big data and cognitive computing

MDPI

Article

Adaptive KNN-Based Extended Collaborative Filtering Recommendation Services

Luong Vuong Nguyen [1], Quoc-Trinh Vo [1] and Tri-Hai Nguyen [2,*]

[1] Department of Artificial Intelligence, FPT University, Da Nang 550000, Vietnam; vuongnl3@fe.edu.vn (L.V.N.); trinhvq@fe.edu.vn (Q.-T.V.)
[2] Department of Computer Science and Engineering, Seoul National University of Science and Technology, Seoul 01811, Republic of Korea
* Correspondence: haint93@seoultech.ac.kr

Abstract: In the current era of e-commerce, users are overwhelmed with countless products, making it difficult to find relevant items. Recommendation systems generate suggestions based on user preferences, to avoid information overload. Collaborative filtering is a widely used model in modern recommendation systems. Despite its popularity, collaborative filtering has limitations that researchers aim to overcome. In this paper, we enhance the K-nearest neighbor (KNN)-based collaborative filtering algorithm for a recommendation system, by considering the similarity of user cognition. This enhancement aimed to improve the accuracy in grouping users and generating more relevant recommendations for the active user. The experimental results showed that the proposed model outperformed benchmark models, in terms of MAE, RMSE, MAP, and NDCG metrics.

Keywords: adaptive K-nearest neighbor; collaborative filtering; recommendation system; user cognition

check for updates

Citation: Nguyen, L.V.; Vo, Q.-T.; Nguyen, T.-H. Adaptive KNN-Based Extended Collaborative Filtering Recommendation Services. *Big Data Cogn. Comput.* **2023**, *7*, 106. https://doi.org/10.3390/bdcc7020106

Academic Editors: Konstantinos Kotis and Dimitris Spiliotopoulos

Received: 28 April 2023
Revised: 29 May 2023
Accepted: 30 May 2023
Published: 31 May 2023

1. Introduction

Collaborative filtering techniques are commonly employed by recommendation systems to provide personalized recommendations based on the prior behavior of users and their preferences [1–4]. However, recent studies have identified several drawbacks and challenges that need to be addressed. One significant problem is the cold start issue, where the system struggles to make recommendations for new items or users with no prior interaction history, resulting in poor performance and usability for novice users [5,6]. To overcome this challenge, innovative strategies such as a hybrid recommendation system combining collaborative filtering with content-based algorithms and performance enhancements are being developed [2,4,5]. Another challenge faced by collaborative filtering is the sparsity problem, where there are not enough overlapping user–item interactions to produce reliable recommendations, which can occur in large datasets or for specialized products that are not widely used. Several approaches have been proposed to address this issue, such as matrix factorization models, clustering techniques, and graph-based algorithms [7]. However, collaborative filtering outcomes are not always transparent or easily interpretable, which is another drawback. It can be challenging to understand how recommendations are made and what influences collaborative filtering models.

The KNN-based collaborative filtering algorithm is a widely used technique in recommendation systems. The fundamental principle of collaborative filtering is to predict user preferences by identifying other users with similar preferences and using their ratings to make recommendations. The KNN-based collaborative filtering algorithm is a type of collaborative filtering that assigns ratings to items by leveraging the ratings of the K most similar users to the target user. The KNN algorithm measures the similarity between users using a distance metric (e.g., Euclidean distance, cosine similarity, and Pearson correlation coefficient). It then identifies the K users who are most similar to the target user and calculates a weighted average of their ratings for each item. These weights are determined

using similarity scores between the target user and each of the K users. The predicted rating for an item is the weighted average of the ratings of K users. Figure 1 illustrates an example of how KNN-based collaborative filtering is used.

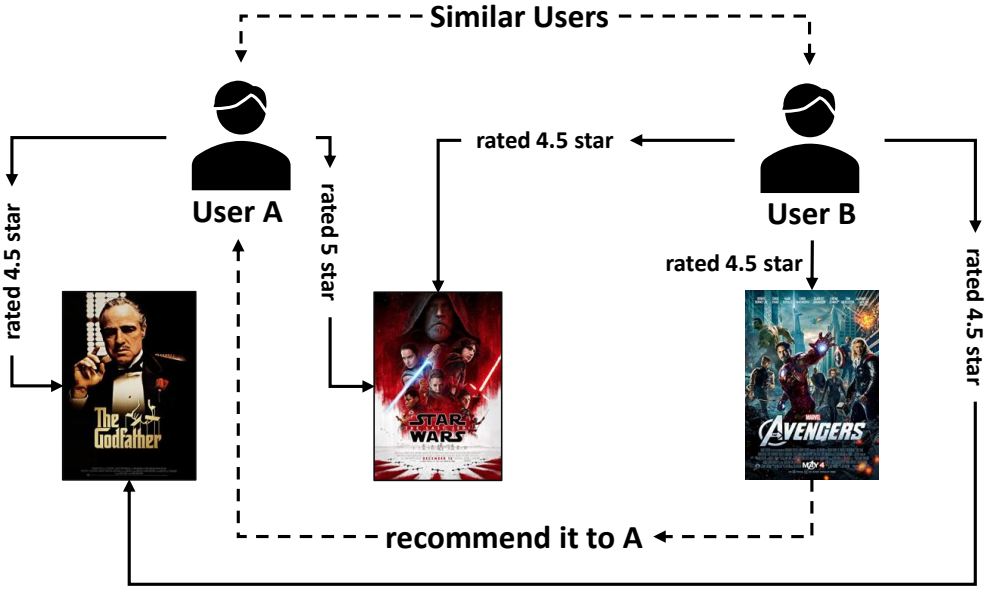

Figure 1. An example of collaborative filtering with the KNN algorithm.

Suppose we have a dataset of movie ratings from various users. The objective is to recommend new movies to each user based on their past preferences. To represent the data, we can create a matrix where each row corresponds to a user and each column represents a movie. The matrix entries contain the ratings given by each user to each movie. To implement collaborative filtering with KNN, we first select a user for whom we want to make recommendations. We then find the K-nearest neighbors of that user based on their past ratings, employing a similarity measure. Once the K-nearest neighbors have been identified, we use the ratings of the neighbors to predict the ratings of the target user for movies they have not yet watched, taking a weighted average of the neighbor ratings, where the weights are the similarities between the target user and the neighbors. Finally, based on the predicted ratings, we recommend the top-rated movies to the target user. For example, in Figure 1, user A has rated the movies "*The Godfather*" and "*Star Wars*" highly, and user B has also rated those movies highly, then we might recommend other similar movies (highly rated by B, "*Avengers*") to user A, based on the ratings of user B. This example illustrates the effectiveness of collaborative filtering based on the KNN algorithm. However, due to the high dimensionality of the data, the KNN algorithm is computationally expensive for large datasets, making it impractical for real-world applications. Finally, the algorithm may suffer from the so-called popularity bias, where items with a large number of ratings are more likely to be recommended, obscuring other items that may interest users.

In this paper, we propose an adaptive recommendation framework that leverages user cognition to provide more practical recommendations. Specifically, our framework aims to overcome the cold start problem by extending the collaborative filtering model with an adaptive KNN algorithm. The proposed recommendation framework consists of three steps. First, the adaptive KNN algorithm is used to cluster users based on their past interactions with the system. Second, we incorporate a user cognition parameter into the cosine similarity metric to dynamically update the user clusters. After updating the user clusters, we generate personalized predicted items to recommend to the active user. Our approach offers a more accurate and personalized recommendation system, particularly for new users who have limited interactions with the system.

The remainder of this paper is as follows. In Section 2, we provide an overview of the KNN algorithm and recent research on recommendation systems. Section 3 presents

our proposed method, which is an adaptive KNN-based collaborative filtering model for recommendation services. We describe the algorithm and explain how it overcomes the limitations of traditional collaborative filtering models. In Section 4, we present the experimental results and a discussion of the findings. Finally, Section 5 concludes the work and outlines future research.

2. Related Work

Most research on recommendation systems today is based on collaborative filtering [8,9]. Collaborative filtering can be divided into two main types: memory-based and model-based algorithms. Memory-based algorithms find users with similar preferences and generate recommendations by analyzing the neighborhood of the target user [9]. These algorithms use various similarity functions, such as the Pearson correlation coefficient (PCC), to measure the similarity between users or items and then calculate a weighted average of ratings provided by neighboring users, to make predictions. While memory-based algorithms benefit from the most recent information, processing many neighbors can be computationally expensive, especially for large user databases. To address this issue, model-based algorithms have been developed [10,11] that incorporate data mining methods to create a model of user ratings and forecast preferences using those models. However, these methods require significant computational resources to process large datasets. Recently, hybrid methods that combine collaborative and content-based filtering have been introduced. The inclusion of semantic information has been used to formalize and classify user attributes and products [12]. These methods aim to address the limitations of traditional collaborative filtering algorithms and improve the accuracy of recommendations.

Recently, several versions of the KNN algorithm have been introduced, each with its own set of advantages and disadvantages. For instance, a pattern classification method utilizing two nearest neighbors, specifically the nearest neighbor line and the nearest neighbor plane, was introduced by Zheng et al. [13]. In the field of nearest neighbor classification, Gao and Wang [14] proposed a center-based method, while Cevikalp et al. [15] explored the leaping hyper disk of each training class. The efficiency of the KNN classifier was investigated by Hernández-Rodríguez et al. [16], who suggested utilizing a tree structure to select the K most similar neighbors, for improved performance. Zhou and Yu [17] introduced an ensemble framework based on the KNN classifier, while Domeniconi and Yan [18] examined the KNN ensemble method and its correlation with error and accuracy. Altinçay [19] designed a multimodal perturbation-based nearest neighbor classifier ensemble through experimentation. Yang et al. [20] used a KNN classifier to classify hyperspectral image data.

Several personalized recommendation systems have been developed in recent years, each with a distinct approach and strengths. Subramaniyaswamy et al. [21] developed a system that uses a domain-specific ontology and an adaptive KNN algorithm, outperforming associative classification algorithms and solving sparsity and cold start issues. Zhang et al. [22] introduced a collaborative user network embedding (CUNE) approach that efficiently identified the top-k semantic friends of users, enhancing traditional model-based recommendation systems. Feng et al. [23] presented a co-cluster-based user–item community detection recommendation system (UICDR) that partitions subgroups of users and items based on their interactions using co-clustering. Walek et al. [24] developed a hybrid recommendation system called Predictory, combining singular value decomposition (SVD)-based collaborative filtering, content-based systems, and a fuzzy expert system. Bathla et al. [25] proposed the AutoTrustRec system, which combines autoencoders and social trust to improve recommendation accuracy. The INHeritance-Based Prediction (INH-BP) technique was proposed by Alhijawi et al. [26] to address recommendation system problems through the use of a genetic algorithm applied to collaborative filtering. Experimental evaluation of the MovieLens datasets [27], which describe the expressed preferences of users for movies and are provided by GroupLens Research at the University of Minnesota, demonstrated that the INH-BP method outperformed traditional techniques.

In addition, several studies have contributed to the field of recommendation systems by addressing various aspects, such as user perception, dynamic modeling, trust inference, contextualization, and algorithm comparison. They provided insights into different techniques and approaches, to enhance the accuracy and effectiveness of recommendation systems in diverse domains. Mican and Sitar-Taut [28] examined how the perceived value of information sources and recommendation systems affected the buying propensity of consumers. An attention-based dynamic user modeling and deep collaborative filtering recommendation technique was proposed by Wang et al. [29]. They made better recommendations for collaborative filtering by utilizing attention mechanisms to capture the changing interests of users. MRS OZ, an organizational recommendation system for Internet commerce, was first presented by Sitar-Taut and Mican [30]. The Onicescu method and Zipf's law were the foundation of the system for creating tailored suggestions for users in e-commerce scenarios. A method for a recommendation systems that is knowledge-driven and uses digital nudging was given by Sitar-Taut et al. [31]. To include domain information in the recommendation process and effectively influence user preferences, their approach used a modified Onicescu technique. For the purpose of capturing the evolution of user preferences over time, Koren [32] presented a collaborative filtering technique with temporal dynamics. By taking into account the temporal elements of user–item interactions, the strategy sought to increase the recommendation accuracy.

Furthermore, collaborative filtering recommendation algorithms for e-commerce were contrasted by Huang et al. [33]. They assessed several algorithms and went over their advantages and disadvantages in relation to e-commerce applications. AgreRelTrust was created by Zahir et al. [34] as a trust inference model for memory-based collaborative filtering recommendation systems. The approach was designed to infer implicit trust ties among users, to improve recommendation accuracy. A collaborative filtering recommendation system based on user attributes and TF-IDF (term frequency-inverse document frequency) was suggested by Ni et al. [35]. They used a technique that took into account both user and item attributes, to enhance the quality of suggestions. Guo and Lu [36] proposed a collaborative, contextual filtering recommendation approach, which included user interest drift features. The model generated context-aware suggestions, while taking into account the evolution of user preferences over time. A recommendation system that combines SVD and weight point rank (WPR) methods was proposed by Widiyaningtyas et al. [37]. Their strategy used item ranking and matrix factorization techniques to increase the suggestion accuracy. To solve the cold start issue, Hasan and Roy [38] suggested an item–item collaborative filtering recommendation system that made use of genre and trust information. In the absence of user preferences, their approach relied on user connections of trust and factored in the item genre similarity to produce suggestions.

3. Adaptive KNN-Based Extended Collaborative Filtering Model

In this section, we first present the basic KNN-based collaborative filtering algorithm and then propose a new collaborative filtering model based on an adaptive KNN algorithm with a user cognition concept.

3.1. Preliminaries

Recommendation systems based on collaborative filtering aim to recommend items of high interest to users by either predicting the rating that a user would give to an item or estimating the probability of interaction between the user and the item. The recommendation process can be simplified as identifying the top N items for which a user u would provide the highest possible ratings and then presenting them to the user u. Mathematically, this can be defined as follows:

$$\mathrm{Top}(u, N) = \underset{i \in I}{\overset{N}{\mathrm{argmax}}}\ \hat{r}_{ui} \tag{1}$$

where I refers to the list of recommendable items, N is the number of items to recommend, and $\hat{r}_{ui}$ is the rating predicted by the recommendation system for user u for item i.

The main idea behind KNN-based collaborative filtering is to find the K most similar users to a target user based on their past behaviors and then recommend items that similar users have also liked or interacted with. The prediction of the rating for user u on item i is formulated as follows:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{v \in U_{ui}^K} \text{sim}(u,v)(r_{vi} - \bar{r}_v)}{\sum_{v \in U_{ui}^K} |\text{sim}(u,v)|} \quad (2)$$

where U_{ui}^K denotes the set of K nearest neighbors of user u who has rated item i, r_{vi} is the actual rating given by neighbor v for item i, and $\bar{r}_u$ and $\bar{r}_v$ are the average ratings of each user (u and neighbor v) calculated based on their rating history. The similarity $\text{sim}(u,v)$ between users is calculated using a distance metric, such as cosine similarity or Pearson correlation coefficient. However, the value of K in a KNN-based collaborative filtering algorithm is usually fixed and determined prior to the recommendation process. This may not be optimal, as different users may have different preferences and levels of similarity, and this inflexibility can lead to suboptimal recommendations. Additionally, KNN-based collaborative filtering may not perform well for large datasets or when there are sparse user–item interactions.

3.2. Adaptive KNN-Based Collaborative Filtering with User Cognition

Adaptive K-Nearest Neighbors (AKNN) is an improved version of the KNN algorithm. The traditional KNN uses a fixed distance metric to classify data points, which can lead to errors when dealing with high-dimensional data or when the structure of the data changes over time. AKNN addresses this issue by adjusting its distance metric based on the local structure of the data. AKNN achieves this by using a local density estimation approach, such as kernel density estimation or nearest neighbor density estimation. It estimates the local density around each data point in the training set and uses this information to calculate a weighted distance metric that considers the underlying structure of the data. AKNN can adapt to changes in the underlying structure of the data over time, leading to better classification outcomes. The AKNN algorithm involves several steps, as follows:

1. Find the appropriate K value: selecting K used for making predictions is usually achieved by cross-validation or other model selection techniques;
2. Estimate local density: AKNN estimates the local density around each data point using techniques such as kernel density estimation or nearest neighbor density estimation. This involves calculating a kernel density estimate using a predetermined function and bandwidth parameter or taking the average distance between each data point and its nearest neighbors;
3. Define an adaptive distance metric: after estimating the local density, AKNN creates an adaptive distance metric that considers the local structure of the data. Typically, the distance between pairs of data points is weighted based on their respective local densities;
4. Classify new data points: AKNN uses the adaptive distance metric to classify new data points based on their proximity to the nearest K neighbors. The class label is assigned based on a majority vote among the K nearest neighbors, weighted by their local densities.

We have developed a new recommendation model called ExtKNNCF, which utilizes the AKNN algorithm with user cognition parameters to create a list of top-N recommendations. The definition of user cognition is as follows:

User Cognition: *For a given user u who interacts within a specific domain of a recommendation system, we define the user cognition C_u as the set of items that the user u has interacted with. The value of each element in C_u is set to 1 if the user u confirms that the items are similar, and 0 otherwise. The user cognition for the user u is formulated as follows:*

$$C_u = \langle i^n | 1 \leqslant n \leqslant \mathbb{N} \rangle \tag{3}$$

The AKNN-based collaborative filtering algorithm is used in the prediction process, to generate a recommended list optimized for popularity and closeness, measured by cognitive similarities, to the preferences of the active user. The cognitive similarities are determined by estimating the Euclidean distance metric between users. The number of interactions with items by users is considered the user cognition and represented as a vector input in this stage. The Euclidean distance is then calculated to measure the distance between users, followed by the computation of cosine similarity to determine the most similar users in terms of their cognition. Let U_{ui}^K be the set of top-K similar neighbors of user u who have cognitive similarity data of item i. Given the cognitive similarity matrix C, the predicted cognitive similarity value for user u to item i is computed as follows:

$$\hat{c}_{ui} = d_{ui} + \frac{\sum_{v \in U_{ui}^K} (c_{vi} - d_{vi}) \operatorname{sim}(u, v)}{\sum_{v \in U_{ui}^K} |\operatorname{sim}(u, v)|} \tag{4}$$

where d_{ui} and d_{vi} are biased cognitive similarity values for user u and neighbor v to item i, respectively, and $\operatorname{sim}(u, v)$ is the cognitive similarity between user u and v. To avoid overfitting, the average of the Euclidean measurements is set as a cognitive threshold, which determines whether to consider user cognition. In addition, a cognitive weight is considered to represent the importance of user cognition in the final distance calculation. Algorithm 1 illustrates the proposed ExtKNNCF approach (we have incorporated syntax from Python and the Numpy library).

Algorithm 1 aims to use the cognitive similarities of a test instance with its nearest neighbors in the training dataset to predict its class label. Its input includes a training dataset **D**, a test instance x, the minimum and the maximum number of neighbors (K_{min} and K_{max}), and user cognition parameters (weight and threshold). The algorithm takes into account the user cognition parameters, including cognitive weight and cognitive threshold, to calculate the final distance between the neighbors and the test instance. The cognitive weight represents the importance of user cognition in the final distance calculation, while the cognitive threshold determines whether to consider user cognition. The algorithm uses the KNN algorithm to find the K nearest neighbors of the test instance in the training dataset, where K is initialized as K_{max}. It calculates the final distance between each neighbor and the test instance by considering their Euclidean distance and the similarity between their user cognition. It sorts the neighbors based on their final distances and adjusts K based on the feedback of the user. Then, it makes a prediction based on the neighbors with the new value of K. The algorithm exits the loop when K equals the minimum value of K ($K = K_{min}$) or the predicted class label differs from the class label of the nearest neighbor. Finally, it returns the predicted class label.

In summary, ExtKNNCF can dynamically adjust the value of K based on data sparsity. The algorithm consists of two main steps: neighborhood selection, and rating prediction. In the first step, the most relevant neighbors for a given user/item are identified based on their similarity scores (measured using cognitive similarities). In the second step, ExtKNNCF computes the predicted rating for a user/item by taking a weighted average of the ratings of its neighbors, where the weights are proportional to their similarity scores. These weights are adjusted according to the data sparsity using a density-based weighting scheme. The scheme assigns higher weights to closer neighbors in dense areas and lower weights to farther neighbors in sparse areas.

Algorithm 1 AKNN-based collaborative filtering with user cognition parameters (ExtKNNCF)

Require: Training dataset **D**; test instance x; minimum and maximum number of neighbors K_{min}, K_{max}; user Cognition parameters = {cognitive_weight, cognitive_threshold}
Ensure: The class label of x predicted by the ExtKNNCF algorithm

```
Initialize K = K_max
# Repeat until a class label is predicted
while True:
    # Find the K nearest neighbors of x in D, K is the number of neighbors
    neighbors = D[np.argsort(cdist(D, np.array([x])), :K]]
    # Calculate the final distance between the neighbors and x
    final_distances = []
    for neighbor in neighbors:
        # Calculate the Euclidean distance between the neighbor and x
        d_euc = np.linalg.norm(x − neighbor)
        # Calculate cosine similarity between the user cognition of the neighbor and x
        s_cog = cosine_similarity(x.cognition, neighbor.cognition)
        if s_cog < cognitive_threshold:
                final_distance = d_euc
        else:
                final_distance = (1 − cognitive_weight) * d_euc + cognitive_weight * s_cog
        final_distances.append((neighbor, final_distance))
    # Sort the final_distances list by final distance
    final_distances.sort(key=lambda x: x[1])
    # Checking K value to make prediction/exit loop
    if K == K_min:
        y_pred = np.argmax(np.bincount(neighbors[:, −1].astype(int)))
        break
    K −= 1
    if K < K_min:
        K = K_max
    if K == K_max:
        continue
    # Make a prediction using the neighbors with the new value of K
    y_pred = np.argmax(np.bincount(neighbors[:, −1].astype(int)))
    if K == K_min or y_pred != neighbors[0, −1]:
        break
    # Return the predicted class label
    return y_pred
```

4. Experiments

In this section, we compare the performance and effectiveness of the recommendations generated by the proposed model with the baselines. We provide details of the datasets, settings, and experimental results in the following subsections.

4.1. Datasets

We used the MovieLens (https://grouplens.org/datasets/movielens/ (accessed on 20 April 2023)) datasets [27], consisting of MovieLens-100 K and MovieLens-1 M, for our experiments, since they are considered the standard datasets for assessing recommendation methods. The ratings in the MovieLens dataset range on a 5-star system, with one being the most "unliked" and five denoting highly "liked". The MovieLens-100K dataset includes 100,000 ratings from 943 users on 1682 movies, while the MovieLens-1M dataset has 1 million ratings from 6040 users on 3900 movies. In addition, we incorporated a ratings-based characteristic to decide whether to recommend the movie, to simplify the categorization process. The movie is considered "recommended" if the rating ranges from 3 to 5, while it is regarded as "not recommended" when it has a rating of 1 or 2. The two datasets were divided into two sets, with 80% assigned to training and the remaining 20% for testing for each experiment. The statistics of the two datasets are shown in Table 1.

Table 1. Statistics of the MovieLens datasets (100 K and 1 M).

	MovieLens-100 k	**MovieLens-1 M**
Number of Users	943	6040
Number of Items	1682	3900
Ratings	100,000	1,000,209
Sparsity	93.70%	99.75%
Rating Range	1–5	1–5
Average Rating	3.53	3.61

4.2. Benchmarks

We compare the proposed collaborative filtering method, ExtKNNCF, to different variations of the KNN-based collaborative filtering and co-clustering collaborative filtering methods, as follows:

- KNN-Basic: a basic KNN-based collaborative filtering algorithm that utilizes distance measurements between samples and other data points in a dataset to predict ratings. It identifies the *K*-nearest neighbors and utilizes majority voting to make rating predictions;
- KNN-w-Baseline: a basic KNN-based collaborative filtering algorithm that takes into account a *baseline* rating [39] to discover the functional connections between an input and output for rating prediction;
- KNN-w-Means: a basic KNN-based collaborative filtering algorithm that takes into account the mean ratings of each user. It computes the mean values for both item and user ratings and uses them to predict ratings;
- Co-Clustering: a collaborative filtering algorithm based on Co-Clustering [40]. Co-Clustering is a process that can efficiently handle high-dimensional and sparse data by simultaneously clustering the columns and rows of a matrix. Unlike traditional clustering, co-clustering seeks to identify blocks (or clusters) of rows and columns that are correlated and exhibit similar performance on a particular subset of columns, or vice versa.

The details of all methods are shown in Table 2.

Table 2. Details of the baseline and proposed methods used for the experiments.

Method	**Algorithm Type**	**Rating Prediction**	**Advance Features**
KNN-Basic	Memory-based	Weighted Average	
KNN-w-Baseline	Memory-based	Baseline Estimate	
KNN-w-Means	Memory-based	Mean-centered	
Co-Clustering	Model-based	Baseline Estimate	Cluster Analysis
ExtKNNCF	Model-based	SVD-based	Top-*N* Recommendations

To evaluate the performance of the recommendation system, we employed four commonly-used evaluation metrics: mean absolute error (MAE) and root mean squared error (RMSE) to measure the accuracy of rating predictions [41], and mean average precision (MAP) and normalized discounted cumulative gain (NDCG) to assess the relevance and ranking of recommended items [42]. MAE and RMSE can range from 0 to infinity, with lower values indicating a better performance. MAP and NDCG can range from 0 to 1, with higher values indicating a better performance. The details of the metrics are as follows:

MAE measures the average absolute difference between predicted and actual ratings for a set of items. It is defined as follows:

$$MAE = \frac{1}{n} \sum_{u,i} |r_{ui} - \hat{r}_{ui}| \quad (5)$$

where n is the total number of ratings, r_{ui} is the actual rating given by user u to item i, and $\hat{r}_{ui}$ is the predicted rating for item i by the recommendation system.

RMSE takes into account the squared differences between predicted and actual ratings. The formula for RMSE is as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{u,i}(r_{ui} - \hat{r}_{ui})^2} \tag{6}$$

MAP measures the mean of average precision for all users, where precision is the fraction of relevant items among the recommended ones. It is calculated as follows:

$$MAP@N = \frac{1}{|U|}\sum_{u=1}^{|U|}\frac{1}{\min(N, n_u)}\sum_{i=1}^{N}\text{Precision}(u,i)\cdot rel(u,i) \tag{7}$$

where U is the set of users, N is the number of recommended items per user, $@N$ represents the top-N recommended items, n_u is the number of relevant items for user u, $\text{Precision}(u,i)$ is the precision at position i for user u, and $rel(u,i)$ is an indicator function that is 1 if item i is relevant for user u and 0 otherwise.

NDCG is a ranking quality metric that considers the relevance and position of recommended items. It reduces the relevance of items at higher positions and normalizes the result using the ideal ranking. It is calculated as follows:

$$NDCG@N = \frac{DCG@N}{IDCG@N} \tag{8}$$

where N is the number of recommendations displayed to the user, $@N$ represents the top-N recommended items, $DCG@N = \sum_{i=1}^{N}\frac{(2^{rel_i}-1)}{\log_2(i+1)}$ is the discounted cumulative gain at position N, rel_i is the relevance score of the ith recommended item, i is the position of the item in the ranking, and $IDCG@N$ is the ideal discounted cumulative gain at position N, which is computed by sorting the items by their relevance scores and computing the DCG of the sorted list.

4.3. Experimental Results and Discussions

Our experiments aimed to assess the effectiveness of the proposed ExtKNNCF model for recommending movies, as compared to various benchmarks. To achieve this, we conducted experiments on the MovieLens dataset and used several evaluation metrics, including MAE, RSME, MAP, and NDCG. By comparing the performance of our proposed model with the benchmarks using these metrics, we aimed to demonstrate the effectiveness and superiority of our approach. The number of recommended items N was set to 25. We first ran the experiment on the MovieLens-100k dataset, to evaluate the prediction performance of the proposed method compared with the baselines. Then, we deployed the same experiment with the other dataset, the MovieLens-1M dataset, since more benchmark datasets would to help validate the effectiveness of the proposed model. Table 3 presents the experimental results of the five methods based on the MAE, RMSE, MAP, and NDCG metrics. Lower MAE and RMSE values indicate better performance of rating prediction accuracy, whereas higher MAP and NDCG values suggest better recommendations. As shown in Table 3, the proposed method, ExtKNNCF, outperformed all other methods, in terms of MAE and RMSE. Co-clustering also performed well, with the second-lowest MAE and RMSE values. Model-based methods, such as co-clustering and ExtKNNCF, tended to perform better than memory-based methods, such as KNN-Basic, KNN-w-Baseline, and KNN-w-Means, as the latter had higher MAE and RMSE values. Regarding the MAP and NDCG in Table 3, ExtKNNCF demonstrated superior performance on both datasets, MovieLens-100K and MovieLens-1M, with the highest values among all the methods. Co-clustering also performed well, with the second-highest values for both metrics. The memory-based methods had inferior performance for both metrics compared to the model-

based methods. For the first experiment on the MovieLens-100K dataset, the proposed method was better by approximately 14%, 15%, 8%, and 11% compared with the second-highest method, co-clustering, in terms of MAE, RMSE, MAP, and NCDG, respectively, while on the MovieLens-1M dataset, the proposed method outperformed the others by 12%, 19%, 6%, and 9% for the MAE, RMSE, MAP, and NCDG metrics, respectively.

Table 3. Experimental results in terms of the MAE, RMSE, MAP, and NDCG evaluation metrics on the MovieLen datasets (100 K and 1 M).

	MovieLens-100 K				**MovieLens-1 M**			
	MAE	**RMSE**	**MAP**	**NDCG**	**MAE**	**RMSE**	**MAP**	**NDCG**
KNN-Basic	0.856	1.087	0.107	0.136	0.803	1.009	0.127	0.149
KNN-w-Baseline	0.843	1.077	0.112	0.143	0.794	0.974	0.142	0.161
KNN-w-Means	0.831	1.063	0.115	0.149	0.783	0.953	0.166	0.173
Co-Clustering	0.824	1.052	0.121	0.156	0.776	0.872	0.170	0.189
ExtKNNCF	0.810	1.037	0.129	0.167	0.764	0.853	0.164	0.198

We conducted an additional experiment by varying the number of recommendations, which aimed to compare the proposed method recommendation prediction accuracy with other algorithms using various similarity measures such as the Jaccard similarity measure (RJaccard), Jaccard mean squared difference Measure (RJMSD) [43,44], Jaccard similarity measure using ratings (Rating-Jaccard) [45], and maximum probabilistic intuitionistic preference-based recommender system (MPIP) [46]. The number of recommendations was set as a set $\{5, 10, 20, 30, 50, 80\}$ for this experiment. We used 5-fold cross-validation during the testing phase with MAE and RMSE metrics. Based on the experimental results presented in Figure 2, our approach outperformed all other algorithms for each number of recommendations in the set $N = \{5, 10, 20, 30, 50, 80\}$. However, it is worth noting that the prediction accuracy decreased when the number of recommendations increased.

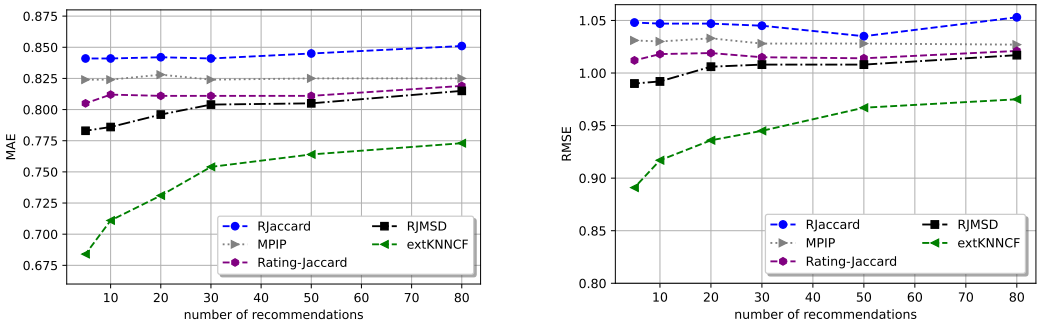

Figure 2. Comparison between the proposed method with RJaccard, RJMSD, Rating-Jaccard, and MPIP according to the top-N recommendations prediction accuracy on MovieLens-100 K.

The cold start problem in this study refers to the challenge of making predictions for new users with limited or no historical data. In the context of KNN, this can involve finding similar instances based on available features and using their labels or classes to estimate the label for the new instance. These solutions are standard in the existing methods mentioned in the baselines section. The proposed ExtKNNCF model relies on collaborative filtering and uses a different approach that involves labeling the cluster of similar users based on their cognitive similarity and using this label to make predictions for new users. Hence, our experiments aimed to demonstrate how the proposed method could outperform the baseline methods in reducing the cold start problem. As shown in Table 3, ExtKNNCF outperformed the other models across all assessment criteria, including MAE, RMSE, MAP, and NDCG, indicating its superior efficacy in recommending items for new users or those with scarce information. Moreover, a further experiment aimed to validate the accuracy improvement in generating top-N recommendations as compared to other methods. The

proposed method demonstrated better performance for all metrics. Therefore, by incorporating cognitive similarity with user preferences, ExtKNNCF generates more tailored and personalized recommendations. This makes it effective in providing recommendations, even when faced with limited user or item information.

5. Conclusions

In this paper, we proposed the ExtKNNCF, an adaptive KNN-based collaborative filtering model that incorporates user cognition parameters. The user cognition parameters allowed us to capture user preferences more accurately, thus enhancing the quality of the recommendations. Through experiments conducted on the MovieLens dataset, including MovieLens-100 K and MovieLens-1 M, we demonstrated that our proposed model effectively generated top-N recommendations. Furthermore, we compared our proposed method with other benchmarks, such as the KNN-Basic, KNN-w-Baseline, KNN-w-Means, and co-clustering methods, as well as the RJaccard, RJMSD, Rating-Jaccard, and MPIP methods, and found that the proposed method outperformed them for different evaluation metrics. Our plans for future work involve expanding our recommendation system to function as a cross-domain recommendation model that can deliver better performance across multiple application domains. We also plan to analyze the execution time performance of the proposed method, to verify its optimization. Moreover, we aim to deploy experiments with more different benchmark datasets, such as FilmTrust and IMDB, to have better validation results.

Author Contributions: Conceptualization, L.V.N. and T.-H.N.; methodology, L.V.N. and T.-H.N.; software, L.V.N. and Q.-T.V.; validation, T.-H.N.; formal analysis, L.V.N. and T.-H.N.; investigation, T.-H.N.; resources, L.V.N.; data curation, Q.-T.V.; writing—original draft preparation, L.V.N.; writing—review and editing, Q.-T.V. and T.-H.N; visualization, L.V.N.; supervision, T.-H.N.; project administration, L.V.N. and T.-H.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study: https://grouplens.org/datasets/movielens/ (accessed on 20 April 2023).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nguyen, L.V.; Jung, J.J.; Hwang, M. OurPlaces: Cross-Cultural Crowdsourcing Platform for Location Recommendation Services. *ISPRS Int. J. Geo-Inf.* **2020**, *9*, 711. [CrossRef]
2. Nguyen, L.V.; Nguyen, T.H.; Jung, J.J.; Camacho, D. Extending collaborative filtering recommendation using word embedding: A hybrid approach. In *Concurrency and Computation: Practice and Experience*; Wiley: New York, NY, USA, 2021; p. e6232. [CrossRef]
3. Nguyen, L.V.; Hong, M.S.; Jung, J.J.; Sohn, B.S. Cognitive Similarity-Based Collaborative Filtering Recommendation System. *Appl. Sci.* **2020**, *10*, 4183. [CrossRef]
4. Sabet, A.J.; Shekari, M.; Guan, C.; Rossi, M.; Schreiber, F.; Tanca, L. THOR: A Hybrid Recommender System for the Personalized Travel Experience. *Big Data Cogn. Comput.* **2022**, *6*, 131. [CrossRef]
5. Nguyen, L.V.; Nguyen, T.H.; Jung, J.J. Content-Based Collaborative Filtering using Word Embedding. In Proceedings of the International Conference on Research in Adaptive and Convergent Systems, ACM, Gwangju, Republic of Korea, 13–16 October 2020; pp. 96–100. [CrossRef]
6. Nguyen, L.V.; Jung, J.J. Crowdsourcing Platform for Collecting Cognitive Feedbacks from Users: A Case Study on Movie Recommender System. In *Proceedings of the Springer Series in Reliability Engineering*; Springer International Publishing: New York, NY, USA, 2020; pp. 139–150. [CrossRef]
7. Isinkaye, F.O.; Folajimi, Y.O.; Ojokoh, B.A. Recommendation systems: Principles, methods and evaluation. *Egypt. Inform. J.* **2015**, *16*, 261–273. [CrossRef]
8. Lara-Cabrera, R.; González-Prieto, Á.; Ortega, F. Deep Matrix Factorization Approach for Collaborative Filtering Recommender Systems. *Appl. Sci.* **2020**, *10*, 4926. [CrossRef]

9. Kumar, P.; Thakur, R.S. Recommendation system techniques and related issues: A survey. *Int. J. Inf. Technol.* **2018**, *10*, 495–501. [CrossRef]
10. de Campos, L.M.; Fernández-Luna, J.M.; Huete, J.F.; Rueda-Morales, M.A. Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian networks. *Int. J. Approx. Reason.* **2010**, *51*, 785–799. [CrossRef]
11. Diez, J.; Delcoz, J.; Luaces, O.; Bahamonde, A. Clustering people according to their preference criteria. *Expert Syst. Appl.* **2008**, *34*, 1274–1284. [CrossRef]
12. Barragáns-Martínez, A.B.; Costa-Montenegro, E.; Burguillo, J.C.; Rey-López, M.; Mikic-Fonte, F.A.; Peleteiro, A. A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition. *Inf. Sci.* **2010**, *180*, 4290–4311. [CrossRef]
13. Zheng, W.; Zhao, L.; Zou, C. Locally nearest neighbor classifiers for pattern classification. *Pattern Recognit.* **2004**, *37*, 1307–1309. [CrossRef]
14. Gao, Q.B.; Wang, Z.Z. Center-based nearest neighbor classifier. *Pattern Recognit.* **2007**, *40*, 346–349. [CrossRef]
15. Cevikalp, H.; Triggs, B.; Polikar, R. Nearest hyperdisk methods for high-dimensional classification. In Proceedings of the 25th international conference on Machine learning—ICML 08, Helsinki, Finland, 5–9 July 2008; ACM Press: New York, NY, USA, 2008; pp. 120–127. [CrossRef]
16. Hernández-Rodríguez, S.; Martínez-Trinidad, J.F.; Carrasco-Ochoa, J.A. Fast k most similar neighbor classifier for mixed data (tree k-MSN). *Pattern Recognit.* **2010**, *43*, 873–886. [CrossRef]
17. Zhou, Z.H.; Yu, Y. Ensembling Local Learners Through Multimodal Perturbation. *IEEE Trans. Syst. Man Cybern. Part B* **2005**, *35*, 725–735. [CrossRef] [PubMed]
18. Domeniconi, C.; Yan, B. Nearest neighbor ensemble. In Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, 26 August 2004; IEEE: Piscataway, NJ, USA, 2004; Volume 1, pp. 228–231. [CrossRef]
19. Altınçay, H. Ensembling evidential k-nearest neighbor classifiers through multi-modal perturbation. *Appl. Soft Comput.* **2007**, *7*, 1072–1083. [CrossRef]
20. Yang, J.M.; Yu, P.T.; Kuo, B.C. A Nonparametric Feature Extraction and Its Application to Nearest Neighbor Classification for Hyperspectral Image Data. *IEEE Trans. Geosci. Remote Sens.* **2010**, *48*, 1279–1293. [CrossRef]
21. Subramaniyaswamy, V.; Logesh, R. Adaptive KNN based Recommender System through Mining of User Preferences. *Wirel. Pers. Commun.* **2017**, *97*, 2229–2247. [CrossRef]
22. Zhang, C.; Yu, L.; Wang, Y.; Shah, C.; Zhang, X. Collaborative User Network Embedding for Social Recommender Systems. In Proceedings of the 2017 SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, Houston, TX, USA, 27–29 April 2017; pp. 381–389. [CrossRef]
23. Feng, L.; Zhao, Q.; Zhou, C. Improving performances of Top-N recommendations with co-clustering method. *Expert Syst. Appl.* **2020**, *143*, 113078. [CrossRef]
24. Walek, B.; Fojtik, V. A hybrid recommender system for recommending relevant movies using an expert system. *Expert Syst. Appl.* **2020**, *158*, 113452. [CrossRef]
25. Bathla, G.; Aggarwal, H.; Rani, R. AutoTrustRec: Recommender System with Social Trust and Deep Learning using AutoEncoder. *Multimed. Tools Appl.* **2020**, *79*, 20845–20860. [CrossRef]
26. Alhijawi, B.; Al-Naymat, G.; Obeid, N.; Awajan, A. Novel predictive model to improve the accuracy of collaborative filtering recommender systems. *Inf. Syst.* **2021**, *96*, 101670. [CrossRef]
27. Harper, F.M.; Konstan, J.A. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.* **2015**, *5*, 1–19. [CrossRef]
28. Mican, D.; Sitar-Taut, D.A. The effect of perceived usefulness of recommender systems and information sources on purchase intention. *Kybernetes* 2023 , *in press*. [CrossRef]
29. Wang, R.; Wu, Z.; Lou, J.; Jiang, Y. Attention-based dynamic user modeling and Deep Collaborative filtering recommendation. *Expert Syst. Appl.* **2022**, *188*, 116036. [CrossRef]
30. Sitar-Tăut, D.A.; Mican, D. MRS OZ: Managerial recommender system for electronic commerce based on Onicescu method and Zipf's law. *Inf. Technol. Manag.* **2019**, *21*, 131–143. [CrossRef]
31. Sitar-Tăut, D.A.; Mican, D.; Buchmann, R.A. A knowledge-driven digital nudging approach to recommender systems built on a modified Onicescu method. *Expert Syst. Appl.* **2021**, *181*, 115170. [CrossRef]
32. Koren, Y. Collaborative filtering with temporal dynamics. *Commun. ACM* **2010**, *53*, 89–97. [CrossRef]
33. Huang, Z.; Zeng, D.; Chen, H. A Comparison of Collaborative-Filtering Recommendation Algorithms for E-commerce. *IEEE Intell. Syst.* **2007**, *22*, 68–78. [CrossRef]
34. Zahir, A.; Yuan, Y.; Moniz, K. AgreeRelTrust—A Simple Implicit Trust Inference Model for Memory-Based Collaborative Filtering Recommendation Systems. *Electronics* **2019**, *8*, 427. [CrossRef]
35. Ni, J.; Cai, Y.; Tang, G.; Xie, Y. Collaborative Filtering Recommendation Algorithm Based on TF-IDF and User Characteristics. *Appl. Sci.* **2021**, *11*, 9554. [CrossRef]
36. Guo, F.; Lu, Q. Contextual Collaborative Filtering Recommendation Model Integrated with Drift Characteristics of User Interest. *Hum. Cent. Comput. Inf. Sci.* **2021**, *11*, 1–18. [CrossRef]
37. Widiyaningtyas, T.; Ardiansyah, M.I.; Adji, T.B. Recommendation Algorithm Using SVD and Weight Point Rank (SVD-WPR). *Big Data Cogn. Comput.* **2022**, *6*, 121. [CrossRef]

38. Hasan, M.; Roy, F. An Item-Item Collaborative Filtering Recommender System Using Trust and Genre to Address the Cold-Start Problem. *Big Data Cogn. Comput.* **2019**, *3*, 39. [CrossRef]
39. Koren, Y. Factor in the neighbors. *ACM Trans. Knowl. Discov. Data* **2010**, *4*, 1–24. [CrossRef]
40. George, T.; Merugu, S. A Scalable Collaborative Filtering Framework Based on Co-Clustering. In Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM 05), Houston, TX, USA, 27–30 November 2005; IEEE: Piscataway, NJ, USA, 2005. [CrossRef]
41. Shani, G.; Gunawardana, A. Evaluating Recommendation Systems. In *Recommender Systems Handbook*; Springer: Boston, MA, USA, 2010 , pp. 257–297. [CrossRef]
42. Radlinski, F.; Craswell, N. Comparing the sensitivity of information retrieval metrics. In Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Geneva, Switzerland, 19–23 July 2010; ACM: New York, NY, USA, 2010; pp. 667–674. [CrossRef]
43. Bag, S.; Kumar, S.K.; Tiwari, M.K. An efficient recommendation generation using relevant Jaccard similarity. *Inf. Sci.* **2019**, *483*, 53–64. [CrossRef]
44. Jain, G.; Mahara, T.; Sharma, S.; Sangaiah, A.K. A Cognitive Similarity-Based Measure to Enhance the Performance of Collaborative Filtering-Based Recommendation System. *IEEE Trans. Comput. Soc. Syst.* **2022**, *9*, 1785–1793. [CrossRef]
45. Ayub, M.; Ghazanfar, M.A.; Khan, T.; Saleem, A. An Effective Model for Jaccard Coefficient to Increase the Performance of Collaborative Filtering. *Arab. J. Sci. Eng.* **2020**, *45*, 9997–10017. [CrossRef]
46. Manochandar, S.; Punniyamoorthy, M. A new user similarity measure in a new prediction model for collaborative filtering. *Appl. Intell.* **2020**, *51*, 586–615. [CrossRef]