User Interest Driven Temporal Collaborative Filtering for Product Recommendation

Monoara Sultana Morzina
Department of CSE
Jahangirnagar University

ABSTRACT
Recommender systems, commonly employed in e-commerce, significantly aid clients in making informed decisions. The recommender system incorporates a wide variety of algorithms but collaborative filtering algorithm is regarded as among the most efficient technologies used in customized recommendation system. However, conventional algorithms merely take user ratings into account, failing to take into account shifts in user interest or the veracity of rating data, which had a significant impact on the system's suggestion quality. An improved approach is suggested in this study to deal with this problem. Initially, a gradual time drop is employed to give the user's rating a weight. Then, the current user's neighbors are chosen from a group of users who share common interests. Finally, the average assessments of his neighbors may indicate the current user's choice for a certain item. According to experimental data, the algorithm has the potential to upgrade the recommendation system's quality and prolong the accuracy of neighbor detection.

Keywords
Recommender System; Collaborative Filtering; Personalized Recommendations; Time-Weighted Ratings; Neighbor Selection.

1. INTRODUCTION
Recommendation systems (RS) are essential for helping consumers make decisions about things like what to buy, what to watch, learning about new music, or reading the news online [1], [2]. Consumers can get their preferred items from recommendation system While they are most commonly associated with e-commerce platforms, their applications extend far beyond shopping [3]. RS leverage user input to curate lists of favored items, employing a variety of algorithms to generate personalized recommendations. These algorithms analyze patterns in user behavior and preferences to enhance the relevance of the suggestions provided. Recommendation algorithms can be broadly classified into four categories: hybrid filtering, content-based filtering, collaborative filtering (CF), and demographic filtering [4]. The user-based nearest neighbor algorithm is the most common recommendation system among them. CF has evolved as one of the most efficient technologies for personalized recommendation systems. Google News [5], [6]. CF is founded on the prime assumption that customers with uniform preferences will evaluate products similarly. The program determines neighbors (people with similar interests) by comparing individuals based on their ratings. By adding the ratings of their neighbors for the same item, the algorithm then forecasts the choice of the active user for that item. Finally, the algorithm recommends the top items that the active user is probably going to appreciate [7].

Nonetheless, there are certain crucial research problems in order to overcome the three main obstacles. A significant drawback of traditional collaborative filtering stems from the way it computes similarity between users or items. Instead of taking into account how user interests change over time, it usually only looks at the similarity score between users [8], [9]. As a result, the same ratings given at different times are treated equally, which can cause the recommendations to become misaligned with the user's current preferences. This lack of temporal consideration means that users may receive outdated or irrelevant recommendations based on past behaviors [10]. In addition to this, the reliability of user ratings plays a crucial role in similarity calculation. Unreliable or erroneous ratings can distort the similarity measure, leading to inaccurate identification of neighbors and ultimately reducing the quality of the recommendations [11]. The second concern is to augment the scalability of CF models. While the expectation of contemporary systems require the search of an enormous number of millions of potential neighbors, these algorithms can explore a significant volume of them in real-time [12]. Furthermore, the site has a lot of information about individual users, which causes performance issues for the current algorithms. Enhancing the caliber of user recommendations is the other difficulty. Customers need products they will appreciate and recommendations they can rely on to assist them. If recommender systems do not continuously provide accurate results, users will respond by abandoning them. These issues are somewhat contradictory because algorithms that spend less time looking for neighbors will be more scalable and of worse quality. To ensure that the solutions found are both practical and helpful, it is crucial to address the two issues at the same time [13].

In our work, we address the challenges of recommender systems using a novel way, user-based collaborative filtering. User-based algorithms may be able to provide the same quality as the item-based algorithms with less online computation. We proposed a new collaborative filtering algorithm that incorporates time-weighting and credit evaluation. By placing greater emphasis on recent ratings, this method accounts for the changing nature of user interests and reflects shifts in user preferences over time. Additionally, it evaluates user evaluations' credibility to increase the similarity measure's dependability. By doing this, this approach improves the recommendation system's performance by increasing the process' accuracy and better meeting the user's present demands. The following contributions are included in this research project:

- The primary emphasis of this research is the usage of changing user preferences.
- Provide a time-weight function that increases recommender systems' accuracy in making predictions.
- Make use of various item content and suggestion settings to address the future anticipation.

2. RELATED WORK

This section provides a concise overview of prior research relevant to collaborative filtering (CF), a common strategy in customized systems for recommending items. Despite its popularity, one important restriction in practical implementations is the relatively sparse nature of user-item interaction data, which has substantially less rated things than total available items.

Zhuo *et al.* [14] tackled this limitation by integrating case-based reasoning (CBR) with CF, thereby enhancing the performance of recommendation systems in online retail environments. Similarly, Jiang *et al.* [15] introduced a Personalized Hybrid Collaborative Filtering (PHCF) approach that unites both consumer and product oriented techniques, yielding notable improvements in recommendation accuracy. Wang *et al.* [16] focused on overcoming scalability and accuracy challenges in CF by incorporating item combination features along with user demographic data. Their method also employs a genetic algorithm to optimize feature weights during user similarity calculations. To address the sparsity problem, Huang *et al.* [17] developed the Association Clusters Filtering (ACF) algorithm, which demonstrated superior performance in cold-start situations while maintaining accuracy and efficiency for non-cold-start scenarios.

Ghazanfar et al. [18] introduced a cascading hybrid recommendation model that leverages item ratings, features, and demographic information. This multi-layered approach significantly outperformed conventional systems by addressing common issues such as data sparsity, scalability, and cold-start problems. He et al. [19] proposed a Time-Context-Based Collaborative Filtering (TBCF) algorithm that integrates time intervals within CF process, improving both prediction accuracy and recall by accounting for temporal dynamics in user behavior. Yun et al. [20] developed a hybrid filtering approach aimed at distributed commercial mirror sites. To handle sparse data in CF, they introduced a webpage rating prediction method that enhances similarity calculations and boosts recommendation precision. Almosallam et al. [21] tackled the problem of sparsity in memory-based CF by employing z-scores instead of explicit ratings and proposed an adaptive model that balances global and item-specific statistics based on data density.

Leung et al. [22] introduced a hybrid recommendation strategy using Cross-Level Association Rule Mining (CLARE), which generates effective recommendations even with insufficient rating data by leveraging item attribute information. Rodrigues et al. [23] developed a framework that merges item-specific CF with socioeconomic data through a cluster-driven weighting method. This approach delivers reliable recommendations to new users, enhancing user satisfaction and potentially increasing business revenue. Ungar et al. [24] formulated a statistical model for CF and evaluated different parameter estimation methods, including K-means variations and Gibbs Sampling. Their model is adaptable for multi-attribute object clustering. O'Connor et al. [25] explored item partitioning using data clustering to improve scalability in CF, although its effect on accuracy was mixed.

Ji *et al.* [26] presented an adaptable matrix utilizing the factorization CF algorithm that incorporates dual selection matrices: consumer-cluster matrix and consumer- index term matrix—rather than an individual user-item matrix. Their results demonstrated high scalability, especially for recent items. Honda *et al.* [27] presented a CF method based on local principal components, integrating fuzzy clustering and PCA to handle datasets with missing values. Koohi *et al.* [28] proposed a neighbor-finding strategy using subspace clustering. They categorized rated data into Inquisitive, Neither inquisitive Nor Uninquisitive, Uninquisitive segments, constructing a hierarchical tree structure to better manage data sparsity. Verma *et al.* [29] devised a personalization platform that integrates CF with the Fuzzy C-Means (FCM) technique for clustering. Unlike K-means, that restricts an item to a single cluster, FCM allows an item to belong to multiple groups, leading to better clustering results and enhanced rating predictions.

A growing body of research focuses on integrating temporal dynamics and user interest modeling to improve recommendation accuracy. Adaptive frameworks that reflect changes in user behavior over time have shown promise in addressing cold-start and sparsity issues. These advancements not only refine the personalization process but also enhance user engagement and commercial effectiveness in e-commerce environments

3. MATERIAL & METHODOLOGY

3.1 Materials

For our explorations, we used the freely accessible dataset from GroupLens (http://www.grouplens.org/) [30], which is widely used for evaluating collaborative filtering algorithms. This dataset, originally collected from the MovieLens website, contains extensive user rating information and has been packaged specifically for research validation purposes. Many collaborative filtering models have adopted this set of data to test the correctness and productivity of their methods. The specific data subset used in our study includes 943 users who have generated 100,000 ratings across 1,682 movies. The rating matrix has a sparsity of around 0.9370, computed as 100,000 / (943 × 1,682), suggesting a fairly sparse dataset. The dataset comprises several files: user.data (user rating data), user.info (user information), user.item (movie details) and user.genre, each contributing to different aspects of the recommendation model.

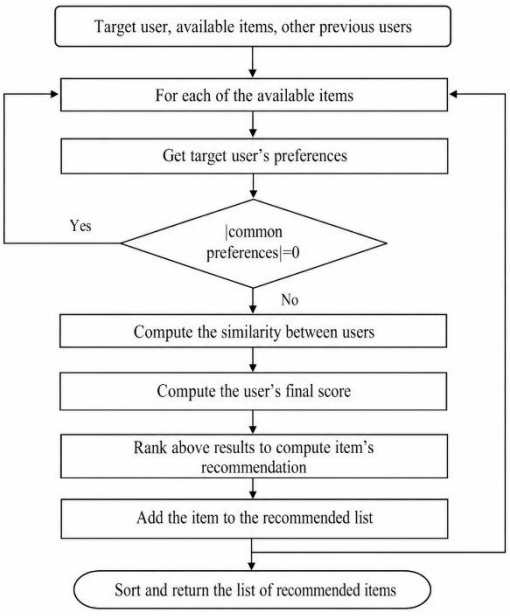

Figure 1: Flowchart of proposed user- based recommendation system

The dataset records are structured in rows, with fields separated by vertical bars (|). Each entry in the user.data file includes the individual's ID, movie ID, assessment score, and a temporal marker. The temporal marker indicates the number of seconds passed since January 1, 1970 (Unix epoch time). The user.info file contains a single line summarizing the overall number of users, items, and scores in the dataset. The user.item file provides detailed report about each movie. Each record includes the movie ID, name, availability date, video release date, and IMDb URL, followed by 19 binary indicators representing various movie genres such as educational, historical, biographical, musical, mystery, animation, romance, suspense, science fiction, horror, drama and comedy. A number of 1 points out that the movie falls within that particular genre, whereas 0 points out that it does not. The user.genre file lists the available genre categories and their corresponding indices. The user.info file describes user attributes. Each line contains the user ID, age, gender, occupation, and postal code, with each field also separated by vertical bars. These attributes can be utilized for demographic-based filtering and analysis.

3.2 Methodology

The suggested approach outlines the process of generating personalized recommendations using a collaborative filtering-based approach. The entire methodological steps are shown in Figure 1. The methodology begins by taking the intended user, available items, and data from previous users as inputs. For each available item, the system retrieves the intended user's choices and looks for those that are common with other users. If no common preferences exist, the process skips to the next item. Otherwise, the affinity among the intended user along with other users is generated to locate adjacent users, and a final score for the intended user's desire for the item is calculated. These scores are ranked to prioritize items, and then included to the suggestion list. At last, the suggested items get organized and presented as a personalized list for the intended user. The more required descriptions regarding the methodology are given below.

3.2.1 Algorithm Description

The recommendation algorithm presented in this paper operates in three main stages. First, user interaction data, particularly the ratings given by users, is gathered and then saved in a database. Next, the k-nearest neighbors (KNN) method is utilized to forecast scores for items that the intended user has not rated yet [31]. A key challenge in KNN lies in accurately measuring the affinity between the intended user and prospective neighbors. To address this, an enhanced similarity model is introduced to reduce calculation bias and enhance recommendation precision, as described in the following sections. Finally, the top N items with the highest predicted ratings are selected and recommended to the target user.

In user-based collaborative filtering systems, consumer rating data is generally depicted as a consumer-item grading matrix R_{m*n}, wherein m signifies the overall amount of consumers and n denotes the overall amount of products. Each element $R_{i,j}$, in the matrix represents the scoring assigned by consumer i to product j, illustrating the consumer's level of choice for that specific item.

3.2.2 Similarity Measures

Recommender platforms use a range of similarity metrics, many of those are rooted in machine learning and vector space models. Although all of these metrics aim to quantify similarity, they differ in how similarity is defined and computed These strategies are broadly classified into distance-driven and correlation-driven procedures. Various techniques exist for measuring similarity between users, each with its own mathematical formulation, leading to potentially different similarity outcomes. The following subheadings [18] provide an overview of certain of the more often used affinity estimation methods.

In collaborative filtering systems, a basic component is the assessment of likeness among people or things. The Apache Mahout library offers a collection of well-established similarity algorithms, enabling developers to seamlessly integrate them into recommender systems for identifying alike user neighborhoods or computing item affinity. Mahout provides various commonly used similarity metrics, including the Manhattan distance, Jaccard similarity, Euclidean Distance Pearson Correlation Coefficient, Cosine Similarity, and Chebyshev distance. Among them, multiple research [18, 19, 20] have demonstrated that the Pearson Correlation Coefficient (PCC) provides more accurate affinity scoring between consumers or products than distinct approaches. Therefore, this paper employs the PCC to represent the foremost affinity measure, that is formally stated as follows:

Pearson Correlation Similarity (PCC): When assessing likeness in collaborative filtering systems, the Pearson correlation coefficient frequently serves as the standard method. It quantifies a linear relationship by comparing how values in one variable track those in another. Below is the equation that measures the affinity of two items, *u* and *v*, based on their corresponding ratings [32].

$$\text{Sim} = \frac{\sum \text{i} \in I_{uv} (R_{u,i} - \overline{R_u}) \ (R_{v,i} - \overline{R_v})}{\sqrt{\text{i} \in I_{uv} (R_{u,i} - \overline{R_u})^2} \ I \ \sqrt{\text{i} \in I_{uv} (R_{v,i} - \overline{R_v})^2}} \quad (1)$$

In this context, $sim(u, v)$ denotes the affinity between user u and user v. $I_{uv} (I_{uv} = \text{I(u)} \cap \text{I(v)})$ represents the collection of items that have been rated by both users. $R_{u,i}$ and $R_{v,i}$ indicate the ratings given by users u and v to item i, respectively, $\overline{R_u}$ *and* $\overline{R_v}$ refer to the arithmetic mean of the ratings provided by users u and v, respectively, across all items they have rated. This coefficient measures the strength of the linear relationship between two paired numerical series [25]. A strong positive correlation between users results in a value close to 1, indicating similar rating patterns. A coefficient around 0 reflects minimal linear correspondence; one near –1 denotes ratings that are inversely related.

Formation of Neighbors: Collaborative Filtering (CF) methods apply statistical techniques to evaluate the similarity between users, thereby identifying a group of users referred to as neighbors. Similarity measures serve as metrics that quantify the degree of relevance between two vectors [22]. In user-based collaborative filtering (UBCF), user similarity is computed by comparing their respective rating vectors. Once these similarities are determined, they are used to construct a neighborhood around the target user for generating recommendations.

Prediction Computation of UB-CF: After locating the set of similar users (neighbor set) in UBCF, the model computes a predicted rating for item i by aggregating the neighbors' ratings—often using a weighted average based on similarity between each neighbor and user u. The predicted evaluation is calculated as a similarity-weighted mean of the neighbors' ratings for item *i*, normalized by the sum of the similarity weights. Let N_t denote the neighborhood for user t [33]. The predicted rating for item i is then calculated as:

$$P_{user-based}(t,i) = \overline{U_t} + \frac{\sum j \in N_t \, (R_{j,i} - \overline{R_j}) \; sim(t,j)}{\sum j \; N_t | \; sim(t,j)|} \quad (2)$$

In this formula, $\overline{U_t}$ represents the mean rating of user t across all items they have assessed. $R_{j,i}$ is the evaluation of item i by neighboring userj, whereas, $\overline{R_j}$ stands for the mean scoring of user j. The term $sim(t,j)$ represents the affinity of intended user t and their neighbor j. These components together help determine how much influence each neighbor's rating should have in estimating the intended user's score for item i.

3.2.3 CF methodology customized for user interest

Traditional user-based collaborative filtering (UBCF) algorithms assess a user's potential preference for an item by examining the feedback of users with comparable interests. Therefore, identifying appropriate neighbor users is a critical step. However, user interests are not static—they evolve over time. As a result, a user may rate the same item differently at different points in time. Traditional approaches typically treat all past ratings equally when searching for similar users, which can lead to inaccurate neighbor selection and suboptimal recommendation quality. In reality, more recent ratings often reflect a user's current preferences more accurately, while older ratings may have less relevance to their present interests [34]. Thus, it is both necessary and reasonable to evaluate rating similarities by considering the time context—comparing the target user's ratings with others made at the same or a relatively close time. This temporal awareness helps in identifying more relevant neighbors. Table 1 demonstrates how user interests might shift over time.

Table 1: Example of user interest changing

	I_1	I_2	I_3	I_4	I_5	Rating Interval
U_1	4	5	-	4	3	5
U_2	3	5	4	4	2	20
U_3	4	4	3	2	3	5
U_4	3	4	4	4	3	10

The table shows, four users' rating for five items are presented along with the corresponding evaluation time intervals, where a larger time value indicates an earlier rating. When similarity is calculated solely based on rating values, the traditional method identifies the potential neighbor ranking as U4<U3<U2. However, by incorporating the time factor, our proposed approach appropriately reduces the influence of user U2's ratings on user U1's preferences, as their evaluations are temporally distant. As a result, the adjusted neighbor ranking becomes U3>U4>U2, Which more truly represents the target user's current interests.

To enhance the influence of recent ratings during recommendation generation, we introduce a time-weighting mechanism based on users' rating timestamps. Drawing inspiration from the forgetting curve in cognitive psychology, we observe that memory retention declines rapidly shortly after learning, and the rate of forgetting gradually slows over time. This represents a nonlinear decay process—from fast to slow [21, 22]. Based on this principle, we adopt a gradual forgetting strategy in this study. By assigning lower weights to older ratings and emphasizing more recent evaluations, the model better reflects the user's current preferences. Denote t_0 as the reference start time and t_r as the time at which the rating took place. The time interval t is thus defined to be the difference between the observed rating time and the reference start time: t = $(t_r - t_0)$. Let t_{Min} as the minimum value of $(t_r - t_0)$ and t_{Max} as its maximum, spanning all rating instances. Using the preceding definitions, the time-driven weighing factor is formulated below:

$$H(t) = m \times (\frac{t - t_{Min}}{t_{Max} - t_{Min}})^2 + 1 - \mathrm{m} \quad (3)$$

In this function, the parameter m controls the rate at which the influence of past ratings decays over time—it reflects the "forgetting ability" of the model. A larger value of m results in a faster decline in the weight assigned to older ratings, thereby emphasizing more recent user behavior. The selection of m depends on how quickly user interests are expected to change within the recommendation system: systems with rapidly shifting user preferences require a higher m, while systems with more stable interests can use a smaller value.

3.2.4 Description of Improved CF algorithm

The Pearson Correlation Coefficient (PCC) approach identifies a target user's neighbors based on evaluated items. However, this technique might result in incorrect similarity evaluations in circumstances where the amount of correlated items is quite restricted. Even when the few shared ratings are numerically close, the resulting similarity score may be misleadingly high. In reality, users with very few commonly rated items may not share similar interests. As a result, it is critical to evaluate both the degree of rating similarity and the number of correlated items. We introduce a weighting factor into the traditional similarity formula to mitigate this issue and enhance the robustness of user-similarity measurements. This adjustment ensures that similarity calculations more accurately reflect true user interest alignment. The modified similarity function is defined as follows:

$$S(u) = \frac{Min(k,\gamma)}{\gamma} \quad (4)$$

The formula includes k, the amount of items correlated to users u and v, and γ, a predetermined threshold to alter the similarity score. This threshold is usually chosen according to the degree of sparsity in the individual–item rating matrix. In highly sparse datasets, where users share fewer commonly rated items, a smaller value of γ is recommended. According to the formula, if k exceeds the threshold γ, no adjustment to the similarity score is necessary. However, if k falls below γ, the similarity must be modified as specified in Equation (5). Finally, the three weighting factors introduced—time, credit, and co-rated item count—are combined to calculate the overall similarity score between users. This combined similarity may be characterized as follows:

$$W = H(t) \times S(u) \quad (5)$$

Next, we apply the adjusted similarity weight to modify the classic Pearson Correlation Coefficient (PCC) approach stated in Formula (1). The improved similarity measure is expressed as follows:

$$Sim^*(u,v) = \frac{\sum i \in I_{uv} \, (R_{u,i} - \overline{R_u}) \; (R_{v,i} - \overline{R_v})}{\sqrt{i \in I_{uv} (R_{u,i} - \overline{R_u})^2} \; I \; \sqrt{i \in I_{uv} (R_{v,i} - \overline{R_v})^2}} \quad (6)$$

By incorporating the enhanced similarity measure into the traditional collaborative filtering (CF) framework, we propose

a novel recommendation algorithm that accounts for dynamic user interest. The primary steps of the algorithm are outlined below:

3.2.5 CF Technique considering evolving user preferences and trust metrics

Input: We use ***T*** to represent the intended user, ***R*** for the consumer–product scoring matrix, and ***k*** for the given number of nearest neighbors.

Step1. By excluding Um from the rating matrix *R*, the model avoids recommending items already assessed by the intended user *T*, which supports a fair evaluation of candidate recommendations.

Setp2. Using Equation (6), compute similarity between the intended user *T* and each other user $u \in Um$, (where $u \neq T$). Rank users by their affinity scores and choose the top-k users; denote this set of nearest neighbors of *T* as $\{j1, j2, \ldots jk\}$.

Step3. Calculate the score $P_{T,i}$ for item i$\in I_c$, which belongs to the candidate set $(I_c = I_n - I_t)$ — as the estimated preference of the current user *T*.

$$P_{T,i} = \overline{U_t} + \frac{\sum j \in N_t \, (R_{j,i} - \overline{R_j}) \; sim(t,j)}{\sum j \; N_t | \; sim(t,j)|} \quad (7)$$

Step4. Generate a recommendation list for the intended user *T* by choosing the *N* items with the greatest predicted ratings.

4. EXPERIMENTAL SETUP

The system configuration for the experiment comprised a laptop with an Intel Core i5-2450M processor (2 cores, 4 threads, 2.5 GHz), 4 GB of DDR3 RAM, and running the Windows 10 operating system. The Python programming language (version 3.6) was used to complete the implementation within the Anaconda3 development environment.

5. EVALUATION METRICS

In the evaluation of recommendation algorithms, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are among the most prevalently used metrics for assessing predictive performance. The Mean Absolute Error (MAE), defined as the average of the absolute differences between predicted and actual values, was utilized as the primary evaluation metric in this study.

Mean Absolute Error: As a statistical measure, this metric evaluates the performance of a recommendation algorithm by assessing the difference between estimated and observed user ratings for each user-item pair in the evaluation dataset. For each prediction, it may be appropriate to compute the magnitude of discrepancy between its forecasted and observed rating to quantify performance. These absolute errors are then summed and divided by the total number of predictions to obtain the Mean Absolute Error (MAE) [35]. MAE is one of the most widely used evaluation metrics due to its simplicity and ease of interpretation. The formula for calculating MAE is as follows:

$$\text{MAE} = \frac{\sum_{i=1}^{N} |pi - qi|}{N} \quad (8)$$

Here, *N* denotes the number of items suggested to the current user. Lower MAE values signify superior predictive accuracy, indicating that the system more closely reflects the user's authentic preferences.

6. RESULT AND DISCUSSION

We undertake three distinct experimental series in this section to determine how effectively the proposed CF algorithm operates. Initially, we examine how the parameter γ affects the prediction performance within $S(u)$. Since data sparsity varies across different e-commerce platforms, dynamically tuning the γ value allows for the optimization of recommendation outcomes. The subsequent figures illustrate how the recommendation performance changes with different γ values.

As shown in Figure 2, setting γ too low—across various neighbor counts (ranging from 5 to 30)—can cause the computed result to default to 1 based on formula (4). Such behavior results from the number of mutually co-rated items between the two users meeting the lower threshold limit. Conversely, assigning a high value to γ may reduce the weight excessively, hindering the accurate identification of the active user's neighbors. Thus, selecting an appropriate threshold is essential. For this particular dataset, the optimal recommendation quality is achieved when γ is set to 15.

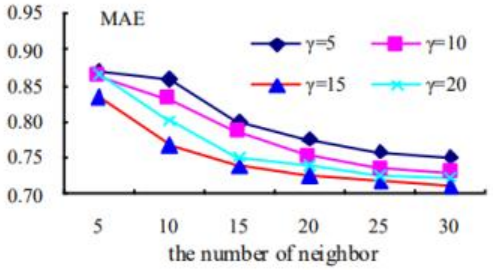

Figure 2: The performance under different γ's value

Next, we analyze how the parameter m influences prediction performance within the time-weighting function $H(t)$, which is designed to capture the rate at which user interest fades over time. Figure 3 presents the MAE values corresponding to different m values, ranging from 0 to 0.8.

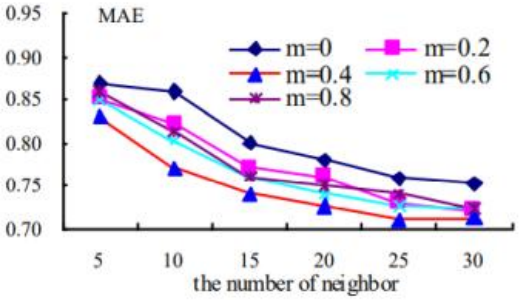

Figure 3: Performance across different values of forgetting velocity m

Setting m to 0 treats all user ratings with equal importance, regardless of when they were made. This strategy is consistent with traditional methodologies but fails to accommodate the temporal evolution of user preferences. However, assigning m a value between 0.2 and 0.8 enables the algorithm to gradually lessen the influence of previous ratings, thus reflecting shifting user interests. In our experiments, the best recommendation performance was achieved when m was set to 0.4, indicating that this value most effectively captures users' current preferences and enhances satisfaction.

Finally, Figure 4 presents a comparative evaluation —highlighting the differences in recommendation performance between the proposed —Interest Change and Credit Evaluation-Based Collaborative Filtering (ICCEBCF) and the traditional User-Based Collaborative Filtering (UBCF) method. This comparison is conducted under the conditions where γ is set to 15 and m is set to 0.4.

This comparison is conducted under the conditions where γ is set to 15 and m is set to 0.4.

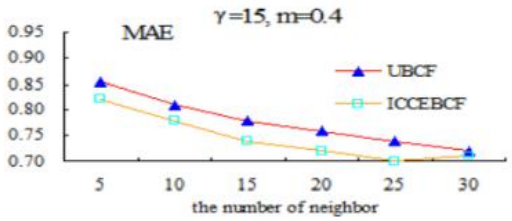

Figure 4: Comparison of recommended performance across two methods

The incorporation of the time-based weighting factor and user credit assessment function leads to a noticeable improvement in performance, as reflected by the lower MAE values across different numbers of neighbors compared to traditional methods. This integrated approach effectively captures users' recent preferences while also assessing the reliability of their ratings. As a result, the generated recommendations are both more accurate and better aligned with users' current interests

7. CONCLUSION

In this research, we provide an efficient, interest-dynamic collaborative filtering technique, designed to improve the recommendation system's accuracy. This approach may allow for more accurate identification of neighbors whose interests align with the intended user's, by including the temporal weight based on the non-linear forgetting function during the user similarity computation. The final results, based on neighbor-averaged ratings, appear to more effectively represent the shifting interests of the user over time. Experimental findings appear to indicate that our strategy can successfully increase the system's performance when compared to the conventional methodology. The major scalability issues brought on by the quickly expanding user base and item count will be addressed in future development.

The suggested approach might be expanded by incorporating contextual data, item or user attribute information, etc. This would be accomplished by altering the conditioned probability calculations. Future research is suggested in the following areas: applying additional metrics (novelty, diversity, serendipity, etc.) to evaluate the suggested approach; expanding the suggested method to user groups; investigating algorithm parallelization to increase scalability; and expanding the suggested method by incorporating social network data, item attributes, and other contextual data.

8. REFERENCES

[1] Sarwar, B. M., Karypis, G., Konstan, J., & Riedl, J. (2002, December). Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In Proceedings of the fifth international conference on computer and information technology (Vol. 1, pp. 291-324).

[2] Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to Recommender Systems Handbook. Springer.

[3] Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. Advances in artificial intelligence, 2009(1), 421425.

[4] Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2010). Recommender systems: an introduction. Cambridge university press.

[5] Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. IEEE Internet computing, 7(1), 76-80.

[6] Das, A. S., Datar, M., Garg, A., & Rajaram, S. (2007, May). Google news personalization: scalable online collaborative filtering. In Proceedings of the 16th international conference on World Wide Web (pp. 271-280).

[7] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994, October). Grouplens: An open architecture for collaborative filtering of netnews. In Proceedings of the 1994 ACM conference on Computer supported cooperative work (pp. 175-186).

[8] Koren, Y. (2009, June). Collaborative filtering with temporal dynamics. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 447-456).

[9] Jiang, W., Chen, J., Jiang, Y., Xu, Y., Wang, Y., Tan, L., & Liang, G. (2019). A New Time-Aware Collaborative Filtering Intelligent Recommendation System. Computers, Materials & Continua, 61(2).

[10] Liu, Q., Chen, E., Xiong, H., Ding, C. H., & Chen, J. (2011). Enhancing collaborative filtering by user interest expansion via personalized ranking. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 42(1), 218-233.

[11] O'Donovan, J., & Smyth, B. (2005, January). Trust in recommender systems. In Proceedings of the 10th international conference on Intelligent user interfaces (pp. 167-174).

[12] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001, April). Item-based collaborative filtering recommendation algorithms. In Proceeding of the 10th international conference on World Wide Web (pp. 285-295).

[13] McSherry, D. (2005). Explanation in recommender systems. Artificial Intelligence Review, 24, 179-197.

[14] Zhuo, G., Sun, J., & Yu, X. (2011, July). A framework for multi-type recommendations. In 2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD) (Vol. 3, pp. 1884-1887). IEEE.

[15] Jiang, Y., Liu, J., Tang, M., & Liu, X. (2011, July). An effective web service recommendation method based on personalized collaborative filtering. In 2011 IEEE international conference on web services (pp. 211-218). IEEE.

[16] Wang, Q., Yuan, X., & Sun, M. (2010, August). Collaborative filtering recommendation algorithm based on hybrid user model. In 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery (Vol. 4, pp. 1985-1990). IEEE.

[17] Huang, C., & Yin, J. (2010, August). Effective association clusters filtering to cold-start recommendations. In 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery (Vol. 5, pp. 2461-2464). IEEE.

[18] Ghazanfar, M. A., & Prugel-Bennett, A. (2010, January). A scalable, accurate hybrid recommender system. In 2010 Third International Conference on Knowledge Discovery and Data Mining (pp. 94-98). IEEE.

[19] He, L., & Wu, F. (2009, August). A time-context-based collaborative filtering algorithm. In 2009 IEEE International Conference on Granular Computing (pp. 209-213). IEEE.

[20] Yun, L., Xun, W., & Huamao, G. (2008, November). A hybrid information filtering algorithm based on distributed web log mining. In 2008 Third International Conference on Convergence and Hybrid Information Technology (Vol. 1, pp. 1086-1091). IEEE.

[21] Almosallam, I. A., & Shang, Y. (2008, June). A new adaptive framework for collaborative filtering prediction. In 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence) (pp. 2725-2733). IEEE.

[22] Leung, C. W. K., Chan, S. C. F., & Chung, F. L. (2007, November). Applying cross-level association rule mining to cold-start recommendations. In 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Workshops (pp. 133-136). IEEE.

[23] Rodrigues, Celine Michael, Sheetal Rathi, and Ganesh Patil. "An efficient system using item & user-based CF techniques to improve recommendation." 2016 2nd International Conference on Next Generation Computing Technologies (NGCT). IEEE, 2016.

[24] Ungar, L. H., & Foster, D. P. (1998, July). Clustering methods for collaborative filtering. In AAAI workshop on recommendation systems (Vol. 1, pp. 114-129).

[25] O'Connor, M., & Herlocker, J. (1999, August). Clustering items for collaborative filtering. In Proceedings of the ACM SIGIR workshop on recommender systems (Vol. 128). UC Berkeley.

[26] Ji, K., & Shen, H. (2014, July). Using category and keyword for personalized recommendation: A scalable collaborative filtering algorithm. In 2014 Sixth International Symposium on Parallel Architectures, Algorithms and Programming (pp. 197-202). IEEE.

[27] Honda, K., Sugiura, N., Ichihashi, H., & Araki, S. (2001, October). Collaborative filtering using principal component analysis and fuzzy clustering. In Asia-Pacific Conference on Web Intelligence (pp. 394-402). Berlin, Heidelberg: Springer Berlin Heidelberg.

[28] Koohi, H., & Kiani, K. (2017). A new method to find neighbor users that improves the performance of collaborative filtering. Expert Systems with Applications, 83, 30-39.

[29] Verma, S. K., Mittal, N., & Agarwal, B. (2013, September). Hybrid recommender system based on fuzzy clustering and collaborative filtering. In 2013 4th international conference on computer and communication technology (ICCCT) (pp. 116-120) IEEE.

[30] Movielens (2024,July 29).GroupLens.http://grouplens. Org/datasets/movielens/ [Accessed on: 21 January, 2025]

[31] Bahrani, P., Minaei-Bidgoli, B., Parvin, H., Mirzarezaee, M., & Keshavarz, A. (2024). A new improved KNN-based recommender system. The Journal of Supercomputing, 80(1), 800-834.

[32] Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. Advances in artificial intelligence, 2009(1), 421425.

[33] Matuszyk, P., & Spiliopoulou, M. (2014, June). Predicting the performance of collaborative filtering algorithms. In Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14) (pp. 1-6).

[34] Zhang, L., Tao, Q., & Teng, P. (2014). An Improved Collaborative Filtering Algorithm Based on User Interest. J. Softw., 9(4), 999-1006.

[35] Zhang, F., Gong, T., Lee, V. E., Zhao, G., Rong, C., & Qu, G. (2016). Fast algorithms to evaluate collaborative filtering recommender systems. Knowledge-Based Systems, 96, 96-103.