Fast and accurate evaluation of collaborative filtering recommendation algorithms

Nikolaos Polatidis[1], Stelios Kapetanakis[1], Elias Pimenidis[2] and Yannis Manolopoulos[3]

[1]School of Architecture, Technology and Engineering, University of Brighton, BN2 4GJ, Brighton, United Kingdom
[2]Department of Computer Science and Creative Technologies, University of the West of England, BS16 1QY, Bristol, United Kingdom
[3]Faculty of Pure and Applied Sciences, Open University of Cyprus, 2220, Nicosia, Cyprus
{N.Polatidis,S.Kapetanakis@Brighton.ac.ukElias.Pimenidis@uwe.ac.uk,Yannis.Manolopoulos@Ouc.ac.cy}

Abstract. Collaborative filtering are recommender systems algorithms that provide personalized recommendations to users in various online environments such as movies, music, books, jokes and others. There are many such recommendation algorithms and, regarding experimental evaluations to find which algorithm performs better a lengthy process needs to take place and the time required depends on the size of the dataset and the evaluation metrics used. In this paper we present a novel method that is based on a series of steps that include random subset selections, ensemble learning and the use of well-known evaluation metrics Mean Absolute Error and Precision to identify, in a fast and accurate way, which algorithm performs the best for a given dataset. The proposed method has been experimentally evaluated using two publicly available datasets with the experimental results showing that the time required for the evaluation is significantly reduced, while the results are accurate when compared to a full evaluation cycle.

Keywords: Recommender Systems, Collaborative Filtering, Evaluation, Mean Absolute Error, Precision.

1 Introduction

Recommender systems are algorithms that are based on opinions of a community of users to provide personalized recommendations of items, such as movies, books, jokes and music among others to users [1, 2]. One of the most successful technologies to provide such recommendations to users is Collaborative Filtering (CF), an approach that is based on a history of common ratings between users. When common ratings exist between users then, in its basic approach, a distance is calculated between users to form a neighborhood of common users using distance metrics such as the Pearson Correlation Coefficient (PCC), the Cosine or the Jaccard similarities [1-3]. However, numerous CF algorithms have been developed in the past few years that improve the quality of the recommendations in one way or another, usually for a specific domain.

The challenge in this area comes in clearly identifying which is the best algorithm. This can take place using an online approach where users are requested to evaluate a system with the click rate being counted, or in an offline approach where evaluation metrics are being used [4]. In the later approach, which is the most common amongst researchers, there should be a suitable volume of data and the selected evaluation metrics should be appropriate for the task. Based on the volume of the data several experimental evaluation rounds need to take place to identify which algorithm is the best [1, 4]. Evaluating recommender systems using offline metrics is usually based on accuracy metrics such as the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE), or on information retrieval metrics such as Precision, Recall, and F1 among others [4]. The drawback of this approach is that depending on (a) how many algorithms need to be tested, (b) the volume of the data, (c) how many metrics will be used, and (d) how many tests are considered enough, the process could prove very time-consuming. It might take from a few hours to several days to execute all the tests, collect the results and go through a manual comparison to conclude which algorithm performs best. To this extent we have developed a randomization-based method that is both practical and effective in recommending a ranked list of collaborative filtering algorithms in a time efficient way. The contributions of the paper are:

- A method for fast evaluation of collaborative filtering algorithms, based on random subset data selection is delivered.
- The proposed method has been evaluated using two publicly available datasets and well-known evaluation metrics.

The rest of the paper is structured as follows: Section 2 contains the related work, Section 3 delivers the proposed method, Section 4 presents the experimental evaluation and Section 5 is the conclusions.

2 Related work

This section is divided into two parts. The first part presents several CF algorithms found in the literature and the second part explains the offline evaluation approaches available in the domain. In the literature there are numerous CF approaches and various evaluation metrics which usually results in a time-consuming procedure.

2.1 Collaborative filtering algorithms

The most traditional and widely used CF algorithms are the ones that form a neighborhood of similar users based on a history of common ratings using a distance metrics such as the PCC or Cosine similarity [1]. The method is usually referred as K nearest neighbors (KNN) where K is the number of neighbors assigned to each user. This traditional method has been extended by Polatidis and Georgiadis [2] where the similarity is divided into multiple levels according to the number of co-rated items and the actual similarity value provided by PCC, while the accuracy and the precision is improved. This work was extended in Polatidis and Georgiadis [5] to create the multiple levels dynamically using information such as the number of users, items, and ratings. The previous dynamic multi-level work has been extended by Shojaei and Saneifar [3] where the authors introduced a fuzzy model which outperforms the dynamic approach that it is based on.

Another work is the one from Anand and Bharadwaj [6] where the authors have utilized sparsity measures based on local and global similarities to improve recommendation quality. Other works include the one from Bobadilla et al., [7] where the authors proposed a new metric that can be used as an alternative to PCC or Cosine that considers both common and uncommon ratings. This method has been validated and it was shown that recommendation quality is improved in terms of accuracy and precision/recall. Another metric from Bobadilla et al., [8] uses singularities to improve the quality of the recommendations.

There are other methods in the literature that aim to improve recommendations in different ways. RF-Rec is such a method that is based on rating frequencies to provide fast and accurate recommendations, while it outperforms the traditional baselines [9]. The method from Liu et al., [10] analyses the disadvantages of traditional recommendation methods such as PCC and Cosine and proposes a metric that is based on Proximity, Impact and Popularity (PIP) which improves the provided recommendation list. Najafabadi et al., [11] proposed a similarity metric that is based on clustering and association rule mining to improve the accuracy.

CF algorithms work in different ways such as using ontologies and dimensionality reduction techniques as proposed by Nilashi et al., [12]. Sarwar et al., [13] introduced the concept of Incremental Singular Value Decomposition (SVD) algorithms that makes recommender systems scalable. HU-FCF is a hybrid user-based fuzzy collaborative filtering method that uses fuzzy logic to improve the accuracy [14]. A work of further interest is the one from Wang et al., [15] where the authors developed a new metric that utilizes entropy to improve recommendations. A more recent work is one that uses a neural network to deliver neural CF and improved quality in the recommendations [16]. Xiaojun [17] delivered an improved collaborative filtering recommendation algorithm that is based on clustering. Dimensionality reduction and clustering have also been used in [18]. AutoSVD++ is a method where CF is delivered with the use of contractive auto-encoders which improves the quality of the recommendations [19].

2.2 Evaluation methods

Recommender systems evaluation using offline evaluation metrics is usually divided into two parts, accuracy, and retrieval.

The first part is to calculate the error of the predictions made using either the Mean Absolute Error (MAE) or the Root Mean Squared Error (RMSE) evaluation metrics [1-2, 4]. Both metrics work in a similar way: In MAE when a rating prediction is made for a user the predicted value is compared to the actual value and an error value is calculated. The smaller the error value is, the better the rating prediction algorithm.

In RMSE the difference is that the error is squared, thus larger error values are more harshly punished.

The second part uses Information Retrieval (IR). Here metrics are being used to evaluate how good a recommendation list is. Metrics such as Precision, Recall, F1, Mean Average Precision (MAP) and the Mean Reciprocal Rank (MRR) are commonly used for evaluation. This case is different to the previous one, with higher values being better. Values are in the range of 0 to 1 or 0 to 100 when converted to a % scale [1-2, 4.

During an evaluation process different metrics are typically used. Researchers perform experiments using different algorithms, they collect results, and evaluate them based on those metrics. Accuracy and IR metrics are well established in the recommender systems community and are used to evaluate algorithms in several rounds of experiments by using different settings for each. While the correct approach is to follow such an experimentation procedure the drawback here is that it is very time consuming, and the reproducibility of the results is often difficult since in published research papers settings used in algorithms and experiments are not explained in detail. The proposed method fills this gap by automating the evaluation process in a fast and accurate approach with default algorithmic and evaluation settings.

3 Proposed method

The novelty of the proposed method is in the random selection of an N number of subsets from the dataset, evaluating and combining the results using an ensemble approach. Each of the N subsets selected is independent from any other, which means that the method does not run concurrently but procedurally, and each time a new execution is running steps 1 to 6 of the method as follows: The MAE value is calculated, converted to a 1 to 100 % and then the accuracy is calculated by subtracting the error value from 100. Then the precision is calculated, and the two metrics are combined to form a new metric that gives a final output value between 0 and 1. This value is then used to rank CF algorithms and recommend a list of CF algorithms with the ones having higher values appearing higher in the list. The above steps run independently from each other for N times and at the final step shown in equation 7 an ensemble of ranking values is calculated using a soft voting ensemble approach. In the equation n is the number of recommended items, p is the predicted rating and r is the actual rating.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |p_i - r_i| \tag{1}$$

Convert MAE value to a 0 to 100 scale as follows:

$$MAE100 = \frac{100}{Max\ rating\ value} \tag{2}$$

At the next step we calculate the error in a % scale as follows:

$$Error100 = MAE * MAE100 \tag{3}$$

Following on, we calculate the accuracy of the algorithm in terms of rating prediction as follows:

$$Accuracy = 100 - Error100 \tag{4}$$

The precision is calculated using equation 5. Precision is a value from 0 to 100 which tells us how good a list of recommendations is. Good recommendations are usually provided to users that satisfy a minimum rating criterion such as 4 out of 5 and “all recommendations” are all the recommendations provided to each user.

$$Precision = \frac{Good\ recommendations}{All\ recommendations} \tag{5}$$

The next step of the method involves the combination of the Accuracy (MAE) and Precision values now that both are in a positive 0 to 100 % value as shown in equation 6.

$$Combine = 2 * \frac{Accuracy\ *\ Precision}{Accuracy\ +\ Precision} \tag{6}$$

The value obtained from equation 6 is then used to rank each collaborative filtering algorithm. The higher the value the higher in the list the algorithm appears. Once this step has finished a voting ensemble algorithm is used as shown in equation 7. At this step the algorithm that gathers the highest ensemble value appears higher in the list, since a re-ranking process takes place.

$$Ensemble = \frac{\text{Combine1} + \text{Combine2} + \ldots\ldots\ldots\ \text{Combine N}}{N} \tag{7}$$

4 Experimental evaluation

This section explains the experiments and includes subsections with the settings, datasets, algorithms, and results. We found that a subset of about 10% of users includes ratings for many of the items which results to similar outputs using MAE and Precision compared to when evaluating with the full dataset and we used three subsets. Therefore, N = 3 with approximately 10% of users with all their ratings for each dataset and the values are rounded up or down to the closest value. For example, from the Epinions dataset 4000 of 40163 have been used and for the MovieTweetings dataset 2000 of 21645 users have been selected. It is shown that in each evaluation cycle of each of the datasets, even when a subset of users is used, all data items are being used in the process. The Java programming language has been used in an Intel Dual core i7 2.5 GHz with 8 gigabytes of RAM computer running Windows 10. 5-fold cross validation has been used in all experiments.

4.1 Datasets

We used two publicly available datasets the statistical information of which is presented in table 1. The datasets were chosen based on their size and specific characteristics such as the number of users and items.

Epinions: This is a general commerce dataset with more items than users and a rating scale from 1 to 5 [20].
MovieTweetings: This is a movies dataset with the details about the users, items and ratings crawled from Twitter and it rating scale is from 1 to 10 [21].

Table 1. Dataset statistics

Dataset	Users	Items	Ratings	Scales
Epinions	40163	139738	664823	[1,5]
MovieTweetings	21645	12989	150000	[1,10]

Figures 1 shows 10 different selections of random users for the Epinions, and MovieTweetings datasets respectively. It is shown that each time a random subset of users is selected the number of ratings remains similar.

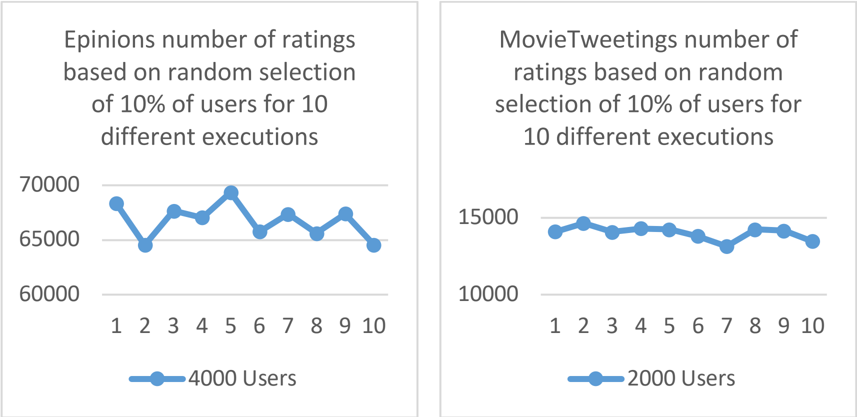

Fig 1. Number of ratings for 4000 users of the Epinions dataset and 2000 users of the MovieTweetings dataset based on 10 random selections

4.2 Algorithms

Three algorithms have been used in the experiments and are described in detail below along with their settings.

KNN: This is the traditional user-user algorithm that forms a neighborhood of similar users using PCC with minimum similarity value of 0.0 and K number of neighbors equals 50 and a minimum overlap of 3 items.

Rf-Rec: This is an algorithm that generates predictions based on the counting and combination of different rating values [9].
Funk SVD: This is an SVD based algorithm that ignores missing values in the rating matrix [22].

4.3 Metrics

Three evaluation metrics have been used in the evaluation process. MAE, Precision and MRR.

MAE: This metric calculates the difference between an actual rating and a predicted rating. It has been defined in equation 1 since it is also used in the method.
Precision: This metric calculates the quality of the recommendations. It has been defined in equation 5 since it is also used in the method.
MRR: This is a metric that can be used to calculate the ordered probability of correctness. It is defined in equation 8 below, where Q is the number of queries executed and rank of "i" is the order in which the first most relevant answer appears within a list of ranked answers.

$$MRR = \frac{1}{|Q|} \sum_{1}^{|Q|} \frac{1}{rank_i} \tag{8}$$

MAE and Precision are used to calculate the accuracy and precision of the three algorithms against each of the datasets for 5 random 5-fold executions. Moreover, the same metrics are being used for 3 random executions of the proposed method to create a ranked list of the algorithms for each dataset. At the end, MRR is used to calculate if the best algorithm is ranked within the first place of the recommended list.

4.4 Results

Tables 2, 3, 4 and 5 present the results for the Epinions dataset. Tables 2 and 3 show the results for the whole dataset using MAE and Precision respectively for top-5 recommendations. Five different executions based on 5-fold cross validation and random data selection took place. The results are consistent, showing that, for MAE, Funk SVD performs the best among the three algorithms, RF-Rec second best and KNN third. For precision results RF-Rec performs the best, followed by Funk SVD and KNN. Tables 4 and 5 present three executions with k (4000 in this case) random users on each. Funk SVD is the best for MAE followed by Rf-Rec and KNN while for precision RF-Rec is the best followed by Funk SVD and KNN.

Tables 6, 7, 8 and 9 present the results for the MovieTweetings dataset. Tables 6 and 7 are the results for the whole dataset using MAE and Precision respectively for top-5 recommendations. Five different executions based on 5-fold cross validation and random data selection took place. The results are consistent, showing that, for MAE, KNN performs the best among the three algorithms, Funk SVD the second best, and RF-Rec the third. For precision results RF-Rec performs the best, followed by Funk SVD and KNN. Tables 8 and 9 present three executions with k (2000 in this case) random users each. KNN is the best for MAE followed by Funk SVD and RF-Rec. For precision RF-Rec is the best, followed by Funk SVD, and KNN.

Table 2. MAE results for Epinions

Methods	1st execution	2nd execution	3rd execution	4th execution	5th execution
KNN (PCC)	0.906	0.907	0.907	0.905	0.909
RF-Rec	0.867	0.867	0.868	0.867	0.867
Funk SVD	0.802	0.802	0.803	0.802	0.802

Table 3. Precision results for Epinions for top-5 recommendations

Methods	1st execution	2nd execution	3rd execution	4th execution	5th execution
KNN (PCC)	0.75	0.75	0.75	0.75	0.751
RF-Rec	0.809	0.808	0.808	0.809	0.808
Funk SVD	0.8	0.8	0.799	0.8	0.799

Table 4. MAE results for Epinions with 4000 users

Methods	1st execution	2nd execution	3rd execution
KNN (PCC)	0.966	0.946	0.945
RF-Rec	0.951	0.943	0.94
Funk SVD	0.88	0.855	0.852

Table 5. Precision results for Epinions with 4000 users for top-5 recommendations

Methods	1st execution	2nd execution	3rd execution
KNN (PCC)	0.765	0.759	0.776
RF-Rec	0.802	0.803	0.807
Funk SVD	0.792	0.793	0.799

Table 6. MAE results for MovieTweetings

Methods	1st execution	2nd execution	3rd execution	4th execution	5th execution
KNN (PCC)	2.526	2.526	2.525	2.528	2.524
RF-Rec	2.613	2.613	2.613	2.613	2.613
Funk SVD	2.569	2.57	2.57	2.57	2.569

Table 7. Precision results for MovieTweetings for top-5 recommendations

Methods	1st execution	2nd execution	3rd execution	4th execution	5th execution
KNN (PCC)	0.746	0.743	0.741	0.746	0.744
RF-Rec	0.85	0.849	0.849	0.848	0.849

Funk SVD	0.82	0.82	0.819	0.819	0.82

Table 8. MAE results for MovieTweetings with 2000 users

Methods	1st execution	2nd execution	3rd execution
KNN (PCC)	2.435	2.447	2.436
RF-Rec	2.587	2.597	2.644
Funk SVD	2.536	2.588	2.615

Table 9. Precision results for MovieTweetings with 2000 users for top-5 recommendations

Methods	1st execution	2nd execution	3rd execution
KNN (PCC)	0.737	0.749	0.737
RF-Rec	0.845	0.853	0.85
Funk SVD	0.817	0.826	0.823

Table 10 presents the Mean Reciprocal Rank results for the datasets followed by the overall value. The recommendation list is the list of the algorithms recommended by the proposed method. The best result represents which is the best algorithm for the corresponding dataset, followed by the actual rank and the reciprocal rank value. Initially, by manually observing the results, one can clearly assess which algorithm is better in terms of MAE and Precision; while all executions use the full dataset and for all datasets the executions are similar to each other. Furthermore, by observing the results using a subset of each dataset, and by comparing algorithms using this approach, it can be observed which algorithm is the best. By randomly choosing subsets the values are not very similar, thus several executions are necessary to get average MAE and Precision values.

Table 10 presents the results of the best performing algorithm (best result) as derived by manually observing the results using the whole dataset while the recommendation list is generated using the proposed method. The results of table 10 are based on the Mean Reciprocal Rank and are 100% accurate. This metric calculates if the best result is in the top position of the recommendation list. Since the rank is 1 for all three the reciprocal rank is also 1 and the Mean Reciprocal Rank is calculated as shown below table 10.

Table 10. Mean Reciprocal Rank results

Dataset	Ranked list based on full executions	Best result based on the proposed method	Rank	Reciprocal rank
Epinions	Funk SVD, Rf-Rec, KNN	Funk SVD	1	1
MovieTweetings	RF-Rec, Funk SVD, KNN	RF-Rec	1	1

Mean Reciprocal Rank: (1+1) / 2 = 1 (100%)

Figure 2 presents the performance evaluation comparison results for the Epinions and MovieTweetings datasets, using MAE, Precision, and the KNN, RF-Rec and FunkSVD recommendation algorithms for top-5 recommendations. The times for the first part (Epinions dataset) of the figure are in hours and for the second (MovieTweetings dataset) in minutes while these are approximate, which means that these might slightly vary according to how many items will be retrieved through the randomization process, the settings, processing power, operating system and background processes running.

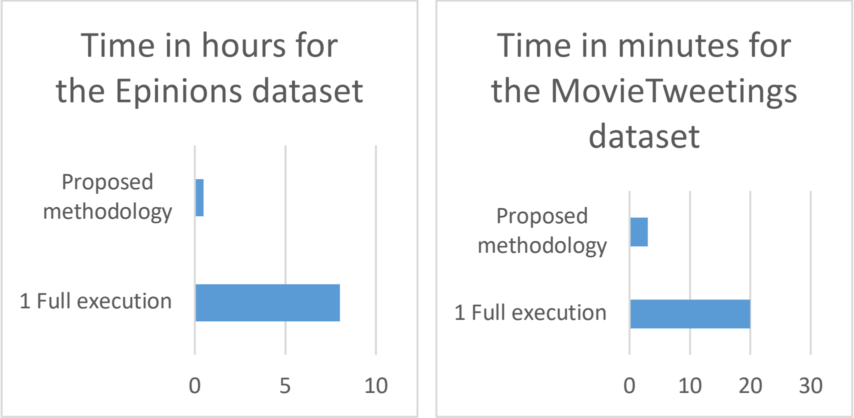

Fig 2. Performance evaluation for the Epinions dataset

The results show that random selection of user subsets and combination of their results is a good approach when evaluating recommender systems. The evaluation and ranking of the algorithms with the use of subsets of the dataset delivers an accurate list of ranked and recommended CF algorithms, while the time required to do so is significantly reduced which is especially useful for larger datasets. For Epinions, a relatively large dataset, the proposed method takes about 30 minutes of processing time while a typical execution that uses the whole dataset takes about 8 hours. For MovieTweetings, a smaller dataset, the proposed method requires about 3 minutes processing time whereas a typical round of evaluation using the whole dataset needs about 20 minutes. The proposed method can also be used as white box evaluation approach to automatically evaluate algorithms, but a limitation is the settings of the algorithms used.

5 Conclusions

Collaborative filtering has matured, and several algorithms can be found in the literature. Additionally, there are several evaluation metrics that can be used to evaluate such algorithms and are either related to accuracy or IR. CF is being used in various domains, and algorithms that are considered good in a domain might not be as good in another.

To find out which algorithm is good a manual and time-consuming procedure of running experiments needs to take place. In this article we delivered an evaluation, ranking and recommendation method which can be used to evaluate CF algorithms using subsets of the dataset in a fast and accurate way. The proposed method has been tested on two publicly available datasets, the size of which is significantly different from each other, with the results indicating that the method is fast, accurate, while it is straightforward to use

The proposed method is stable and can be used as a basis for a white box method which can automate the evaluation process and allow researchers to reproduce results without the worry of omitted settings and parameters. Thus, in the future we aim to investigate how the selection of subsets can assist in the evaluation of deep learning recommendation algorithms especially applying only IR metrics. In addition to that, we aim to deliver a white-box evaluation approach which researchers will be able to use for experimentation and will allow the experimental results to be reproduced in a straightforward way.

References

1. Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. Knowledge-based systems, 46, 109-132.
2. Polatidis, N., & Georgiadis, C. K. (2016). A multi-level collaborative filtering method that improves recommendations. Expert Systems with Applications, 48, 100-110.
3. Shojaei, M., & Saneifar, H. (2021). MFSR: A Novel Multi-Level Fuzzy Similarity Measure for Recommender Systems. Expert Systems with Applications, 114969.
4. Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1), 5-53.
5. Polatidis, N., & Georgiadis, C. K. (2017). A dynamic multi-level collaborative filtering method for improved recommendations. Computer Standards & Interfaces, 51, 14-21.
6. Anand, D., & Bharadwaj, K. K. (2011). Utilizing various sparsity measures for enhancing accuracy of collaborative recommender systems based on local and global similarities. Expert systems with applications, 38(5), 5101-5109.
7. Bobadilla, J., Serradilla, F., & Bernal, J. (2010). A new collaborative filtering metric that improves the behavior of recommender systems. Knowledge-Based Systems, 23(6), 520-528.
8. Bobadilla, J., Ortega, F., & Hernando, A. (2012). A collaborative filtering similarity measure based on singularities. Information Processing & Management, 48(2), 204-217.
9. Gedikli, F., Bagdat, F., Ge, M., & Jannach, D. (2011, September). RF-REC: Fast and accurate computation of recommendations based on rating frequencies. In 2011 IEEE 13th Conference on Commerce and Enterprise Computing (pp. 50-57). IEEE.
10. Liu, H., Hu, Z., Mian, A., Tian, H., & Zhu, X. (2014). A new user similarity model to improve the accuracy of collaborative filtering. Knowledge-Based Systems, 56, 156-166.
11. Najafabadi, M. K., Mahrin, M. N. R., Chuprat, S., & Sarkan, H. M. (2017). Improving the accuracy of collaborative filtering recommendations using clustering and association rules mining on implicit data. Computers in Human Behavior, 67, 113-128.

12. Nilashi, M., Ibrahim, O., & Bagherifard, K. (2018). A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques. Expert Systems with Applications, 92, 507-520.
13. Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2002, December). Incremental singular value decomposition algorithms for highly scalable recommender systems. In Fifth international conference on computer and information science (Vol. 1, No. 012002, pp. 27-8).
14. Son, L. H. (2014). HU-FCF: a hybrid user-based fuzzy collaborative filtering method in recommender systems. Expert Systems with Applications: An International Journal, 41(15), 6861-6870.
15. Wang, W., Zhang, G., & Lu, J. (2015). Collaborative filtering with entropy-driven user similarity in recommender systems. International Journal of Intelligent Systems, 30(8), 854-870.
16. Wang, X., He, X., Wang, M., Feng, F., & Chua, T. S. (2019, July). Neural graph collaborative filtering. In Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval (pp. 165-174).
17. Xiaojun, L. (2017). An improved clustering-based collaborative filtering recommendation algorithm. Cluster Computing, 20(2), 1281-1288.
18. Zarzour, H., Al-Sharif, Z., Al-Ayyoub, M., & Jararweh, Y. (2018, April). A new collaborative filtering recommendation algorithm based on dimensionality reduction and clustering techniques. In 2018 9th International Conference on Information and Communication Systems (ICICS) (pp. 102-106). IEEE.
19. Zhang, S., Yao, L., & Xu, X. (2017, August). AutoSVD++ An Efficient Hybrid Collaborative Filtering Model via Contractive Auto-encoders. In Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval (pp. 957-960).
20. Massa, P., Souren, K., Salvetti, M., & Tomasoni, D. (2008). Trustlet, open research on trust metrics. Scalable Computing: Practice and Experience, 9(4).
21. Dooms, S., De Pessemier, T., & Martens, L. (2013, October). Movietweetings: a movie rating dataset collected from twitter. In Workshop on Crowdsourcing and human computation for recommender systems, CrowdRec at RecSys (Vol. 2013, p. 43).
22. Funk, S. (2006). https://sifter.org/~simon/journal/20061211.html