Journal of Autonomous Intelligence (2023) Volume 6 Issue 3
doi: 10.32629/jai.v6i3.723

ORIGINAL RESEARCH ARTICLE

The use of restricted Boltzmann machines for clustering collaborative filtering

Soojung Lee

Department of Computer Education, Gyeongin National University of Education, Anyang 13910, Republic of Korea; sjlee@gin.ac.kr

ABSTRACT

Collaborative filtering-based recommender systems have been successfully serviced through commercial online systems to assist people with searching the information useful to them. However, several problems inherent in such systems still exist, although a lot of research work has been devoted to finding solutions. This work focuses on clustering collaborative filtering to address the scalability problem. It proposes a novel method to determine the clustering criteria for enhancing the prediction and recommendation accuracy of the systems, which is typically degraded when the clustering algorithm is integrated into collaborative filtering. We use a restricted Boltzmann machine to find the genre preference of users, which is then inputted into the clustering algorithm to cluster users. Various experiments are conducted to evaluate the performance of the proposed method. As a result, our method showed superior performance in terms of various performance criteria compared to previous clustering collaborative filtering methods and some of the major traditional systems.

Keywords: collaborative filtering; recommender system; restricted Boltzmann machine; clustering algorithm; K-means clustering

ARTICLE INFO

Received: 14 June 2023
Accepted: 31 July 2023
Available online: 30 September 2023

1. Introduction

In the flood of information, people tend to spend a lot of time searching for products they want. The recommendation algorithm, which emerged as a major method to solve these difficulties, is essentially used in online commercial systems. There are various types of products recommended by the recommender system, and almost all the online market products such as books, music, travel, clothing, and movies are examples. Successful recommender systems include Amazon, YouTube, and eBay[1–3].

There are many ways to implement a recommender system, such as content-based filtering, collaborative filtering, trust-based filtering, social network-based filtering, and hybrid filtering[4,5]. Among them, the collaborative filtering (CF) technique is regarded as the most widely and practically used and studied in the literature. Its basic principle is that it determines a recommendation list by referring to the item preference history of the users similar to the current user. It has the great advantage of avoiding the need to keep information such as user profiles or item attributes, which are usually difficult to collect in reality, but it only maintains the item preference history of users. Preference information is obtained either explicitly from the user ratings data or implicitly from product purchases, viewing time,

or click-throughs[6].

This paper focuses on collaborative filtering, for its advantages and wide and successful usage in practical recommender systems. It is usually classified into memory-based and model-based, where the former is simple to implement, known to be efficient, and easily adapted to the data environment[4–6]. It maintains and stores ratings for items given by users and calculates items to recommend from this data. However, since the time required to obtain users similar to the current user is proportional to the amount of stored rating information, the cost of finding similar users is high, in case there are too many users of the system. This is called the scalability problem, which is chronic and inherent in CF systems[2,4]. Conversely, if there are very few system users, it is difficult to obtain preference items of similar users. This data sparsity problem is another main drawback of CF along with cold-start, gray-sheep, and scalability[6].

Various data mining or machine learning algorithms are used to address these shortcomings of CF systems. This approach, named the model-based CF technique, attempts to learn a model from user ratings. Some of the popular models include matrix factorization, support vector regression, Bayesian networks, latent semantic models, and clustering[4,6]. However, there are disadvantages such as the cost of building a model and the need for continuous learning to keep it up to date.

This paper addresses clustering-based CF. It is recognized as a useful tool to reduce the critical scalability problem of CF while providing recommendation quality comparable to traditional CF[7]. Clustering CF is one of the very useful techniques because it does not cause huge matrix computation or information loss required by singular value decomposition (SVD) or principal component analysis (PCA).

Due to the recent development of deep learning, research is being conducted to develop CF-based recommender systems using neural networks that deviate from the existing memory-based or model-based methods[5,8–11]. In this study, neural networks are used for a different purpose than before. Specifically, we use a Restricted Boltzmann Machine, a type of neural network, to find the genre preference of users, which is then inputted into the clustering algorithm to cluster users. This approach can be considered unique and original, as previous clustering-based CF systems usually cluster users based on their rating values, instead of the genre preference information. Therefore, not only the rating value itself but also its context can be identified by integrating its genre into the proposed method. Since a movie usually belongs to several genres, this can also help overcome data sparsity. Various experiments were conducted to evaluate the performance of the proposed method using a public dataset including genre information. As a result, our method showed superior performance in terms of various performance criteria compared to the existing CF methods with clustering based on user ratings. Furthermore, it significantly outperforms some of the major CF systems with traditional similarity measures.

The remainder of this paper is organized as follows: In the next section, we discuss previous works related to this study. Section 3 describes the proposed method, followed by performance experiments in Section 4. Section 5 concludes this paper.

2. Related work

2.1. Clustering collaborative filtering

In the traditional CF system, to decide whether to recommend an item for an active user, the system consults the ratings of his neighbors for the item to produce a predicted rating. An active user or a target user refers to the user who is currently accessing the system and waiting for a recommendation service. Neighbor users consist of those who have shown similar rating behavior to items in the past. This process may require significant computational load due to similarity calculation[12] between the users, degrading the online performance of the system. Clustering CF is an effective way to reduce this problem.

Through clustering of users, similar users belong to one cluster, and other users belong to another cluster. By forming clusters, the neighbor set is defined only within a cluster. Since the clustering CF algorithm refers only to other similar users in the same cluster for rating prediction and the size of a cluster is usually much smaller than the total number of users, it can solve the scalability problem of the CF system.

K-means, as a representative clustering algorithm, has been used in several studies to improve the performance of CF systems. Xue et al. suggested a smoothing-based method based on clusters, which combines the advantages of both memory-based and model-based CF[13]. The study of Gong[14] presented an approach that joins the user clustering and item clustering techniques together to solve the scalability and sparsity problems in CF. A recommender system for the online shopping market is proposed by Kim and Ahn[15] that applied genetic algorithms to K-means clustering. Nilashi et al. presented a method for obtaining rating values of items using regression techniques for each cluster using K-means in a multi-criteria user rating environment[16]. Recent work by Liu et al.[17] suggests a hybrid news recommendation algorithm, which combines the content-based recommendation algorithm and collaborative filtering, using the term frequency-inverse document frequency (TF-IDF) method and K-means clustering, to extract the news features and also applies SVD technology to solve the matrix sparse problem.

Self-organizing map (SOM) is widely known as another clustering algorithm, which performs clustering through neural network-based unsupervised learning. In Ye's study, ratings of unrated items are filled by association rule mining and then estimated by SOM clustering[18]. Some studies suggested a combination of SOM and K-means as an ensemble method and showed better recommendation performance than a single clustering method[19]. In the study of Lee et al.[20], SOM is combined with CF for a recommender system, where all users are segmented by demographic characteristics, and users in each segment are clustered using the SOM network. Roh et al. proposed a model consisting of profiling, inferring, and predicting steps, which combines a CF algorithm with SOM and case-based reasoning[21]. Purbey et al. presented a detailed description and performance evaluation of the SOM algorithm used for CF[22].

Although K-means and SOM are used in various ways, they have a disadvantage in that it is difficult to determine the initial number of clusters. However, K-means has the advantage of being concise and easy to implement but has the disadvantage that its performance is greatly influenced by the initial center values of clusters and can fall into local optima[6]. On the other hand, in SOM, it is difficult to determine initial weights and termination conditions. To improve these disadvantages, Liao and Lee proposed a method of automatically configuring clusters[7]. In addition, Fremal and Lecron performed clustering using item metadata information[23], and Najafabadi et al. clustered song data by mining indirect interaction information of users and association rules[24].

Recently, context-aware recommender systems have drawn much attention from researchers, but integration of restricted Boltzmann machines (RBM) with them is seldom considered. Pu and Zhang proposed a time-based user clustering collaborative filtering recommendation algorithm, where all users and their interest changes are considered and the user rating time is added to increase the similarity between users when users are clustered. In addition, rating time is added to reduce the search space of similar users in the cluster[25].

2.2. Recommender system using RBM

Deep learning is a technology for discovering hidden characteristics in data, used for feature extraction or dimensionality reduction. It has recently been actively applied in the field of recommender systems[10,11,26]. An RBM is used to extract features inherent in user ratings of items in recommender systems[27,28]. RBM is reported to improve the accuracy of the recommender system by modeling the correlation between user-rated items or the relationship between users[29]. RBM is also used to model group profiles and group characteristics, helping to develop a recommender system through group preferences[30].

Salakhutdinov et al. presented an RBM model with only two layers for the progress of the learning and inference process using user ratings and showed that the RBM can be successfully applied to the Netflix dataset[28]. Sahoo et al. developed an intelligent health recommender system using RBM-CNN (Convolutional Neural Network), which showed that big data analysis plays a very important role in decision-making from the patient's health point of view[31]. Verma et al. add supervision by exploiting user demographic information and item metadata to the RBM for collaborative filtering, where their RBM formulation improves significantly on the existing RBM-based approach and yields result at par with the state-of-the-art latent factor-based models[32]. Yang and Lu addressed the current problem of the RBM model for collaborative filtering in the literature that it only deals with categorized ratings and explicit feedback, and proposed an extension of the RBM method with implicit feedback. Their model is said to directly predict preferences for a new user given feedback on a few items[33].

3. Proposed approach

3.1. Methodology

The proposed method assumes that the system maintains the ratings given by the users to the items. In addition, the genre information of the items is also stored, where an item may belong to one or more genres. In the case of a movie recommendation system, genres may be classified into action, drama, mystery, and the like. In the case of the music domain, the genres include the classics, new-age, pop songs, etc. If the users' exact rating values are not maintained by the system, they can be inferred from indirect evaluation information such as the number of clicks, the time spent on the homepage, and so on, which is another research field[6] and is outside the scope of this paper.

The suggested method addresses the scalability problem of the recommender system. The most popular strategy to tackle this problem is to integrate the clustering algorithm into the system. However, despite reducing the scalability problem to some extent, this strategy is known to introduce a problem of degrading recommendation performance accuracy[14,23]. Therefore, deciding what to set for the criterion for clustering is an important factor to help solve this problem. For this, we utilize the RBM to select the best criterion.

Figure 1 depicts the overall process of the suggested method. It consists of three main steps, extraction of genre rating frequency of users, the clustering algorithm execution, and producing the recommendation for an active user. The first step is the output of the RBM which serves as a criterion for clustering of users. Once clustering is complete, the system finds the cluster to which an active user belongs and produces recommendation results by consulting the ratings of the nearest neighbors in the same cluster of the user. Details of the procedure are described in the next section.

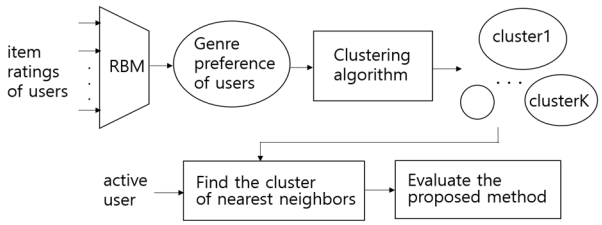

Figure 1. System flow of the proposed method.

3.2. Restricted Boltzmann machine

Our dataset is assumed to have discrete ratings from one to five. Specifically, user ID, item ID, and rating information are available in the dataset. Datasets such as MovieLens or FilmTrust, which are widely used in academia, provide such information[4]. This study uses the MovieLens dataset which provides genre information of items in addition.

Figure 2 presents the RBM of this study. Like the model in the study of Salakhutdinov et al.[28], our model has a binary value for each hidden unit, and the visible unit is composed of softmax units. Each visible unit corresponds to an item. Therefore, the total number of visible units is equal to the total number of items provided by the system. Users are trained through separate RBMs but share weights and biases.

The RBM in **Figure 2** shows the visible layer representing the ratings of a user, where the user rating for item one ($item_1$) is three, thus only the box corresponding to 3 in the figure (the black square in the figure) has an initial value of 1, and the rest has an initial value of 0. It is shown that this user has not rated $item_2$. Originally, the optimal number of hidden units in an RBM can be obtained through learning, but in this study, it was set to be the same as the number of movie genres provided by the MovieLens dataset. Therefore, since the probability of each hidden unit having the value of one can be obtained as a learning result of the RBM, it is possible to grasp the genre preference of each user.

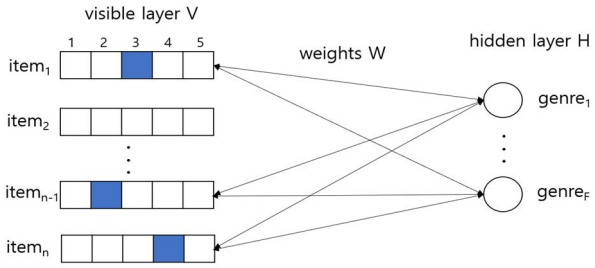

Figure 2. The proposed RBM model.

As proposed in the study of Salakhutdinov et al.[28], RBM learning is approximated through the CD (Contrastive Divergence) algorithm, in which alternating Gibbs sampling is done repetitively. We adopt this approach for learning and details of the algorithm can be referred to the study of Salakhutdinov et al.[28]. **Table 1** lists up notations for our RBM, where the parameter values were determined based on the work in the study of Hilton[34] and used in the experiments of this study. The probability that the hidden unit g has the binary value one is calculated as follows, where σ is a sigmoid function.

$$prob(h_g = 1|V) = \sigma\left(b_g + \sum_i v_i w_{ig}\right) \quad (1)$$

This probability for each hidden unit is interpreted as the degree of user preference for the corresponding genre. After obtaining the probabilities for all users through RBM learning, users are clustered according to the genre preferences, i.e., probabilities, as shown in the system flow in **Figure 1**. Details of our clustering algorithm follow.

Table 1. Notations used by our restricted Boltzmann machine.

Parameter	Description	Value
N	Number of epochs	200
v_i	Visible unit i	-
h_g	Hidden unit g	-
b_g	Bias for hidden unit g	Random number in (−0.1, +0.1)
w_{ig}	Weight connecting visible unit i and hidden unit g	$N(0, 0.01^2)$
F	Number of hidden units	18
η	Learning rate	0.01–0.05
α	Momentum	0.1–0.5

3.3. Clustering

In this paper, we employ the K-means algorithm to cluster users. The basic idea of the K-means algorithm is to have users within a cluster show higher similarity than those belonging to different clusters. To measure the degree of similarity between users, this study uses the genre preference of users, which is in contrast to most existing studies that use user ratings[12,13,24,25]. Hence in our approach, users with similar genre preferences are clustered together.

Algorithm 1 shows the description of our proposed clustering algorithm. In Step 1, each cluster center is determined as a random user. Specifically, the center is defined as the genre preference of the user, which is obtained from Equation (1) for each genre. Step 2 is iterated until there is no change of any cluster center or the pre-set number of iterations is reached. This step consists of two main processes. The first is to determine the cluster to which each user belongs, and the second is to newly calculate the center of each cluster. In step 2.2, we decide to measure the similarity between users and cluster centers using Pearson correlation. Let us denote the preference for genre g of user u and of the cluster center c as $f_{u,g}$ and $f_{c,g}$, respectively. Then similarity between them is computed as

$$COR_{gr}(u,c) = \frac{\sum_g (f_{u,g} - \bar{f}_u)(f_{c,g} - \bar{f}_c)}{\sqrt{\sum_g (f_{u,g} - \bar{f}_u)^2}\sqrt{\sum_g (f_{c,g} - \bar{f}_c)^2}}, \quad g \in G_{u,c} \tag{2}$$

where $G_{u,c}$ is the set of common genres of u and c. The average genre preferences of u are computed as

$$\bar{f}_u = \frac{1}{|G_{u,c}|}\sum_{g \in G_{u,c}} f_{u,g} \tag{3}$$

After user clustering through Steps 2.1~2.3 is completed, there may be a change in the set of users constituting each cluster. Hence, a new center should be obtained from the genre preference values of users belonging to the cluster. That is, given the set of genres G and the set of users U in cluster C, the center c of cluster C is $\{f_{c,g}, \mathrm{g} \in \mathrm{G} | f_{c,g} = \frac{1}{|U|}\sum_u f_{u,g}\}$, where u belongs to C.

Algorithm 1 The proposed clustering algorithm

```
Input parameters:
    K: number of clusters
    N: threshold for the number of iterations
begin
1. For each cluster Ci, select a user randomly as its center.
2. while (iter ≤ N and any center of a cluster is changed) do
2.1     for each user u do
2.2         Compute similarity between u and every cluster center.
2.3         Let u belong to the cluster with the highest similarity.
2.4     for each cluster do
2.5         Compute its new center.
2.6     iter = iter + 1
3. end while
end
```

4. Performance analysis

4.1. Experimental background

4.1.1. Dataset

The dataset used to measure the performance of the proposed method should provide user ratings in integer values. Moreover, it is desirable to have a small rating range so that the network learning time is as low as possible. As one of the public datasets popularly used in the related research, we select MovieLens 1M (http://www.grouplens.org). This dataset consists of 6040 users, 3952 movies, and 802,553 user ratings.

MovieLens consists of a total of 18 genres, and each movie item provided by the system belongs to one or more genres. In this study, the ratio of the training data set and the test data set was set at a typical ratio of 8:2. Description of the dataset is shown in **Table 2**.

Table 2. Dataset description.

Feature	Value
Number of users	6040
Number of items	3952
Number of ratings	802,553
Rating range	1–5 (integer)
Mean number of ratings per user	132.87
Sparsity level	0.96638
Recommendation threshold	4
Number of genres	18

4.1.2. Experimented methods

The performance of a CF system usually depends on how accurately the system predicts the ratings of unrated items for an active user. Typical memory-based CF methods usually select the most similar users (nearest neighbors) of the active user and integrate the ratings of these neighbors to produce the predicted ratings. Hence, the similarity measure of the system is very important for system performance, on which several related research works are presented[6,12].

We selected major traditional similarity measures most popularly used in this CF research field for performance comparison. They are Pearson correlation (COR) which is reported to show the best performance among the traditional measures[6], the cosine similarity (COS), and the mean squared differences (MSD). Also, to relatively evaluate the efficiency of our clustering criteria, we executed a clustering-based CF algorithm with K-means (KM as a legend). K-means clusters users based on user ratings instead of user genre preference as in our strategy. Moreover, we made a performance comparison with CF algorithms adopting well-known clustering methods, SOM[20,35] and fuzzy C-means (FCM)[36] which are reported to perform competitively[37]. The proposed method uses PROP as a legend.

4.1.3. Performance metrics

There are several aspects developed to estimate the performance of the CF systems such as prediction accuracy, recommendation accuracy, rank accuracy, novelty, diversity, etc.[1,6,12]. In this experiment, we adopt two most commonly used aspects, prediction accuracy and recommendation accuracy. For the first, we used Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), which are defined as follows.

$$MAE = \frac{1}{n}\sum |\hat{r}_{u,i} - r_{u,i}| \tag{4}$$

$$RMSE = \sqrt{\frac{1}{n}\sum (\hat{r}_{u,i} - r_{u,i})^2} \tag{5}$$

where $r_{u,i}$ is the real rating of item i given by user u and $\hat{r}_{u,i}$ is its prediction by the system.

Precision and recall are two representative metrics measuring recommendation accuracy. A recommendation list is constructed by the system with those items having predicted ratings over the recommendation threshold. In our experiment, considering the rating range of the dataset, we set the recommendation threshold at four as in some previous works using the same dataset[38,39]. Precision refers to

the ratio of the number of items a user actually prefers among the list recommended by the system. Recall is the ratio of how many items are actually included in the recommendation list out of the total items that the user prefers.

4.2. Results

4.2.1. Determining the number of clusters

The clustering algorithm, a component of the proposed method in this study, is known to be greatly affected by the number of clusters[6,17]. Hence, we conducted experiments of PROP with the varying number of clusters K. **Figure 3** pictures MAE performance for the varying number of nearest neighbors and with different clusters. It is roughly figured out that MAE tends to decrease with the number of clusters. This is obviously because there will be neighbors with better accuracy when the cluster size becomes larger. However, it is noted that reducing the number of clusters smaller than three does not help in terms of MAE.

We also investigated F1 performance with varying K and showed the results in **Figure 3**. F1 is a recommendation quality metric merging precision and recall with equal weight. The results are shown for a different number of recommended items. That is, the figure depicts how much the user would satisfy as more items are recommended. It is seen that F1 generally increases with the increasing number of items, although it experiences a little decrease after the number of items exceeds 60 for all the methods. From these results, we set the number of clusters to three in the ensuing experiments, taking both the cluster size for scalability and performance into consideration.

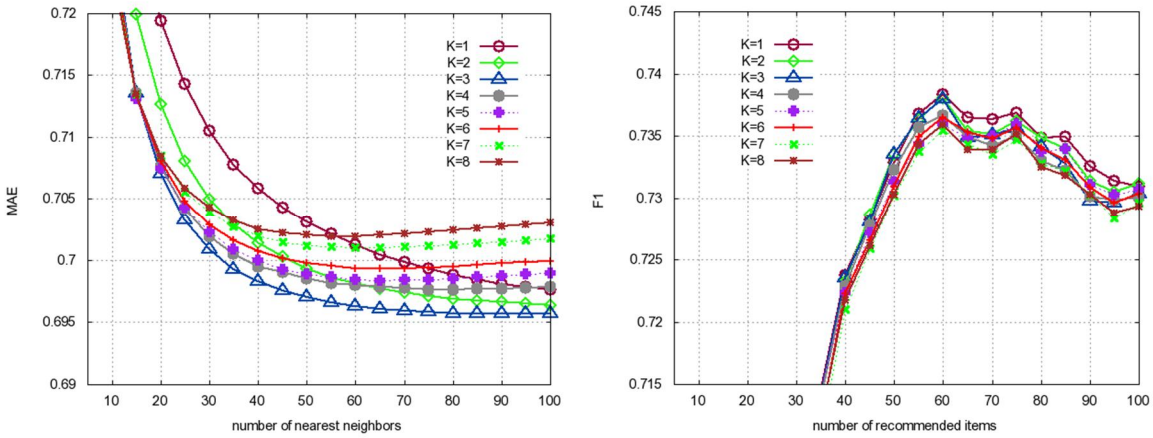

Figure 3. Mean absolute error (MAE) and F1 of PROP with the varying number of clusters (K).

4.2.2. Performance comparison

Figure 4 depicts prediction accuracy results in terms of MAE and RMSE. According to the definitions, the lower the values, the better the performance. We used the same scale range for both results for precise comparison. Note that the results become stabilized with the number of nearest neighbors for all the methods, implying that there seems no need to obtain more and more neighbors to achieve better prediction accuracy once the result reaches some threshold.

The MAE and RMSE results seem almost analogous in that the performance ranking is the same for both but the gaps between the results make a difference. That is, the best-performed method in both MAE and RMSE turns out to be PROP in general followed by FCM, COR, KM, SOM, and MSD, in that order, while COS performs worst, for a sufficient number of nearest neighbors consulted. However, it is noted that in terms of RMSE, KM and SOM perform relatively worse, than in terms of MAE, when compared with all the other methods. Observe that the gap between the RMSE results of KM/SOM and those of COR tends to be larger with the increasing number of neighbors, although they use COR for computing similarity. Hence, it is discovered that the cases of a large difference between the real and predicted ratings are more often

found using KM/SOM than other methods, as RMSE amplifies the contributions of the errors, according to the definition. Also, it is obviously because of the well-known fact that clustering-based CF algorithms suffer from performance degradation due to clustering. FCM is seen to be less affected since it adopts the fuzzy concept in its algorithm. Notably, PROP seldom exhibits such drawback of clustering CF and performs almost the best even with clustering.

Figure 5 shows the F1 results of the methods with the same scale range as **Figure 4** for easy comparison. As more items are recommended, F1 seems generally better, fluctuating for some duration, and then becomes a bit worse for still more items. Nevertheless, it is noted that COR performs best, followed by PROP. In this result, PROP again outperforms FCM and KM as in prediction accuracy, although the gap is not that large. This proves that our strategy of utilizing user genre preference as clustering criteria is superior and even defeats the fuzzy concept incorporated into clustering. It is also observed that SOM is even outperformed by COS and MSD, showing that CF performance adopting clustering can be worse than the traditional CF algorithms.

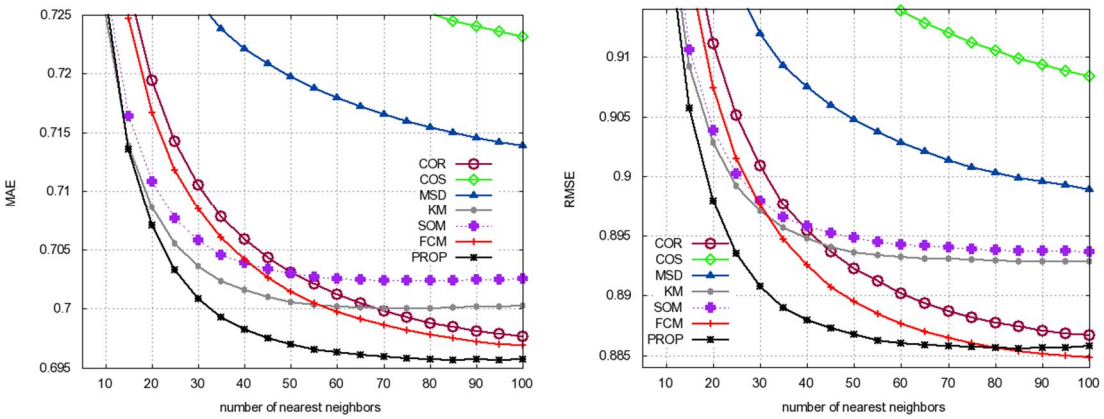

Figure 4. Prediction accuracy of the experimented methods with the varying number of nearest neighbors.

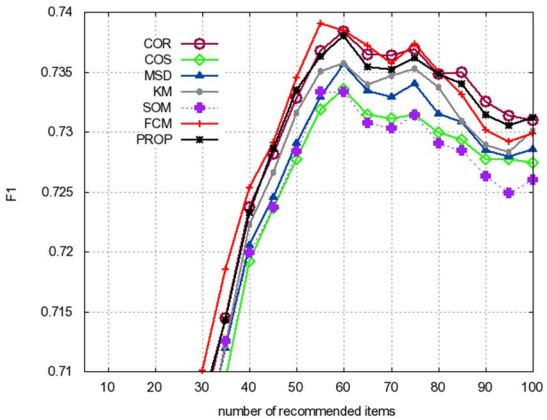

Figure 5. F1 results of the experimented methods with the varying number of recommended items.

Since F1 is a comprehensive measure of precision and recall, each performance result was examined to more closely observe where the performance difference occurs. **Tables 3** and **4** show the results. It is seen that almost ignorable differences occur between the results of the methods in both metrics. As in F1, COR, FCM, and PROP are in general the best performing in precision and recall, while SOM and COS perform worst overall. Consequently, from all the experimental results, it is concluded that our clustering method using the outcome of the RBM is proved efficient compared to the user rating-based clustering method of KM and that it even outperforms other clustering CF algorithms such as FCM and SOM.

Table 3. Precision with the varying number of recommended items.

#Recommended items	COR	COS	MSD	KM	SOM	FCM	PROP
10	0.7928	0.7866	0.7924	0.7954	0.7901	0.7981	0.7957
20	0.7639	0.7585	0.7611	0.7633	0.7613	0.7675	0.7666
30	0.7457	0.7410	0.7428	0.7428	0.7401	0.7458	0.7470
40	0.7238	0.7192	0.7205	0.7222	0.7175	0.7233	0.7237
50	0.7055	0.7004	0.7015	0.7045	0.6983	0.7045	0.7060
60	0.6883	0.6837	0.6856	0.6860	0.6824	0.6875	0.6880
70	0.6683	0.6634	0.6651	0.6663	0.6613	0.6666	0.6672
80	0.6529	0.6485	0.6499	0.6524	0.6481	0.6539	0.6529
90	0.6396	0.6357	0.6361	0.6363	0.6322	0.6361	0.6387
100	0.6278	0.6248	0.6256	0.6285	0.6230	0.6267	0.6288

Table 4. Recall with the varying number of recommended items.

#Recommended items	COR	COS	MSD	KM	SOM	FCM	PROP
10	0.4904	0.4859	0.4900	0.4914	0.5027	0.5076	0.4916
20	0.6071	0.6025	0.6048	0.6065	0.6086	0.6126	0.6080
30	0.6704	0.6661	0.6683	0.6682	0.6721	0.6776	0.6717
40	0.7235	0.7191	0.7204	0.7221	0.7223	0.7274	0.7228
50	0.7623	0.7571	0.7586	0.7606	0.7612	0.7672	0.7633
60	0.7961	0.7911	0.7935	0.7931	0.7925	0.7976	0.7959
70	0.8197	0.8142	0.8160	0.8185	0.8153	0.8207	0.8186
80	0.8401	0.8345	0.8364	0.8380	0.8331	0.8392	0.8402
90	0.8569	0.8508	0.8520	0.8529	0.8534	0.8569	0.8555
100	0.8744	0.8702	0.8718	0.8706	0.8697	0.8737	0.8734

5. Conclusion

This study proposed a novel CF algorithm with clustering. It addressed a major problem of scalability in CF systems and attempts to enhance the recommendation accuracy that can be degraded by clustering. The proposed method has three stages, RBM learning to extract the genre rating frequency of users, clustering algorithm execution, and producing recommendations. This approach is quite different from previous clustering-based CF systems which usually cluster users based on their rating values, instead of the genre preference information as ours. Various experiments were conducted to evaluate the performance of the proposed method. As a result, our method showed superior performance in terms of various performance criteria compared to existing clustering CF methods as well as some of the major traditional CF systems.

In this study, only a simple restricted Boltzmann machine with two layers was used, but future research on learning through the construction of an in-depth model is promising for better performance. Also, only the genre preference of users is extracted through the network, but other useful user features may be further obtained through more extensive learning, which can be inputted into the clustering algorithm. As another future work, it is desirable to verify the performance of the proposed method in various aspects by conducting additional experiments using datasets with different characteristics.

Conflict of interest

The author declares no conflict of interest.

References

1. Aamir M, Bhusry M. Recommendation system: State of the art approach. *International Journal Computer Applications* 2015; 120(12): pp. 25–32. doi: 10.5120/21281-4200
2. Chen R, Hua Q, Chang YS, et al. A survey of collaborative filtering-based recommender systems: from traditional methods to hybrid methods based on social networks. *IEEE Access* 2018; 6: 64301–64320. doi: 10.1109/ACCESS.2018.2877208
3. Jaiswal S, Jaiswal T. Survey on recommender system using deep learning networks. *Artificial Intelligence Evolution* 2020; 72–89. doi: 10.37256/aie.122020435
4. Jalili M, Ahmadian S, Izadi M, et al. Evaluating collaborative filtering recommender algorithms: A survey. *IEEE Access* 2018; 6: 74003–74024. doi: 10.1109/ACCESS.2018.2883742
5. Jothilakshmi SL, Bharathi R. Survey on collaborative filtering technique for recommender system using deep learning. In: Kannan RJ, Thampi SM, Wang SH (editors). *Computer Vision and Machine Intelligence Paradigms for SDGs*. Springer; 2023.
6. Su X, Khoshgoftaar TM. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* 2009; 2009: 421425. doi: 10.1155/2009/421425
7. Liao CL, Lee SJ. A clustering based approach to improving the efficiency of collaborative filtering recommendation. *Electronic Commerce Research and Applications* 2016; 18: 1–9. doi: 10.1016/j.elerap.2016.05.001
8. Batmaz Z, Yurekli A, Bilge A, Kaleli C. A review on deep learning for recommender systems: Challenges and remedies. *Artificial Intelligence Review* 2019; 52(1): 1–37. doi: 10.1007/s10462-018-9654-y
9. Mu R, Zeng X, Han L. A survey of recommender systems based on deep learning. *IEEE Access* 2018; 6: 69009–69022. doi: 10.1109/ACCESS.2018.2880197
10. Wang H, Wang N, Yeung DY. Collaborative deep learning for recommender systems. *arXiv* 2015; arXiv:1409.2944. doi: 10.48550/arXiv.1409.2944
11. Zhang S, Yao L, Sun A, Tay Y. Deep learning based recommender system: a survey and new perspectives. *ACM Computing Surveys* 2019; 52(1): 1–38. doi: 10.1145/3285029
12. Khojamli H, Razmara J. Survey of similarity functions on neighborhood-based collaborative filtering. *Expert Systems with Applications* 2021; 185: 115482. doi: 10.1016/j.eswa.2021.115482
13. Xue G, Lin C, Yang Q, et al. Scalable collaborative filtering using cluster-based smoothing. In: Proceedings of the 28th annual international ACM SIGIR conference on Research and Development in Information Retrieval; 15 August 2005; Salvador, Brazil. pp. 114–121.
14. Gong S. A collaborative filtering recommendation algorithm based on user clustering and item clustering. *Journal of Software* 2010; 5(7): 745–752. doi: 10.4304/jsw.5.7.745-752
15. Kim KJ, Ahn H. A recommender system using GA K-means clustering in an online shopping market. *Expert Systems with Applications* 2008; 34(2): 1200–1209. doi: 10.1016/j.eswa.2006.12.025
16. Nilashi M, Jannach D, Ibrahim O, Ithnin N. Clustering- and regression-based multi-criteria collaborative filtering with incremental updates. *Information Sciences* 2015; 293: 235–250. doi: 10.1016/j.ins.2014.09.012
17. Liu J, Song J, Li C, et al. A hybrid news recommendation algorithm based on K-means clustering and collaborative filtering. *Journal of Physics: Conference Series* 2021; 1881: 032050. doi: 10.1088/1742-6596/1881/3/032050
18. Ye H. A personalized collaborative filtering recommendation using association rules mining and self-organizing map. *Journal of Software* 2011; 6(4): 732–739. doi: 10.4304/JSW.6.4.732-739
19. Tsai CF, Hung C. Cluster ensembles in collaborative filtering recommendation. *Applied Soft Computing* 2012; 12(4): 1417–1425. doi: 10.1016/j.asoc.2011.11.016
20. Lee M, Choi P, Woo Y. A hybrid recommender system combining collaborative filtering with neural network. In: De Bra P, Brusilovsky P, Conejo R (editors). *Adaptive Hypermedia and Adaptive Web-Based Systems*. Springer; 2002. Volume 2347. pp. 531–534.
21. Roh TH, Oh KJ, Han I. The collaborative filtering recommendation based on SOM cluster-indexing CBR. *Expert Systems with Applications* 2003; 25(3): 413–423. doi: 10.1016/S0957-4174(03)00067-8
22. Purbey N, Pawde K, Gangan S, Karani R. Using self-organizing maps for recommender systems. *International Journal of Soft Computing and Engineering* 2014; 4(5): 47–50.
23. Fremal S, Lecron F. Weighting strategies for a recommender system using item clustering based on genres. *Expert Systems With Applications* 2017; 77(1): 105–113. doi: 10.1016/j.eswa.2017.01.031
24. Najafabadi MK, Mahrin MN, Chuprat S, Sarkan HM. Improving the accuracy of collaborative filtering recommendations using clustering and association rules mining on implicit data. *Computers in Human Behavior* 2017; 67: 113–128. doi: 10.1016/j.chb.2016.11.010
25. Pu X, Zhang B. Clustering collaborative filtering recommendation algorithm of users based on time factor. In:

Proceedings of the 2020 Chinese Control And Decision Conference (CCDC); 22–24 August 2020; Hefei, China. pp. 364–368.
26. Zheng Y, Tang B, Ding W, Zhou H. A neural autoregressive approach to collaborative filtering. *arXiv* 2016; arXiv:1605.09477. doi: 10.48550/arXiv.1605.09477
27. Deng S, Huang L, Xu G, et al. On deep learning for trust-aware recommendations in social networks. *IEEE Transactions on Neural Networks and Learning Systems* 2017; 28(5): 1164–1177. doi: 10.1109/TNNLS.2016.2514368
28. Salakhutdinov R, Mnih A, Hinton G. Restricted Boltzmann machines for collaborative filtering. In: Proceedings of the 24th International Conference on Machine Learning; 20 June 2007; New York, United States. pp. 791–798.
29. Georgiev K, Nakov P. A non-iid framework for collaborative filtering with restricted Boltzmann machines. *The 30th International Conference on Machine Learning* 2013; 28(3): 1148–1156. doi: 10.5555/3042817.3043065
30. Hu L, Cao J, Xu G, et al. Deep modeling of group preferences for group-based recommendation. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence; 27 July 2014; Canada. pp. 1861–1867.
31. Sahoo AK, Pradhan C, Barik RK, Dubey H. DeepReco: deep learning based health recommender system using collaborative filtering. *Computation* 2019; 7(2): 1–25. doi: 10.3390/computation7020025
32. Verma S, Patel P, Majumdar A. Collaborative filtering with label consistent restricted Boltzmann machine. *arXiv* 2019; arXiv:1910.07724. doi: 10.48550/arXiv.1910.07724
33. Yang F, Lu Y. Restricted Boltzmann machines for recommender systems with implicit feedback. In: Proceedings of the 2018 IEEE International Conference on Big Data (Big Data); 10–13 December 2018; Seattle, WA, USA. pp. 4109–4113.
34. Hinton GE. A practical guide to training restricted Boltzmann machines. In: Montavon G, Orr GB, Müller KR (editors). *Neural Networks: Tricks of the Trade.* Springer; 2012. pp. 599–619.
35. Sarne GML. A collaborative filtering recommender exploiting a SOM network. In: Bassis S, Esposito A, Morabito FC (editors). *Recent Advances of Neural Network Models and Applications*. Springer; 2014. pp. 215–222.
36. Koohi H, Kiani K. User based collaborative filtering using fuzzy c-means. *Measurement* 2016; 91: 134–139. doi: 10.1016/j.measurement.2016.05.058
37. Lee S. Fuzzy clustering with optimization for collaborative filtering-based recommender systems. *Journal of Ambient Intelligence and Humanized Computing* 2022; 13(9): 4189–4206. doi: 10.1007/s12652-021-03552-8
38. Bobadilla J, Hernando A, Ortega F, Gutierrez G. Collaborative filtering based on significances. *Information Sciences* 2012; 185(1): 1–17. doi: 10.1016/j.ins.2011.09.014
39. Iftikhar A, Ghazanfar MA, Ayub M, et al. An improved product recommendation method for collaborative filtering. *IEEE Access* 2020; 8: 123841–123857. doi: 10.1109/ACCESS.2020.3005953