collaborative filtering, a deep learning-based variant that combines a generalized form of matrix factorization and multi-layer-perceptrons to model non-linearities in user-item interactions [9]. Although originally designed for implicit feedback systems, the architecture can be adjusted to include explicit rating prediction capabilities.

Katarya and Verma discuss their movie recommendation system on the MovieLens dataset that uses k-means clustering with a specialized optimization algorithm to dynamically adjust cluster assignments. Ventocilla et. al. further cements the utility of clustering algorithms from a visualization perspective, demonstrating how multidimensional projections can elucidate relationships in the data [17].

3.3 Interactive Recommender Systems

Previous implementations of interactive recommender systems provide valuable insights we can use to guide our own work.

SmallWorlds [6] represents user-based collaborative filtering through a horizontally linked graph with five distinct layers, improving interpretability and reducing clutter. Users can improve recommendations by dragging nodes that adjust their weight, but relying on binary interest indicators limits understanding. We address this by incorporating rating scales and specific movie data to help further organize nodes by similarity for deeper insights.

PeerChooser [16] arranges user nodes in a circular layout by similarity, offering a clear view of user neighborhoods. However, its genre-focused connections simplify visualization at the cost of movie-specific insights. Like SmallWorlds, it allows weight adjustments via node dragging and enhances exploration with interactive profile previews.

TasteWeights [3] uses a horizontal layout to connect users to recommendations through an intermediate layer that adds transparency. While it enables preference adjustments via sliders, it lacks node-dragging interactivity and does not visually encode relationship strength or topological information. These studies all demonstrate that interactive visualizations enhance transparency, user engagement, and recommendation accuracy.

4 Proposed Method

4.1 Dataset

The primary dataset used in this project is the MovieLens Latest Dataset [7], which contains approximately 33 million movie ratings provided by over 300,000 users. Each row in the dataset consists of a user ID, a movie ID, and the corresponding rating assigned by the user. For our project, we used a smaller version of this dataset that contains 200k records, which can be found on GroupLens.

To structure the data for traditional collaborative filtering, we transform it into a user-item interaction matrix. Specifically, we pivot the dataset to create a matrix where each row represents a unique user, each column corresponds to a specific movie, and each entry $a_{i,j}$ denotes the rating given by user i to movie j . For neural collaborative filtering, we create a training set of (user, item, interaction) tuples, which can be batched and fed into the model.

4.2 Recommender Systems

The goal of our project from a computational perspective is to increase the expressivity of traditional collaborative filtering models. One strategy is to keep the algorithms the same but combine their results in a meaningful way, similar to how ensemble methods leverage the strengths of multiple models to generate better predictions. Another strategy is to employ a model that can capture more complex or non-linear patterns inherently through deep learning. Through the models described in the proceeding sections, we test both these intuitions.

4.2.1 User-Based & Item-Based Systems

User-based collaborative filtering (UBCF) recommends items by identifying users with similar preferences to the target user and suggesting items that those similar users have rated highly.

The similarity between two users, u and v , is computed using the cosine similarity metric:

$$\text{sim}(u, v) = \frac{\sum_{i \in I_{uv}} r_{u,i} r_{v,i}}{\sqrt{\sum_{i \in I_{uv}} r_{u,i}^2} \sqrt{\sum_{i \in I_{uv}} r_{v,i}^2}} \quad (1)$$

where I_{uv} is the set of items rated by both users u and v , and $r_{u,i}$ represents the rating given by user u to item i . Before calculating the similarity between two users, each user's mean rating is subtracted from their individual ratings. This process helps to normalize user preferences by removing individual biases in rating scale usage.

Predictions for unseen items are computed with a similarity-weighted average of similar users. Since ratings had been mean-centered, we must add the target user's average rating back.

$$\hat{r}_{u,j} = \frac{\sum_{v \in N_u} \text{sim}(u, v) \cdot r_{v,j}}{\sum_{v \in N_u} |\text{sim}(u, v)|} + \bar{r}_u \quad (2)$$

where N_u represents the set of top similar users to user u who have rated item j .

Item-based collaborative filtering (IBCF) is very similar to user-based collaborative filtering except it makes calculations using the transposed user-item matrix. It identifies items that are similar to items the user has already rated. Predictions are generated using a user's previous ratings, weighted by how similar the previous items are to the new item.

4.2.2 Hybrid Systems

Our novel hybrid approach blends user-based and item-based collaborative filtering to generate high quality recommendations. This model uses a tunable parameter α to determine the optimal weighting between both prediction methods:

$$\text{prediction} = \alpha \cdot \text{ubcf_pred} + (1 - \alpha) \cdot \text{ibcf_pred} \quad (3)$$

When $\alpha = 1.0$, the system only leverages user-based predictions, and when $\alpha = 0.0$, only item-based predictions are used. Instead of manually setting α to an arbitrary value, we optimize it through regression, which has a closed form solution. Our model computes both prediction types for each user-item pair and then finds the α value that minimizes the overall squared error using the following equation:

$$\alpha = \frac{\sum (x \cdot y)}{\sum (x \cdot x)} \quad (4)$$

where x is the difference between user-based and item-based predictions and y is the difference between ground truth ratings and item-based predictions. While initially set to minimize error, we give the user full control over this α parameter from our interface.

Within this approach, we introduce another novel addition - a minimum threshold for the number of shared items (co-rated movies in UBCF and shared users in IBCF). Traditional similarity measures are based solely on these shared items, with no existing minimum threshold for the number of items that need to be in common. For example, if a user shares only one co-rated movie with another user, and they rate it the same, the similarity would be considered perfect. This can be misleading, as someone who has rated 20+ movies similarly but with slight variations would be penalized, despite being more valuable for predictions. To address this, we set the minimum number of co-rated movies to 5.

4.2.3 Neural Collaborative Filtering

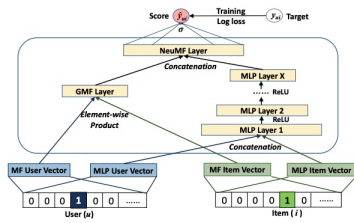


Figure 1: NeuMF Architecture Diagram

Traditional collaborative filtering methods apply fixed similarity functions directly to user and item vectors, which limits their expressivity by modeling interactions in a purely linear fashion. Matrix factorization (MF) [9] partially addresses this limitation by embedding user and item vectors into a latent space, but the latent feature interaction is still modeled linearly using dot products. Neural Collaborative Filtering (NCF) overcomes these limitations by parameterizing the interaction function f through deep learning, allowing for a complex and expressive model.

Neural Matrix Factorization (NeuMF) combines Generalized Matrix Factorization (GMF) with a Multi-Layer Perceptron (MLP) to predict user-item interactions. GMF learns weighted linear interactions by replacing the dot product with a learnable element-wise product, while MLP captures nonlinearities by concatenating user and item embeddings and passing them through hidden layers. As shown in Figure 5, NeuMF merges the output of these two components, enabling it to model both linear and nonlinear patterns in user behavior.

Furthermore, the original model architecture was designed for implicit feedback. Implicit feedback is equivalent to binary interaction data (e.g. 0 if a user has not interacted with an item and 1 otherwise) as opposed to absolute ratings.

The model generates a score indicating the probability of a user interacting with a given item. This is more commonly used in practice, since users rarely rate all content they engage with. However, to maintain continuity with our other models, we modify NeuMF to predict continuous ratings on a 0-5 scale by adjusting the model architecture to handle regression. This includes modifying the loss function to Mean Squared Error (MSE, increasing layer depths, normalizing all input ratings on a [0, 1] scale and multiplying the output of the sigmoid function by the maximum rating to generate an appropriately scaled prediction.

Many recommendation systems also suffer from popularity-bias, where the model lacks exposure to "long tail," or less popular, items and thus recommends a similar set of popular items across all users [1]. To mitigate this, we add a popularity regularization term to our loss function that penalizes more popular items which is formulated as:

$$\mathcal{L} = \mathcal{L}_{base} + \lambda \cdot \mathcal{L}_{pop} \quad (5)$$

where

$$\mathcal{L}_{base} = \frac{1}{|B|} \sum_{(u,i) \in B} (\hat{r}_{ui} - r_{ui})^2 \quad (6)$$

$$\mathcal{L}_{pop} = \frac{1}{|B|} \sum_{(u,i) \in B} \hat{r}_{ui}^2 \cdot \rho(i)$$

\mathcal{L}_{base} is the standard MSE loss and \mathcal{L}_{pop} is the popularity regularization term. Here, λ represents the popularity regularization strength, \hat{r}_{ui} is the predicted rating and ρ_i is the normalized popularity score of item i . The model thus incurs a higher penalty when it predicts high ratings for existing popular items, ideally improving exposure to less popular items during training. Finally, we implement the xQuAD post-processing re-ranking approach to improve content discovery while still customizing recommendations [1]. xQuAD calculates a diversity score for each potential recommendation based on how well different segments of the item catalog are already represented in the current recommendation list, balancing relevance with diversity.

4.3 Visualization

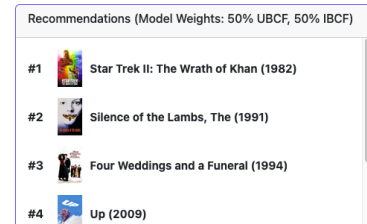


Figure 2: Recommendations for a user

To improve transparency in our recommendation outputs, we build an interactive network visualization using Plotly and Cytoscape. These technologies provide a polished,

professional interface while allowing for extensive customization of styling and interactions through CSS and JavaScript. In addition to displaying a user's top recommendations 2, each of our three modeling approaches - User-Based, Item-Based, and Neural - features its own distinct network visualization, further clarifying the mechanics of each model. Due to the robustness and interactivity of this approach, we believe our visualization to be on par with, or surpass existing state-of-the-art interfaces.

4.3.1 Visualization for User-based Collaborative Filtering

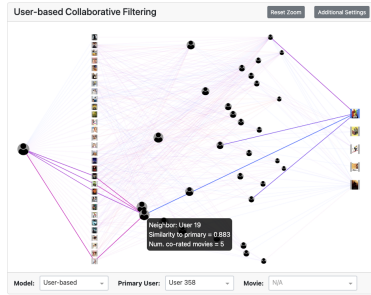


Figure 3: User-Based Collaborative Filtering Visualization

Figure 3 shows a screenshot of our user-based collaborative filtering model. The graph is organized horizontally, with the primary user on the far left and the final recommendations on the far right. Just to the right of the primary user is the *profile layer*, which contains the movies the user has already seen. These movies are sorted by decreasing rating, allowing the user to easily identify which of their past ratings have the most influence on the final recommendations.

The third layer shows the user's *neighborhood*—the other users most similar to the primary user based on their user profile. The size and position of neighbor nodes are meaningful: neighbors who are more similar appear larger and are positioned farther to the left, closer to the primary user. Additionally, the neighborhood layer is sorted vertically by the number of co-rated movies, with neighbors who have a greater overlap in watched movies placed higher up in the graph. Hovering over a neighbor reveals a tooltip with their similarity score and the number of co-rated movies.

Connections (edges) between the layers represent ratings. The edges from the primary user to the profile layer show the ratings the primary user gave to their movies. Edges from the neighbors to both the profile and recommendation layers represent the ratings those neighbors assigned. Each edge is colored by rating: deeper blue indicates a higher rating (up to 5 stars), while deeper red indicates a lower rating (closer to 0.5 stars). Hovering over an edge displays a tooltip with the exact rating value.

These connected edges across all four layers are the core of the visualization, enabling users to trace how their own

rated movies influence the similarity to their neighbors and ultimately lead to their final recommendations.

4.3.2 Visualization for Item-based Collaborative Filtering

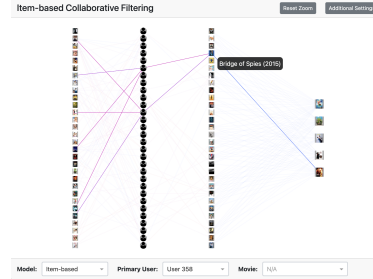


Figure 4: Item-Based Collaborative Filtering Visualization

Figure 4 displays a screenshot of the interactive visualization generated from the IBCF model for User 358. The structure proceeds from left to right, mirroring the logic of the recommendation process. The first column shows thirty movies that User 358 has rated followed by a layer of other users who have also rated at least one of those same movies. The third layer displays movies that are most similar to those in the user's profile, based on co-rating patterns aggregated across the entire user base. These "similar movies" are determined using cosine similarity between item vectors, as implemented in the IBCF algorithm. Each edge from the second to third layer represents a user-to-movie connection that reinforces item similarity, again maintain the same color gradient as the UBCF graph to reflect rating strength. On the far right, the personalized recommendations layer is predicted for User 358. These movies are not yet rated by the user but are inferred to be of high interest based on how similar they are to previously rated items. The set of edges connecting similar movies to the recommended ones reflects the weighted contribution of similarity scores and known ratings to each predicted value. These visual connections provide a transparent path from user history to recommendation, making it easy to trace how each suggestion was derived.

4.3.3 NeuMF Graph

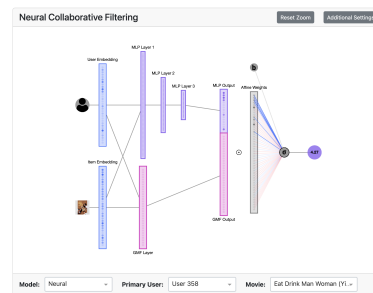
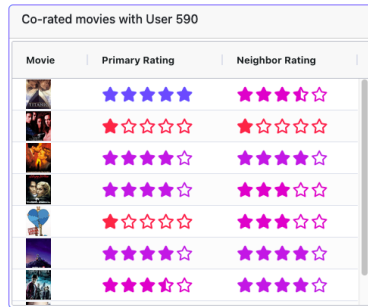


Figure 5: Neural Collaborative Filtering Visualization

Figure 5 displays a screenshot of our neural matrix factorization model. On the left, the graph shows 32-dimensional user and item embeddings, where the value of each element in the embedding is reflected through dot opacity and shown on hover. The diagram traces the input flow through two parallel paths: the GMF path (in pink, bottom) and the MLP path (in purple, top). The GMF applies a weighted element-wise multiplication to the user-item embeddings, and the MLP processes the embeddings through a series of hidden layers ($128 \rightarrow 64 \rightarrow 32 \rightarrow 16$) to capture more complex relationships between User 358 and the movie *Eat Drink Man Woman*. The outputs of both are concatenated to form a 48-dimensional final layer, where learned affine weights (whose edges are visualized as blue for positive, red for negative) and the sigmoid function produce a raw output between 0 and 1. This is de-normalized and thus produces the predicted score of 4.27, as shown above.

Ultimately, the visualization allows users to trace the latent features that generate the prediction by hovering over different components, allowing for transparency in what would otherwise be considered a "black box" recommendation model.

4.4 Interactivity



Co-rated movies with User 590		
Movie	Primary Rating	Neighbor Rating
	★★★★★	★★★★★
	★★★★★	★★★★★
	★★★★★	★★★★★
	★★★★★	★★★★★
	★★★★★	★★★★★
	★★★★★	★★★★★
	★★★★★	★★★★★
	★★★★★	★★★★★

Figure 6: Co-rated Grid

Our network graph is highly interactive. At a high level, users can switch between different model visualizations and select different primary users. Within each graph, clicking on nodes reveals additional helpful information. When a node is clicked, all of its connected edges are revealed and highlighted. By default, edges are obscured and non-interactive to maintain a clean view; clicking on the background returns the graph to this default state.

Clicking on a neighbor node provides the most detailed interaction. Not only are the neighbor's edges to the profile layer highlighted, but the corresponding edges from the primary user to those same movies are also highlighted, allowing users to directly compare ratings (seen in Figure 3). Additionally, a table (Figure 6) appears showing all co-rated movies along with the ratings given by both the primary user and the selected neighbor.

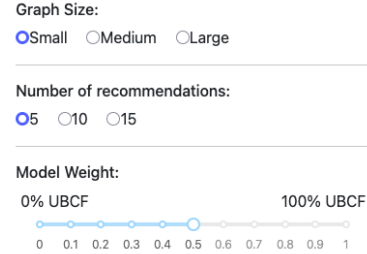


Figure 7: Additional Control Settings

An additional settings panel (Figure 7) provides users with manual control over key visualization parameters. Users can adjust the maximum number of nodes displayed in both the profile and neighbor layers, and independently control the number of recommendations visualized. A separate slider modifies the alpha parameter, which interpolates between User-Based Collaborative Filtering (UBCF) and Item-Based Collaborative Filtering (IBCF) models. This parameter only impacts the recommendation panel in the top-left corner and does not alter the underlying graph layout. Setting alpha to 0% fully prioritizes IBCF, aligning the displayed recommendations with those from the IBCF graph; setting it to 100% fully prioritizes UBCF, matching the recommendations shown in the UBCF graph.

Providing this level of interaction and control is critical for our system, as it allows users to explore model behavior and understand the influence of different signals.

5 Evaluation

We undergo robust evaluation process to test the accuracy, usability, and usefulness of our models and visualizations. Our aim is to understand the following: 1) how do traditional collaborative filtering approaches compare to deep learning approaches in terms of recommendation accuracy and runtime 2) does our interactive visualization improve user trust in recommendations, as measured through user experience surveys?

5.1 Experimental Design

5.1.1 Models

We evaluate the accuracy of our recommendation models using regression-based metrics appropriate for explicit feedback systems: root mean-squared error (RMSE) and mean absolute error (MAE). All three models used leave-one-out (LOO) evaluation, where one interaction per user is withheld for testing. While LOO is the standard evaluation approach for many recommendation systems, this method may introduce higher variance in error metrics compared to alternatives, potentially resulting in noisier performance assessments. The comparative results across models are presented in Table 1.

5.1.2 UX Research

To assess the usability of our tool, we conducted User Experience (UX) studies involving 10 participants. Each participant was tasked with interacting with the visualization, exploring the recommendations generated and interpreting how different factors (e.g. model type, user- vs. item-based weighting, etc.) influence the suggested content. Following the interaction, the participant completed a survey measuring the following key aspects: ease of interaction, indication of trust improvement, perception of novelty, and perceived recommendation accuracy. The survey uses a combination of 5-point Likert-scale questions and 2 open-ended feedback prompts.

5.2 Results

Table 1: Comparison of Recommendation Models

Metric	UBCF	IBCF	NCF
RMSE	0.7673	0.4962	0.9624
MAE	0.5856	0.3636	0.74
Runtime	5.24m	50.1m	0.8m

5.2.1 User-Based and Item-Based Systems

As shown in Table 1, the IBCF model achieves the best predictive accuracy, having the lowest RMSE (0.4962) and MAE (0.3636). The UBCF model performs slightly worse, but with a significantly lower runtime. These results are expected, since user preferences are more variable than movie characteristics, which can make item-similarity more consistent than user-similarity. Additionally, there were over 9000 distinct movies in our dataset but only roughly 600 distinct users, which explains the drastic runtime difference.

5.2.2 NeuMF System

Table 2: Diversity Metrics for NeMF

Metric	Value
Item Coverage	0.20%
Unique Item Percentage	5.26%
Item Concentration	19.61%

The NeuMF model demonstrates slightly worse accuracy compared to the UBCF/IBCF previous approaches, though still on par with other SOTA deep learning recommenders [15]. More concerning is the extreme popularity bias of the model, as reflected in the diversity metrics in Table 2. Despite the popularity regularization strategies discussed

in Section 4.2.3, the model shows alarming content diversity issues, with low item coverage (less than 0.1%) and a high item concentration (19.61%). For example, the movie *Tin Men (1987)* is recommended to over 90% of all users. We hypothesize that this result is due to adapting the model from implicit to explicit feedback, because the MSE-based objective function encourages the model to recommend more widely-rated content at the expense of long-tail items.

Nevertheless, NeuMF had the fastest runtime (< 1 minute), indicating that it is the most scalable model. Thus, future work should focus on addressing the content diversity problem to leverage this model’s computational efficiency.

5.2.3 Interactive Visualization

Upon completion of our UX studies, we found that users responded positively to the interactive visualization. On average, participants rated Ease of Interaction at 4.23/5 and Trust in Recommendation at 4.2/5, suggesting strong usability and credibility of our system. The consistency and narrow range of scores highlights navigating the interface was intuitive and the recommendations were reliable.

6 Conclusions and Discussion

We were able to develop an accurate and transparent recommendation model that allows users to explore the sources of their movie recommendations. The project’s success was demonstrated through a fully functional and interactive node-mapping visualization, clearly linking users to their recommended movies. Additionally, we evaluated the effectiveness of our model through ten UX studies, measuring how well the recommendations aligned with participants’ preferences and how they impacted users’ trust in recommendation systems.

As a potential future extension of the project, a clustering-based model could enhance transparency in users’ movie recommendations. Hierarchical clustering could first be used to group together similar movies, followed by feature reduction through Uniform Manifold Approximation and Projection (UMAP) to enable intuitive 2D visualization of the resulting clusters. For each user, previously rated movies could be visually highlighted, allowing users to explore the clusters to which those movies belong and enriching the exploratory recommendation experience.

The motivation behind this project was to address a key shortcoming in modern recommendation systems: lack of transparency. Through interactive visualizations and insights from user experience studies, we developed a movie recommendation system that directly addresses this issue. Our hope is that streaming services and other media platforms can adopt similar approaches to provide users with greater visibility into how recommendations are made — ultimately enhancing the entertainment experience for all.

All team members have contributed a similar amount of effort.

References

- [1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. Managing popularity bias in recommender systems with personalized re-ranking. *Proceedings of the 32nd International FLAIRS Conference*, 2019.
- [2] Rishabh Ahuja, Arun Solanki, and Anand Nayyar. Movie recommender system using k-means clustering and k-nearest neighbor. In *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pages 263–268. IEEE, 2019.
- [3] Svetlin Bostandjiev, John O’Donovan, and Tobias Höllerer. Tasteweights: a visual interactive hybrid recommender system. In *Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys ’12*, page 35–42, New York, NY, USA, 2012. Association for Computing Machinery.
- [4] Mohamed Amine Chatti, Mouadh Guesmi, and Arham Muslim. Visualization for recommendation explainability: a survey and new perspectives. *ACM Transactions on Interactive Intelligent Systems*, 14(3):1–40, 2024.
- [5] Songjie Gong. A collaborative filtering recommendation algorithm based on user clustering and item clustering. *J. Softw.*, 5(7):745–752, 2010.
- [6] Brynjar Gretarsson, John O’Donovan, Svetlin Bostandjiev, Christopher Hall, and Tobias Höllerer. Smallworlds: Visualizing social recommendations. *Computer Graphics Forum*, 29(3):833–842, 2010.
- [7] F. Maxwell Harper and Joseph A. Konstan. Movielens user rating dataset, 2015.
- [8] Chen He, Denis Parra, and Katrien Verbert. Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities. *Expert Systems with Applications*, 56:9–27, 2016.
- [9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. *Proceedings of the 26th International Conference on World Wide Web (WWW)*, pages 173–182, 2017.
- [10] Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work, CSCW ’00*, page 241–250, New York, NY, USA, 2000. Association for Computing Machinery.
- [11] Yifan Hu and Lei Shi. Visualizing large graphs. *WIREs Computational Statistics*, 7:115–136, 2015.
- [12] Hyung W. Kim, Keejun Han, Mun Y. Yi, Joonmyun Cho, and Jinwoo Hong. Moviemine: personalized movie content search by utilizing user comments. *IEEE Transactions on Consumer Electronics*, 58(4):1416–1424, 2012.
- [13] Hyeyoung Ko, Suyeon Lee, Yoonseo Park, and Anna Choi. A survey of recommendation systems: recommendation models, techniques, and application fields. *Electronics*, 11(1):141, 2022.
- [14] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 426–434, 2008.
- [15] Microsoft. Microsoft recommenders. GitHub Repository, 2023. Algorithms for building recommendation systems including implementations of several state-of-the-art algorithms for recommendation systems.
- [16] John O’Donovan, Barry Smyth, Brynjar Gretarsson, Svetlin Bostandjiev, and Tobias Höllerer. Peerchooser: visual interactive recommendation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI ’08*, page 1085–1088, New York, NY, USA, 2008. Association for Computing Machinery.
- [17] Elio Ventocilla and Maria Riveiro. A comparative user study of visualization techniques for cluster analysis of multidimensional data sets. *Information visualization*, 19(4):318–338, 2020.
- [18] Jing Yi and Zhenzhong Chen. Deconfounded cross-modal matching for content-based micro-video background music recommendation. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–25, 2024.