*Artificial Intelligence Technologies and Applications*
*C. Chen (Ed.)*

*doi:10.3233/FAIA231341*

# Research on the Evaluation of Algorithms of Recommendation Systems of We-Media and Possible Future Development

Minghao YOU[1]
*Wuhan Britain-China School, Wuhan, 430000, China*

**Abstract.** With the exponential growth of we-media platforms, the recommendation of personalized content has become crucial for enhancing user satisfaction and engagement. However, the inherent dynamics of we-media data and the evolving nature of user preferences pose significant challenges for recommendation algorithms. This paper delves into these challenges and manages to solve them by examining past recommendation systems, such as collaborative filtering, content-based, and hybrid systems, by evaluating them with the k-means clustering model and MAP algorithm based on SQL. By examining the results, the trend in improving accuracy and stability is verified, and the characteristics of each algorithm are concluded, which may bring concerns to critics and shed light on the future development of the field.

**Keywords.** We-media, recommendation system, MAP, SQL, k-means clustering.

## 1. Introduction

This research paper explores the algorithm and model of personalized recommendation for we-media platforms and investigates the potential for optimization. With the increasing popularity of we-media platforms, personalized recommendation systems have become crucial for enhancing user experience and content engagement [1]. By examining current algorithms and models and the experimental results gained, this study aims to identify potential areas for improvement and propose optimization strategies.

### *1.1 Research Background*

We-media platforms, also known as social media or user-generated content platforms, have witnessed exponential growth over the past decade. These platforms enable users to create and share content, fostering a participatory culture where individuals can express themselves, connect with others, and engage in online communities. We-media platforms encompass a wide range of platforms, including social networking sites, video-sharing platforms, and content curation platforms.

One of the key drivers behind the success of we-media platforms is the personalized recommendation system. These recommendation systems play a crucial role in enhancing user experiences by delivering relevant and engaging content to individual

[1] Corresponding author: Minghao YOU, Wuhan Britain-China School, Wuhan, 430000, China; Email: steve.you1@outlook.com

users. Personalized recommendations help users discover content tailored to their interests, preferences, and behaviors, thus increasing content engagement and platform stickiness [2].

The significance of personalized recommendation systems in we-media platforms can be attributed to several factors. First, traditional content discovery methods, such as search engines or chronological feeds, may not adequately address the diverse and dynamic nature of content on these platforms [3]. Personalized recommendations augment these methods by bringing attention to content that users may not have actively searched for but are likely to find interesting.

Second, we-media platforms are flooded with an overwhelming amount of content, making it challenging for users to navigate and consume relevant information [4]. Personalized recommendations alleviate information overload by curating and presenting a subset of content tailored to each user's preferences. This not only saves users time and effort but also enhances their overall satisfaction with the platform.

Third, personalized recommendations also benefit content creators and platform owners. By promoting relevant content to interested users, creators can gain visibility, reach a wider audience, and increase their content's impact [5]. Platform owners benefit from increased user engagement and retention, leading to higher ad impressions, user-generated content, and platform monetization opportunities.

In summary, given the rapid evolution of we-media platforms and the constant influx of new content types [6], it is essential to continually optimize and enhance personalized recommendation algorithms and models. This research aims to analyze existing algorithms and models used in personalized recommendation systems for we-media platforms and explore opportunities for optimization. By addressing these challenges and improving recommendation effectiveness, we can further enhance the user experience and contribute to the continued growth and success of we-media platforms.

### *1.2 Relevance of the Research*

Transitioning from the background information to the specific topic of personalized recommendation systems on we-media platforms, it is important to understand the existing algorithms that play a crucial role in delivering tailored content suggestions to users.

Collaborative filtering and content-based filtering are two distinct but complementary approaches in personalized recommendation systems. Collaborative filtering leverages user-item interaction data to identify patterns and similarities among users or items, while content-based filtering focuses on the semantic characteristics of items and user preferences [7].

By combining these two techniques, hybrid approaches can further enhance recommendation accuracy and coverage by incorporating additional factors such as demographical information or contextual data [8]. These factors provide additional context and personalization to the recommendations, resulting in an even stronger linkage between user preferences and recommended items.

It is important to note that these are just a few examples of existing algorithms used in personalized recommendation systems, and these methods will be elaborated in the following chapters since they provide an enhanced foundation for further development. In addition, the choice of algorithm depends on factors such as available data, system requirements, and the specific problem being addressed [9]. Researchers will continue to explore new algorithms and techniques to improve recommendation accuracy, diversity, and user satisfaction on we-media platforms.

### *1.3 Research objectives*

Building upon the understanding of existing algorithms in personalized recommendation systems on we-media platforms, the research objectives are highly relevant to these existing models.

1) To survey and evaluate the existing algorithms and models employed in we-media platforms for personalized recommendations, by examining these algorithms and models, we can gain insights into their effectiveness, limitations, and applicability to we-media platforms. We will compare their performance, scalability, and ability to handle challenges such as data sparsity, cold start problems, and real-time recommendation requirements.

2) To identify optimization opportunities in the existing algorithms and models for personalized recommendation in we-media platforms, through critical analysis, we aim to pinpoint specific areas or aspects where improvements can be made to enhance their performance, effectiveness, and efficiency.

3) To explore and propose optimization strategies for personalized recommendation algorithms in we-media platforms. This objective involves investigating various strategies, such as content-based filtering techniques, collaborative filtering methods, hybrid approaches, integration of machine learning and deep learning, contextual and sequential modeling, and the incorporation of user feedback and preferences. We will evaluate the potential of these strategies to overcome the limitations and challenges faced by current algorithms and models.

4) To analyze and interpret the results obtained from the experiments, this objective involves analyzing the performance metrics, user satisfaction, and engagement indicators to assess the impact of the optimized algorithms and models. We will critically evaluate the effectiveness of the optimizations in enhancing the user experience and content engagement within we-media platforms.

### *1.4 Significance of the Research*

By fulfilling these research objectives, this study holds the potential to significantly contribute to the field of personalized recommendation systems on we-media platforms and benefit both users and developers.

First, by improving user experiences, personalized recommendations can provide users with more relevant, diverse, and engaging content. This leads to increased user satisfaction, longer session durations, and higher user retention rates. Users benefit from a platform that understands their preferences and delivers content tailored to their interests, resulting in a more enjoyable and personalized experience.

Additionally, content creators are empowered through optimized recommendations. With improved visibility, creators can reach a larger audience, gain more exposure, and expand their influence. This fosters creativity, promotes diversity in content creation, and provides opportunities for emerging talent to showcase their work on we-media platforms.

Finally, the study contributes to the research field by analyzing and comparing existing algorithms and models used in personalized recommendation systems. Understanding the strengths and limitations of current approaches helps identify areas for optimization and guides future research endeavors. This research-driven approach spurs advancements in recommendation algorithms, benefiting both academic researchers and industry practitioners.

In conclusion, optimizing personalized recommendation algorithms in we-media platforms has significant benefits across different dimensions. It enhances user experiences, empowers content creators, and contributes to advancements in the research field. By prioritizing these aspects, we can create more effective and personalized recommendation systems that foster user satisfaction, support content creators, and drive the evolution of recommendation algorithms.

## 2. Data Collection

First, we need to collect statistics for the database. In this case, we chose several movie recommendation applications, such as Douban, Rotten Tomato, and IMDB, as our information sources [10-13]. The collection of data for a movie recommendation app involves gathering a variety of specific attributes related to movies. These attributes are crucial in providing accurate and personalized recommendations to users. One of the primary types of data collected is movie information. This includes details such as movie titles, release dates, directors, genres, and runtimes. In addition, user-generated ratings and reviews play a significant role in assessing the quality and popularity of movies. Additionally, detailed plot summaries and descriptions are essential attributes collected for each movie. Thus, these attributes can be included in the database and allow more quantitative analysis.

### *2.1 Set up Database*

Second, the database needs to be set up. This involves selecting a suitable database management system such as MySQL or PostgreSQL. Once the database management system is chosen, we can proceed with creating the necessary tables and defining their relationships. We need to set up the database tables to store the necessary data. We create three tables: Users, movies, and ratings. The Users table stores user information, the Movies table stores movie information, and the rating table stores user ratings for different movies. By structuring our data in this way, we can easily retrieve and analyze it for our recommendation system.

**Table 1**. Movie Database.

| Table Name | Attribute1 | Attribute 2 | Attribute 3 |
|---|---|---|---|
| Users | UserID | Username | |
| Movies | MovieID | Title | Genre |
| Ratings | UserID | MovieID | Rating |

With these formalized tables (Table 1), data can be efficiently queried by SQL lines, which enables the utilization of more complex recommendation systems.

## *2.2 Implementation of recommendation algorithms in SQL*

Accordingly, we need to create SQL lines that can serve as each recommendation system. Collaborative filtering (Algorithm 1) is a technique that recommends items to users based on the preferences of similar users. In this step, we calculate similarities between users based on their shared ratings for movies. By finding users who have rated movies similarly, we can recommend unseen movies to a target user based on what similar users have liked. To calculate these similarities, we join the Ratings table with itself on movie_id, excluding the same users. Using the dot product of their ratings, we calculate the similarity score. This calculation is performed for every pair of users, and the results are stored in the UserSimilarities table. Each row in this table represents a pair of users and their similarity score. To recommend movies to a specific user, we join the Movies, Ratings, and UserSimilarities tables, retrieving movies rated by similar users and ordering them by the similarity score.

**Algorithm 1. Collaborative Filtering in Pseudocode**

```
//Calculate similarities between users
CREATE TABLE UserSimilarities {
    user1: INTEGER,
    user2: INTEGER,
    similarity: DECIMAL(10, 8)
}

FOR EACH u1 IN Ratings:
    FOR EACH u2 IN Ratings:
        IF u1.movie_id == u2.movie_id AND u1.user_id != u2.user_id:
            numerator = 0
            denominator1 = 0
            denominator2 = 0
            FOR EACH r1 IN u1.ratings:
                FOR EACH r2 IN u2.ratings:
                    IF r1.movie_id == r2.movie_id:
                        numerator += r1.rating * r2.rating
                        denominator1 += r1.rating * r1.rating
                        denominator2 += r2.rating * r2.rating
            similarity = numerator/(SQRT(denominator1) * SQRT(denominator2))
            INSERT INTO UserSimilarities(user1, user2, similarity)

// Get recommendations for a specific user (e.g., user_id = 1)
recommendations = []
FOR EACH m IN Movies:
    FOR EACH r IN Ratings:
        FOR EACH s IN UserSimilarities:
            IF r.movie_id == m.movie_id AND r.user_id == s.user2 AND s.user1 == 1:
                recommendations.append(m.title)
SORT recommendations BY s.similarity DESC
recommendations = recommendations [0:10]
```

Content-based filtering (Algorithm 2) recommends items to users based on the attributes or features of the items they have interacted with in the past. In this step, we calculate genre preferences for each user based on their rated movies' genres. By analyzing the genres that a user prefers, we can recommend similar movies to them. To calculate these preferences, we join the Ratings table with the Movies table on movie_id. We then group the data by user_id and genre, counting the occurrence of each genre in their rated movies. The results are stored in the UserGenrePreferences table, where each row represents a user, a genre, and the preference count. To recommend movies to a specific user, we join the Movies, Movies, and UserGenrePreferences tables, retrieving movies with similar genres to those preferred by the user. The movies are ordered by the preference count to prioritize the most preferred genres.

**Algorithm 2. Content-based Filtering in Pseudocode**

```
// Calculate genre preferences for each user
CREATE TABLE UserGenrePreferences
FOR EACHrating in Ratings:
movie_id = rating.movie_id
user_id = rating.user_id
genre = GetGenreFromMovie(movie_id)
IncrementPreferenceCount(user_id, genre)
// Get recommendations for a specific user (e.g., user_id = 1)
recommendations = []
FOR EACH rating in Ratings:
movie_id = rating.movie_id
user_id = rating.user_id
genre = GetGenreFromMovie(movie_id)
preference_count = GetPreferenceCount(user_id, genre)
AddToRecommendations(recommendations, movie_id, preference_count)

SortRecommendations(recommendations)
recommended_movies = GetTopMovies(recommendations, 10)
```

The hybrid method is a combination of these two algorithms. Thus, to achieve this algorithm, we need several procedures to implement it with similar codes. First, we execute an SQL query to retrieve essential information about movies from our database. This includes attributes such as movie titles, release dates, directors, genres, and runtimes. Additionally, we employ descriptive words to explain how these data help categorize and organize movies within the app. These attributes allow users to easily search and discover movies based on their preferences. Next, we incorporate user-generated ratings and reviews into our data collection process. We store attributes such as user ratings, review scores, and sentiment analysis results in our database. These attributes provide valuable insights into movie quality and popularity, aiding in personalized recommendations. The descriptive words convey how user interactions shape the app's movie suggestions. Furthermore, we gather attributes related to the cast and crew members. This includes the names of actors, actresses, directors, and writers involved in each movie. We store this information in our database to enable users to explore movies based on their favorite artists. Here is an example SQL query to fetch these data. Finally, all the recommendation algorithms are implemented in SQL, and further evaluation can be applied.

## *2.3 Import Clustering K-means Model*

K-means clustering is a popular unsupervised machine learning algorithm. It efficiently partitions data into distinct clusters based on their similarities. The algorithm aims to minimize the sum of squared distances between data points and their respective cluster centers [13]. The k-means clustering model involves several steps. First, initialization is performed by randomly selecting k data points as initial centroids or cluster centers. Second, each data point is assigned to the nearest centroid based on the chosen distance metric, forming the initial clustering.

Next, the centroids are updated by recalculating their positions based on the mean position of the data points assigned to each cluster. This iterative process of assignment and update is repeated until convergence, which occurs when the centroids no longer move significantly or after a specified number of iterations. Finally, the output of the k-means clustering model is the set of clusters, where each data point belongs to a specific cluster based on its proximity to the corresponding centroid.

The k-means clustering model finds applications in various domains, such as customer segmentation, image compression, anomaly detection, and recommendation systems. It provides a simple yet effective way to organize large datasets into meaningful groups, allowing for further analysis and decision-making based on the identified clusters.

Now, we will implement it in SQL. Clustering techniques group similar items or users together based on their characteristics. In this step, we apply k-means clustering on user preference vectors to group users with similar movie preferences into clusters. By clustering users, we can recommend movies to a specific user within the same cluster, as users within a cluster are likely to have similar tastes. First, we generate user preference vectors by joining the Ratings table with the Movies table and assigning binary values (1 or 0) to each genre based on whether the user has rated a movie in that genre. These vectors serve as input for the k-means clustering algorithm. By applying k-means clustering, we assign each user to a particular cluster based on their preference vector. The resulting clusters are stored in the UserClusters table, where each row represents a user and their assigned cluster number. To recommend movies to a specific user, we join the Movies, Ratings, and UserClusters tables, retrieving movies rated by users within the same cluster and sorting them by their ratings. The recommended movies are then presented to the user. The pseudocode (Algorithm 3) and flowchart in Figure 1 are shown as follows:

**Algorithm 3. K-means Model in Pseudocode**

```
//Generate user preference vectors
user_preference_vectors = SELECT user_id,
IF genre = 'Action' THEN 1 ELSE 0 END AS action,
IF genre = 'Comedy' THEN 1 ELSE 0 END AS comedy,
IF genre = 'Drama' THEN 1 ELSE 0 END AS drama
FROM ratings
JOIN movies ON ratings.movie_id = movies.movie_id;

//Apply k-means clustering
user_clusters = SELECT user_id, CLUSTER_NUMBER() OVER () AS cluster_number
FROM user_preference_vectors
CLUSTER BY (action, comedy, drama);
```

```
//Get recommendations for a specific user within the same cluster
specific_user_recommendations = SELECT movies.title
FROM movies
JOIN ratings ON movies.movie_id = ratings.movie_id
JOIN user_clusters ON ratings.user_id = user_clusters.user_id
WHERE user_clusters.cluster_number = (SELECT cluster_number FROM
user_clusters WHERE user_id = 1)
ORDER BY ratings.rating DESC
LIMIT 10;
```

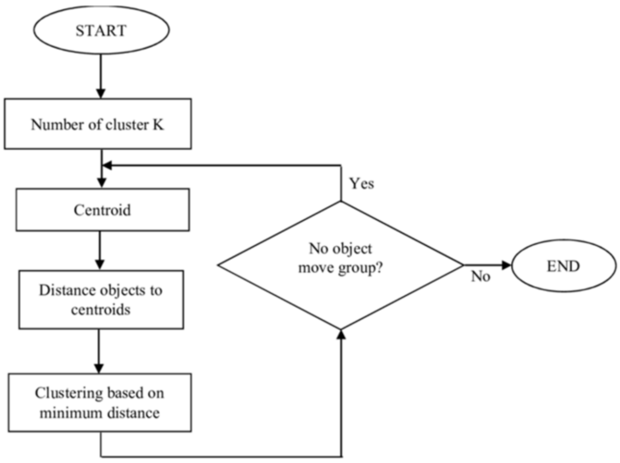
START
Number of cluster K
Centroid
Distance objects to centroids
Clustering based on minimum distance
No object move group?
Yes
No
END

**Figure 1.** K-means Model Flowchart [14].

## 3. Data Standardization

To standardize and organize data in a movie database, several important steps need to be followed. First, it is essential to calculate the mean and standard deviation of specific columns to understand the distribution and variability of the data. Once the mean and standard deviation have been calculated, the data can be standardized. Standardization is a process that transforms the data to have a mean of 0 and a standard deviation of 1, making it easier to compare and analyze. Alternatively, if min-max scaling is preferred, it is necessary to calculate the minimum and maximum values of a column. Once the minimum and maximum values have been determined, the data can be scaled within the range of 0 to 1.

By following these steps, the movie database can be organized and standardized effectively. It allows for consistent measurement and comparison of variables such as movie titles, release years, genres, directors, and ratings. Standardizing and organizing the data systematically ensures that reliable and comparable information is available for analysis and decision-making purposes within the movie database.

However, due to the limited sample size, the outcome may not be precise enough to draw effective conclusions. In this case, confidence intervals can be a valuable mathematical approach to ensure reliable insights. Confidence intervals provide a range of likely values for population parameters related to movie characteristics, enabling us to quantify the uncertainty around our estimates.

For example, we consider estimating the average user ratings for a specific genre of movies in the database. By calculating confidence intervals, we can determine a range within which we expect the true population mean of user ratings to fall. This interval allows us to make informed inferences about the overall perception of movies within that genre, even with limited sample data.

To calculate the confidence interval, we use the following formula:

$$Confidence\ Interval = Sample\ Mean \pm (Critical\ value \times Standard\ Error) \tag{1}$$

The sample mean represents the average rating obtained from the limited sample data available for the genre. The critical value is determined based on the desired level of confidence, such as 95% confidence corresponding to a critical value of 1.96 for a large sample size. The standard error measures the variability in the sample data and can be computed using the formula:

$$Standard\ Error = Standard\ Deviation\ /\ \sqrt{(Sample\ Size)} \tag{2}$$

By incorporating confidence intervals in our analysis, we can assess the precision of our estimates and gauge the reliability of our findings regarding user ratings for movies in a particular genre. A wider confidence interval indicates greater uncertainty, suggesting that our estimate may be less precise due to the limited sample size. Conversely, a narrower interval implies a more precise estimate with reduced uncertainty, allowing us to draw stronger conclusions about the average user ratings.

### *3.1 Metrics*

To measure similarity in a movie recommendation system, various algorithms can be used. One common method is collaborative filtering in Figure 2, which calculates the cosine similarity between users' ratings or interactions with movies [15]. The cosine similarity formula is as follows:

$$cosine\_sim(u, v) = dot\_product(u, v)\ /\ (||u|| * ||v||) \tag{3}$$

Here, $u$ and $v$ represent two users, and $dot\_product(u, v)$ calculates the dot product of their rating vectors. $||u||$ and $||v||$ denote the Euclidean norms of the rating vectors. The resulting cosine similarity value ranges from -1 to 1, where 1 indicates high similarity, 0 denotes no similarity, and -1 implies dissimilarity.

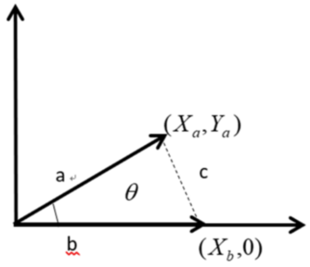

**Figure 2.** Cosine Similarity Visualization [16].

For example, if we have two users, User A and User B, and their rating vectors are as follows:

User A: [4, 3, 5, 0, 2]

User B: [3, 2, 4, 1, 0]

To calculate the cosine similarity between these users, we first compute the dot product:

$dot\_product$(A, B) = (43) + (32) + (54) + (01) + (2*0) = 45

Next, we calculate the Euclidean norms:

||A|| = $sqrt$((4^2) + (3^2) + (5^2) + (0^2) + (2^2)) = $sqrt$(54) ≈ 7.35

||B|| = $sqrt$((3^2) + (2^2) + (4^2) + (1^2) + (0^2)) = $sqrt$(30) ≈ 5.48

Using these values, we can compute the cosine similarity:

$cosine\_sim$(A, B) = (7.35 * 5.48)/45 ≈ 0.89

This cosine similarity score indicates some degree of similarity between User A and User B in their movie preferences. To evaluate the stability of a recommendation system, mean average precision (MAP) is commonly used. MAP measures the average precision across different recommendation lists for a given user [17]. The formula for calculating MAP is as follows:

$$MAP = (1 / |U|) * \Sigma(P(u)) \tag{4}$$

Here, $|U|$ represents the total number of users, and $P(u)$ is the precision for each user $u$. Precision is computed by comparing the recommended movies with the actual user ratings and determining how many relevant recommendations were made. However, to obtain the most accurate result and analyze the variation in stability of each algorithm, a precision-recall curve, which can show both the similarity and stability, should be plotted. With the method followed in Figure 3, we can calculate the precision and recall.

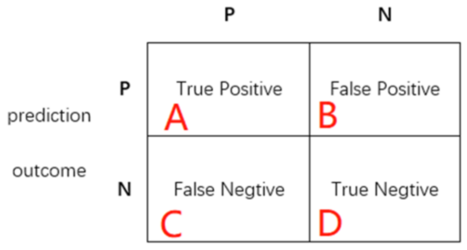

**Figure 3**. Precision-Recall Deduction.

$$Precision = \frac{A}{A+B};\ Recall = \frac{A}{A+C} \tag{5}$$

For example, we have a test set of 100 users, and for a specific user, 10 movies are recommended. Upon comparing these recommendations with the user's actual ratings, it is found that 5 movies are relevant. The precision for this user is 5/10 = 0.5.

By calculating such precision for all users and averaging them, MAP can be obtained. A higher MAP value indicates a more stable recommendation system with better overall precision.

To measure response time in a movie recommendation system, various factors need to be considered. One approach is to focus on the time taken from when a user request is received until the recommendations are delivered. This can be divided into three key stages: data preprocessing, algorithm computation, and result retrieval.

For instance, if the data preprocessing stage takes an average of 50 milliseconds, the algorithm computation takes 200 milliseconds, and the result retrieval takes 100 milliseconds, the total response time would be 350 milliseconds. Monitoring and optimizing these individual stages can help improve the overall response time of the system.

Furthermore, throughput is crucial in evaluating response time. Supposing the system can handle 100 user requests per second without significant delays, this indicates a throughput of 100 recommendations per second. By simulating concurrent user requests and monitoring the system's performance under different loads, the maximum throughput and response time can be determined. This helps evaluate the efficiency and scalability of the recommendation system, ensuring that it can handle increasing user demands without performance degradation. With these metrics, we can reasonably analyze the advantages and limitations of these three algorithms.

## *3.2 Results*

From the research results in Table 2, it is not difficult to see that collaborative filtering, as the earliest developed algorithm, is not comparable to content-based and hybrid methods in terms of similarity. In contrast, the hybrid, as a mature algorithm, performs well in this field. In terms of stability, compared with the other two algorithms, the content-based algorithm is the most unstable. Finally, in terms of response time, collaborative filtering is slightly inferior to the others.

**Table 2**. Result I (Evaluation in Three Dimensions).

| | Average Similarity (0-1) | Stability (0-1) | Average Response time |
|---|---|---|---|
| Collaborative filtering | 0.78 | 0.90 | 0.32ms |
| Content-based | 0.81 | 0.69 | 0.27ms |
| Hybrid | 0.85 | 0.88 | 0.29ms |

In addition to Table 2, the precision-recall curve in Figure 4 provides deeper insight. According to the graph, collaborative filtering and content-based filtering are decent in terms of the degree of stability performance because their curves are very smooth. B, on the other hand, is constantly changing, most notably at recall = 0.5. Content-based methods have excellent similarity from recall = 0.5 to 1 as the largest area under the curve.

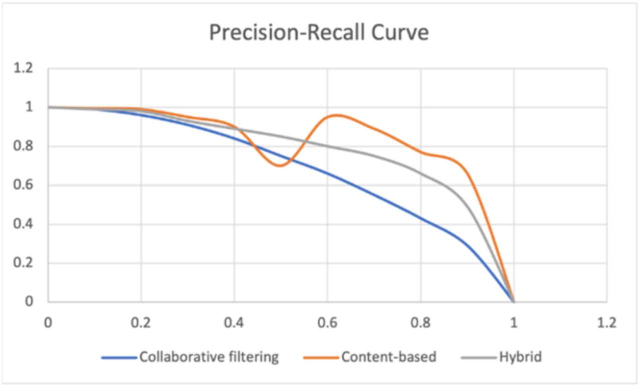

**Figure 4**. Result II (Precision-Recall Curve).

With all this information, the interpretation of the results follows. First, the result of collaborative filtering and hybrid filtering is not unexpected, as the hybrid tends to be a well-improved algorithm in almost every aspect based on collaborative filtering. However, the randomness in the content-based algorithm is more valuable to study. At recall = 0.5, the precision of the algorithm decreases drastically. We believe and have verified that this is not an experimental error but a limitation of the algorithm. As recall means the portion of the true positive in the positive set, permutations of possible parameters will be maximized when the portion reaches 0.5. For example, there are 10 positive tags for users to choose from, but each user will choose randomly according to their preferences. Thus, when they choose 5 or 6 as true positive tags, the permutation will be 10P5=10P6, which is the maximum number of permutations allowed. The probability of the algorithm to output the precise result will be much lower. In addition, the high similarity between recall = 0.5 and 1 can be explained by the same example. The more one chooses to be the true positive tag, the larger the recall portion is, and thus, the higher the precision is.

## 4. Conclusion

With the continuous development of we-media, recommendation algorithms will also continue to evolve. The main achievement of this research is to summarize the characteristics and development trends of each algorithm, which is convenient for future generations to make further choices. Additionally, from the evaluation results, we also perceive the major drawbacks of the recommendation algorithm. The accuracy of the algorithm is constantly improving, and modern user participation algorithms, such as likes and small tags, are even more capable of improving accuracy. However, does increasing the accuracy of algorithms truly improve society [18]? There are many comments that the major we-media industry, by constantly iterating algorithms to improve user stickiness, may do more harm than good to the users themselves. The most typical is the information cocoon effect [19]. The more users rely on we-media products, the farther away they are from comprehensive social information, and at the same time, some users will lose their potential consumption power [20]. To improve this problem, future researchers should unite with the social sciences, psychology, and communication and no longer simply focus on the accuracy of algorithms to maximize social benefits.

## References

[1] Bobadilla, Jesús, et al. "Recommender systems survey." Knowledge-based systems 46 (2013): 109-132.
[2] Lu, Jie, et al. "Recommender system application developments: a survey." Decision Support Systems 74 (2015): 12-32.
[3] Cambazoglu, B. Barla, and Ricardo Baeza-Yates. Scalability challenges in web search engines. Springer Nature, 2022.
[4] Valkenburg, Patti M. "Social media use and well-being: What we know and what we need to know." Current Opinion in Psychology 45 (2022): 101294.
[5] Milano, Silvia, Mariarosaria Taddeo, and Luciano Floridi. "Recommender systems and their ethical challenges." Ai & Society 35 (2020): 957-967.
[6] Sarwar, Badrul, et al. "Item-based collaborative filtering recommendation algorithms." Proceedings of the 10th International Conference on Worldwide Web. 2001.

[7] Yang, Yong, and Young Chun Ko. “Design and application of handicraft recommendation system based on improved hybrid algorithm.” International Journal of Pattern Recognition and Artificial Intelligence 36.02 (2022): 2250008.

[8] Zhang, An, et al. “Incorporating bias-aware margins into contrastive loss for collaborative filtering.” Advances in Neural Information Processing Systems 35 (2022): 7866-7878.

[9] Zhang, An, et al. “Incorporating bias-aware margins into contrastive loss for collaborative filtering.” Advances in Neural Information Processing Systems 35 (2022): 7866-7878.

[10] Douban. (2023) https://www.douban.com

[11] Rotten Tomato. (2023) https://www.rottentomatoes.com

[12] Imdb, (2023). https://www.imdb.com

[13] Ikotun, A. M., Ezugwu, A. E., Abualigah, L., Abuhaija, B., & Heming, J. (2022). K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. Information Sciences.

[14] Saldarriaga-Zuluaga, S.D.; López-Lezama, J.M.; Muñoz-Galeano, N. Optimal Coordination of Over-Current Relays in Microgrids Using Principal Component Analysis and K-Means. *Appl. Sci.* 2021, *11*, 7963. https://doi.org/10.3390/app11177963

[15] Singh, R. H., Maurya, S., Tripathi, T., Narula, T., & Srivastav, G. (2020). Movie recommendation system using cosine similarity and KNN. International Journal of Engineering and Advanced Technology, 9(5), 556-559.

[16] Zhihun. (2023) https://zhuanlan.zhihu.com/p/154108167

[17] Ferrari Dacrema, M., Cremonesi, P., & Jannach, D. (2019, September). Are we truly making much progress? A worrying analysis of recent neural recommendation approaches. In Proceedings of the 13th ACM conference on recommender systems (pp. 101-109).\

[18] Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., & Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: an overview. Bioinformatics, 16(5), 412-424.

[19] Peng, H., & Liu, C. (2021). Breaking the information cocoon: when do people actively seek conflicting information?. Proceedings of the Association for Information Science and Technology, 58(1), 801-803.

[20] Li, J., Zhao, H., Hussain, S., Ming, J., & Wu, J. (2021, March). The dark side of personalization recommendation in short-form video applications: an integrated model from information perspective. In International Conference on Information (pp. 99-113). Cham: Springer International Publishing.