Thesis for the Degree of Master of Engineering

CKComCF: Canopy–K-means Clustering based Combined Collaborative Filtering

by

Kuan Sao I

Department of IT Convergence and Application Engineering

The Graduate School

Pukyong National University

February 2021

CKComCF: Canopy–K-means Clustering based Combined Collaborative Filtering

캐노피-k-평균 클러스터링 기반 결합 협업 필터링

Advisor: Prof. Ha-Joo Dong

by

Kuan Sao I

A thesis submitted in partial fulfillment of the requirements

for the degree of

Master of Engineering

in Department of IT Convergence and Application Engineering,

The Graduate School,

Pukyong National University

February 2021

CKComCF: Canopy–K-means Clustering based Combined Collaborative Filtering

A dissertation

by

Kuan Sao I

Approved by:

Prof. Young-Bong Kim [illegible]

(Chairman)

Prof. Oh-Heum Kwon [illegible]

(Member)

Prof. Ha-Joo Song [illegible]

(Member)

February 19, 2021

Contents

List of Tables

List of Figures

CKComCF: Canopy–K-means Clustering based Combined Collaborative Filtering

Kuan Sao I

Department of IT Convergence and Application Engineering,

The Graduate School,

Pukyong National University

Abstract

In recent times, big data is revolutionizing every aspect of human lives. It has attracted more attention in academia and industry. To quickly obtain information, information filtering systems are essential. In the e-commerce industry, the recommendation system (RS) to predict the preferences of people has been prevalent over the few years. Collaborative filtering (CF) is one of the most conventional algorithms of RS. However, CF suffers from data sparsity and scalability issues. Thus, we propose Canopy–K-means Clustering-based Combined Collaborative Filtering (CKComCF) to solve the challenge of data sparsity and scalability. In particular, the prediction outcomes of user-based CF (UbCF) and item-based CF (IbCF) are integrated using a weighting approach, which is based on the root-mean-square error (RMSE) minimization. Experiment results based on two real-life datasets of MovieLens and Netflix Prize demonstrate that the proposed RMSE-minimization method outperforms the traditional CF methods, improving the accuracy by 64.24% (UbCF with MovieLens) and 13.72% (IbCF with Netflix Prize). The proposed CKComCF model outperforms the existing improved CF method, reducing the calculation time by 41.84% (MovieLens) and 64.77% (Netflix Prize).

1. Introduction

Big data is of interest to researchers in academia and industry because of its enormous importance in research and e-commerce. Big data involves making mathematical predictions based on huge amounts of data to infer various probabilities [1]. In this big data era, information filtering systems are widely used to retrieve information for improving operational efficiency. The emergence of big data has effectively enabled people to passively obtain all kinds of information. Recommendation system (RS) is an information retrieval and decision support tool that can automatically recommend items (e.g., music, restaurants, and books) to users by using historical records of their behavior and potential personal data [2]. RS can provide useful references to help inexperienced users select items [3].

Since the 1990s, personalized recommendation techniques have been divided into content-based [4] and collaborative filtering (CF) approaches [5]. In generally, content-based techniques include probability statistics and natural language processing to mine features and information of items; afterward, recommendations are made for similar items. CF is categorized into user-based (UbCF) and item-based (IbCF) approaches. According to the preferences and historical behaviors of similar users, UbCF predicts a list of users' favorite items. Similarly, based on the sales records of the items, IBCF predicts that similar items will be pushed to users [3]. As the information about the interaction among neighbors is merely required by CF, it has been one of the most successful and frequently used personalized recommendation techniques [2]. However, the sparsity and scalability of data are the most challenging issues of CF.

This study introduces a weighting approach that minimizes the root-mean-square error (RMSE) to enhance UbCF and IbCF based on Canopy–K-means clustering. Experiments using actual datasets (Movielens and Netflix Prize) exhibit higher accuracy compared with the traditional CF. The remainder of this study is organized as follows: Section 2 reviews traditional CF and similarity measures. Section 3 presents the mathematical formulation of the proposed model in detail. Section 4 describes the experimental results and analysis. Finally, Section 5 presents the conclusions of the study and the directions for future work.

2. Related works

In this section, first, we review the traditional CF and then present the existing approach to improve the traditional CF, e.g., compositing with pre-cluster data and combining CF with weighting.

2.1. Traditional Collaborative Filtering

The traditional CF relies on a user-item rating data matrix to calculate the similarities of users or items to find users. The accuracy of recommendations is affected by the quality of neighbors. Therefore, the key to UbCF and IbCF is the calculation of similarities among users or items.

To establish a user-item rating matrix, suppose there is a list of m users $U = \{u_1, u_2, \cdots, u_m\}$ and a set of n items $I = \{i_1, i_2, \cdots, i_n\}$ and the user-item rating matrix R as shown in (1).

$$R = \begin{bmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,n} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{m,1} & r_{m,2} & \cdots & r_{m,n} \end{bmatrix} \qquad \textbf{(1)}$$

where column vector $\vec{u} = \{r_{u,1}, r_{u,2}, \cdots, r_{u,n}\}$ denotes the ratings of user u, and row vector $\vec{i} = \{r_{1,i}, r_{2,i}, \cdots, r_{m,i}\}$ denotes the ratings of item i.

To calculate the similarities, few classic and well-known similarity measures are addressed based on CF [6], [7], e.g., Cosine similarity[8], Pearson correlation coefficient

(PCC)[9], and Jaccard[10] similarity measures. They can be measured either as a correlation or distance. Among, PCC considers the mean rating that helps to seek the neighbor more accurately [11]. Therefore, it is widely employed for CF similarity calculation. The similarity between user a and b can be calculated as provided in (2):

$$PCC(a,b) = \frac{\sum_{c \in C_{a,b}} (r_{a,c} - \bar{r}_a)(r_{b,c} - \bar{r}_b)}{\sqrt{\sum_{c \in C_{a,b}} (r_{a,c} - \bar{r}_a)^2} \sqrt{\sum_{c \in C_{a,b}} (r_{b,c} - \bar{r}_b)^2}} \qquad \textbf{(2)}$$

where $r_{a,c}$ denotes the rating of user a for item c. $C_{a,b}$ is the set of items that is rated by both user a and b. $\overline{r_a}$ denotes the mean rating of user a. PCC can be also applied to measure the similarities between item and item.

To predict the ratings, the traditional CF has involved the k-NN algorithm to obtain an ordered nearest neighbor set of target users or target items; after that, it recommends favorite items or mutual users of k numbers that are most similar neighbors from the target user or target item. UbCF can predict the approximate value of the unrated item i of the target user; the function is displayed in (3):

$$p_{u,i}^{U} = \overline{r_u} + \frac{\sum_{v \in NN(u)} PCC(u,v) \times (r_{v,i} - \overline{r_v})}{\sum_{v \in NN(u)} |PCC(u,v)|} \qquad \textbf{(3)}$$

where $p_{u,i}^{U}$ denotes the predicted rating of user u for item i. $NN(u)$ is a set of nearest neighbor users of user u, who are the most Top-N similar users based on their ratings. $r_{v,i}$ denotes the actual rating of neighbor user v for item i. $\overline{r_v}$ denotes mean rating for all items that user v has rated.

However, as the number of products or users on the site exponentially increases, the variability of items will affect the similarity index and reduce the accuracy of recommendations [12]. Compared with UbCF, IbCF can significantly improve the scalability of CF and provide better quality [13]–[15], the function is displayed in (4):

$$p_{u,i}^{I} = \overline{r_i} + \frac{\sum_{j \in NN(i)} PCC(i,j) \times \left(r_{u,j} - \overline{r_j}\right)}{\sum_{j \in NN(i)} |PCC(i,j)|} \qquad \textbf{(4)}$$

where $p_{u,i}^{I}$ denotes the predicted rating of item i from user u. $NN(i)$ is a set of nearest neighbor items of item i. $r_{u,j}$ denotes the actual rating of neighbor item j for user u. r_j denotes the actual mean rating of user user u.

2.2. Improved Collaborative Filtering

Researchers proposed various methods to construct RS to address the shortcomings of CF [16]. Owing to the huge matrix, the similarity and computational complexity of finding N nearest neighbors are considerably high. The K-means algorithm applied to the CF can effectively improve the calculation speed [15], [17]. K-Means algorithm is a classic clustering algorithm [18]. It clusters similar users or at high convergence speed. While calculating the similarity among sample data in the cluster, the initial division must be provided; therefore, it is a challenge to find the optimal number of clusters in the training process. The Canopy clustering approach can easily solve this problem [19]–[21]. First, apply the Canopy clustering algorithm for initial user-item clusters; then, the number of canopies is employed to initial K-means.

On the other hand, UbCF and IbCF have different benefits. For example, when the timeliness is high, the items change more frequently against the interest of the users, and the performance of UbCF is better. When the number of users is much larger than the number of items and the frequency of changing items is not high, the items are relatively stable than the users, and the performance of IbCF is better. A method combining UbCF and IbCF has been proposed [22]. This method first predicts the results of the user-item rating matrix using UbCF and IbCF separately. Then, it combines the predicted results to obtain the optimized prediction using a weighted-average approach based on mean absolute percentage error (MAPE). The prediction results prove the usefulness of the fusion method because of the benefits of combining UbCF and IbCF. However, when the dataset is huge, this algorithm encounters high scalability pressure.

3. Proposed Model

The effectiveness of CF makes it usable in several cases. The benefits of combining the predicted results are more satisfactory, but calculating UbCF and IbCF separately will increase the computational load and increase the scalability and sparsity shortcomings [22]. At the same time, dynamic weight is the key to obtain better prediction results. Few researchers, pointed out that Canopy–K-means can effectively speed up the calculation time [19]–[21]. Therefore, we propose CKComCF model to enhance the performance of the transitional CF, reduce the data sparsity load, and improve the prediction accuracy.

3.1. ComCF weighting

For target users and items that have not yet been evaluated, combined with the ratings of neighboring users and neighboring items, the prediction results can be dynamically predicted using uncertain weights. The function is displayed in (5):

$$p_{u,i} = \omega_1 \times p_{u,i}^{U} + \omega_2 \times p_{u,i}^{I} \qquad \textbf{(5)}$$

where $p_{u,i}$ predicts the rating of user u for item i. It is the weighted and merged rating value of two predicted ratings, UbCF rating $p_{u,i}^{U}$ and IbCF rating $p_{u,i}^{I}$. The weights, ω_1 and ω_2, are calculated in (6) and (7).

3.1.1 Comparison of CF PCC and library PCC

Similarity calculation has always been an important part of CF, and it has the ability to affect the results of the entire recommendation. In the process of calculating the PCC of CF and the PCC of other general libraries, there are calculation differences. Moreover, this tiny loophole will be ignored accidentally and get very different results. When Spark uses the PCC function, the missing values of the rating matrix will be filled with 0, so that the average of the PCC calculation is incorrect. When using Pandas and excel, the missing values of the rating matrix will ignore the entire column vector, which will also cause calculation errors. The key factor is the common ratings of the column vector, which is not cared about in these usual libraries, so it is necessary for the PCC of CF to be fine-tuned or established artificially. The appendix A 1 shows the difference between the PCC calculation required by CF and the calculation of PCC similarity using pandas, spark libraries and excel.

3.2. RMSE-minimization for ComCF

RMSE is employed as the performance measure for the prediction generation of UbCF and IbCF. RMSE measures the average absolute deviation between the real and predicted ratings. Based on the error propagations, the weight between UbCF and IbCF can be measured dynamically, thereby improving the system quality (6) and (7):

$$\omega_1 = 1 - \frac{RMSE(R^U)}{RMSE(R^U) + RMSE(R^I)} \quad \textbf{(6)}$$

$$\omega_2 = 1 - \frac{RMSE(R^I)}{RMSE(R^U) + RMSE(R^I)} \quad \textbf{(7)}$$

where ω_1 and ω_2 denote the weighted average as RMSE-minimization. R^U and R^I denote the sets of ratings for users and items, respectively.

When $\frac{\omega_1}{\omega_2} = 1$, then $\omega_1 = \omega_2 = 0.5$, indicating that the recommendation based on the user group and the recommendation based on the item group have the same weight and have the same impact on the recommendation result.

When $\omega_1 > \omega_2$, indicating that the recommendation based on the user group is more important than the recommendation based on the item group, and the user group has a greater influence on the recommendation result.

When $\omega_1 < \omega_2$, indicating that the recommendation based on the item group is more important than the recommendation based on the user group, and the item group has a greater influence on the recommendation result.

ω_1 and ω_2 balance the influence of UbCF and IbCF on the final recommendation result, avoid the excessive deviation of the two influencing factors and reduce the recommendation quality of the algorithm.

3.2.1 RMSE vs MAPE

Generally, the mean absolute percentage error (MAPE) can reflect the average value of the absolute difference between the actual value and the predicted value of the regression model, expressed as a percentage of the actual value [22]. It normalizes the error of each point to reduce the absolute error effect caused by a single outlier. However, RMSE (8) can well reflect the degree of deviation between the predicted value of the regression model and the true value [23]. If there are individual outliers with very large deviations, it can be detected sensitively. For sudden changes in user interests or instantaneous changes in the popularity of items, RMSE can capture them, resulting in a better training system.

3.3. Canopy–K-means based Combined Collaborative Filtering (CKComCF)

To overcome the aforementioned shortcomings, we employ the Canopy–K-means algorithm before the traditional CF. Accordingly, CKComCF uses the Canopy algorithm to enhance the robustness of the impact of k on the K-means algorithm. Further, clustered data can decrease the computational complexity of the traditional CF and enhance the instantaneity of RS. Moreover, the fusioned UbCF and IbCF can increase the recommendation accuracy.

First, we used the unsupervised pre-clustering to obtain the number of canopies for k in K-means. It is efficient when the sample dataset is huge. According to the number of canopies, K-means can provide the initial input to cluster users and items. In each cluster, we calculate the PCC similarity distance between each sample. Once the user and item similarity matrices are calculated, UbCF and IbCF can run separately. After that, the first RMSE for the later error propagation can be calculated. Finally, using the proposed RMSE weighted-average method generate the final prediction from CKComCF. The proposed algorithm, CKComCF, is presented as pseudocode in Algorithm 1.

3.3.1 Canopy and K-means Clustering process

Canopy steps: initial dataset and Threshold T1, T2. T1 is 0.3 and 0.2 is applied. Since the score is between 0-5, in the process of calculating Canopy, Euclidean distance will be used to calculate the distance between vectors, thereby obtaining clusters. In order to obtain more accurate clustering, we also tried to use the effects of 0.1, 0.4, 0.7, etc., none of which is as good as this combination. The Threshold between the five-point score matrix is around 0.2, 0.3.

The huge sparse scor1e matrix, when dealing with K-means clustering, will encounter a big problem. There are several methods can be used to analyze the data with the missing values for K-means [24]. For instance, use the global constant to fill in the missing value, disregard the tuple, manually fill in the missing value; using the average attribute to fill in the missing value; for all samples belonging to the same class as the tuple given, use the average attribute; use the most likely value to fill in the missed value [25].

3.4 Other method to improve CF

Using different kinds of factors form users or items profile. Such as time factor, social network information, location, item's tag, hot item penalty or etc. Depends on different structure of exist datasets, some of factor is require but the dataset does not provide. However, Canopy–K-means method can only use the basic rating matrix to predict effectively. Appendix A 2 shows a survey about GroupLens[1] Datasets.

[1] https://grouplens.org/

Algorithm 1 Canopy–K-means based Combined Collaborative Filtering

```
input $R$ (User-item actual rating matrix)
input $t_1 \leftarrow 0.3$ (Threshold $T1$ in Canopy clustering)
input $t_2 \leftarrow 0.2$ (Threshold $T2$ in Canopy clustering)
input $x \leftarrow 5$ (Number of Top-N neighbors in CF)
output $P$ (User-item predicted rating matrix)

procedure CKComCF($R, t_1, t_2$)
    $U \leftarrow R$
    $I \leftarrow R^T$

    $k^U \leftarrow$ Canopy($U, t_1, t_2$)                                  ▷ Canopy clustering
    $k^I \leftarrow$ Canopy($I, t_1, t_2$)

    $\{M_1^U, M_2^U, \dots, M_{k^U}^U\} \leftarrow$ KMeans($U, k^U$)        ▷ K-means clustering
    $\{M_1^I, M_2^I, \dots, M_{k^I}^I\} \leftarrow$ KMeans($I, k^I$)

    $P^U \leftarrow \{\}$                                   ▷ User-based Collaborative Filtering (UbCF)
    for $c \leftarrow 1$ to $k^U$ do
        $S \leftarrow \{\}$
        for $u \leftarrow$ Min($M_c^U$) to Max($M_c^U$) do                 ▷ PCC for UbCF
            $\vec{u} \leftarrow M_c^U$.get($u$)
            for $v \leftarrow$ Min($M_c^U$) to Max($M_c^U$) do
                $\vec{v} \leftarrow M_c^U$.get($v$)
                $s_{u,v} \leftarrow$ PCC($\vec{u}, \vec{v}$)
                $S$.insert($s_{u,v}$)
            end for
        end for
        $p_{u,i} \leftarrow$ UbCF($M_c^U, S, x$)                            ▷ Calculate the UbCF
        $P^U$.insert($p_{u,i}$)
    end for
    $e^U \leftarrow$ RMSE($P^U$)                                            ▷ Calculate the RMSE of UbCF

    $P^I \leftarrow \{\}$                                   ▷ Item-based Collaborative Filtering (IbCF)
    for $c \leftarrow 1$ to $k^I$ do
        $S \leftarrow \{\}$
        for $i \leftarrow$ Min($M_c^I$) to Max($M_c^I$) do                 ▷ PCC for IbCF
            $\vec{i} \leftarrow M_c^I$.get($i$)
            for $j \leftarrow$ Min($M_c^I$) to Max($M_c^I$) do
                $\vec{j} \leftarrow M_c^I$.get($j$)
                $s_{i,j} \leftarrow$ PCC($\vec{i}, \vec{j}$)
                $S$.insert($s_{i,j}$)
            end for
        end for
        $p_{u,i} \leftarrow$ IbCF($M_c^I, S, x$)                            ▷ Calculate the IbCF
        $P^I$.insert($p_{u,i}$)
    end for
    $e^I \leftarrow$ RMSE($P^I$)                                            ▷ Calculate the RMSE of IbCF

    $\omega_1 \leftarrow 1 - \frac{e^U}{e^U + e^I}$                  ▷ Calculate the ComCF weight using RMSE

    $\omega_2 \leftarrow 1 - \frac{e^I}{e^U + e^I}$

    $P \leftarrow \omega_1 \times P^U + \omega_2 \times P^I$       ▷ Calculate the Combined Collaborative Filtering
    return $P$
end procedure
```

4. Experiment

In this section, we detail our experiments and results. First, experimental datasets and evaluation metrics are introduced. The results and experimental analysis are discussed at the end.

All the experiments are carried out with the Dask.distributed framework (https://distributed.dask.org) and run on a workstation equipped with 2 x 8-cores Intel Xeon E5-2620v4@2.10GHz CPU, 128GB DDR4-2666 ECC/REG memory, and Samsung 960 EVO PCIe 250GB SSD.

4.1. Dataset

The experiments have been conducted on two widely used and well-known datasets, i.e., MovieLens (https://grouplens.org/datasets/movielens) and Netflix Prize (https://www.kaggle.com/netflix-inc/netflix-prize-data). The MoveieLens dataset contains 100,836 ratings (0.5-5 scales with 0.5 increments) from 610 users on 9,742 movies, in which each user has rated at least 20 items at a density rate of 1.69%. The Netflix Prize dataset contains 100,480,507 ratings (1-5 scales with 1 increment) from 480,189 users on 17,770 movies, in which each user has rated at least 1 item at a density rate of 1.17%. Moreover, we randomly sampled the data based on the Gaussian distribution as 30 users on 100 items, in which each user has rated at least 1 item at density rates of 8.63% and 7.47%. The specifications of these datasets are listed in Table 1.

Table 1 Dataset specification

	MovieLens	**MovieLens (sampled)**	**Netflix Prize**	**Netflix Prize (sampled)**
#Users	610	30	480,189	30
#Movies	9,742	100	17,770	100
#Ratings	100,836	259	100,480,507	224
Rating range	[0.5, 5], 0.5	[0.5, 5], 0.5	[1, 5], 1	[1, 5], 1
Density	0.0169	0.0863	0.01177	0.07466
Max #ratings of user	2,698	57	17,653	26
Min #ratings of user	20	1	1	1

4.2. Evaluation

In order to evaluate the recommended accuracy of the proposed algorithm, RMSE is employed to measure our performance prediction algorithm. RMSE can be obtained by calculating the standard deviation between the actual rating and the predicted rating for each user. The smaller the value of RMSE, the higher the accuracy of the recommendation algorithm. The formula of RMSE is as (8):

$$RMSE(P) = \sqrt{\frac{1}{|R|}\sum_{(p_{u,i}\in P)\cap(r_{u,i}\in R)}\left(p_{u,i} - r_{u,i}\right)^2} \qquad \textbf{(8)}$$

where P denotes the predicted rating matrix, while R is the actual rating matrix. $p_{u,i}$ denotes user u's predicted rating for item i, while $r_{u,i}$ denotes user u's actual rating for item i respectively.

4.3. Result

The neighbors of CF users or items that have similar preferences based on recommendations, which are denoted as $NN(u)$ and $NN(i)$ in (3) and (4). Top-N or other kinds of pre-filtering methods can be used to determine these neighbors. In our experiments, k-NN is employed, which requires the parameter x. Choosing the proper x is required as it affects the accuracy. Figure 1 - Figure 4 are showing that RMSE significantly changes depending on x.

We derived the proper x value, which is the smallest RMSE obtained from traditional CF—UbCF and IbCF for calculating the proposed RMSE-based ComCF by testing the x range [1, 30] of UbCF and the x range [1, 100] of IbCF. Moreover, the MAPE-based CF[22] was compared in the whole analysis.

We performed two experiments to show the effectiveness of RMSE-minimization for ComCF and the effectiveness of CKComCF in MovieLens and Netflix Prize datasets.

4.3.1. Effectiveness of RMSE-minimization for ComCF

For the MovieLens dataset, x obtained values of 12 from UbCF and 31 from IbCF for the smallest RMSE and obtained 12 from UbCF and 8 from IbCF for the smallest MAPE (Figure 1). For RMSE-based ComCF based on (6) and (7), the accuracy of RMSE was 0.2326 and for MAPE-based ComCF based on [22], the accuracy of RMSE was 0.2838.

For the Netflix Prize dataset, x obtained a value of 7 from UbCF and IbCF for the smallest RMSE and obtained 7 from UbCF and 6 from IbCF for the smallest MAPE (Figure 2). For RMSE-based ComCF by our proposed equations, the accuracy of RMSE was 0.1609, and for MAPE-based ComCF based on [22], , the accuracy of RMSE was 0.1699. The results are listed in Table 2.

The results of the proposed RMSE-minimization approach are listed in Table 3. For the MovieLens dataset, the proposed approach has improved accuracy by 54.37%, 14.45%, 18.04% compared with UbCF, IbCF, and MAPE-based ComCF, respectively. For Netflix Prize dataset, the proposed approach has improved accuracy by 48.95%, 29.21%, 5.30% compared with UbCF, IbCF, and MAPE-based ComCF, respectively. Therefore, the proposed RMSE-based ComCF is much better than the traditional CF and more accurate than the MAPE-based ComCF.

4.3.2. Effectiveness of CKComCF

For Canopy clustering, hyperparameters, $T1$ and $T2$ are set as 0.3 and 0.2, respectively. The pseudocode of the proposed approach is displayed as Algorithm 1. For the MovieLens dataset, the results of Canopy clustering are 3 and 4 for k^U and k^I, respectively, and x has obtained values of 9 and 12 for UbCF and IbCF, respectively (Figure 3). For the Netflix Prize dataset, the results of Canopy clustering are 3 and 4 for k^U and k^I, respectively, and x came out 5 and 9 for UbCF and IbCF, respectively (Figure 4). The results are listed in Table 4. Furthermore, the calculation time of each algorithm for different datasets is listed in Table 5.

The results of the proposed CKComCF result are listed in Table 6. As per the comparative analysis, our approach did not improve the accuracy, but significantly reduced the calculation time by 41.84% and 64.77% for MovieLens and Netflix Prize datasets, respectively. On the other hand, in comparison with the traditional CF—UbCF and IbCF––our approach did not always improve the accuracy but consistently maintained it. In comparison with the UbCF, our approach has improved the accuracy by 48.72% and 22.08% for MovieLens and Netflix Prize datasets, respectively. In comparison with the IbCF, our approach has improved accuracy by 48.72% for the MovieLens dataset and 53.18% for the Netflix Prize dataset.

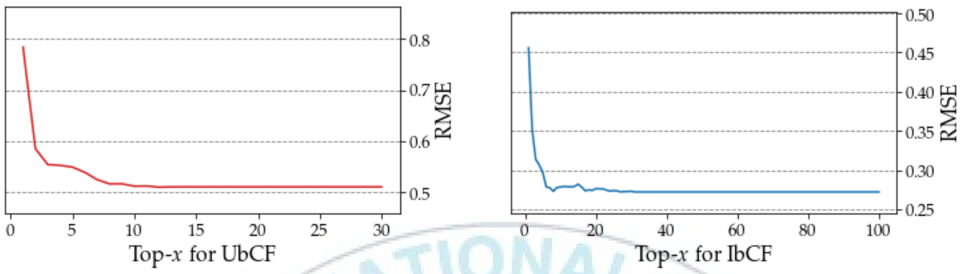


Figure 1 ComCF k-NN with MovieLens dataset

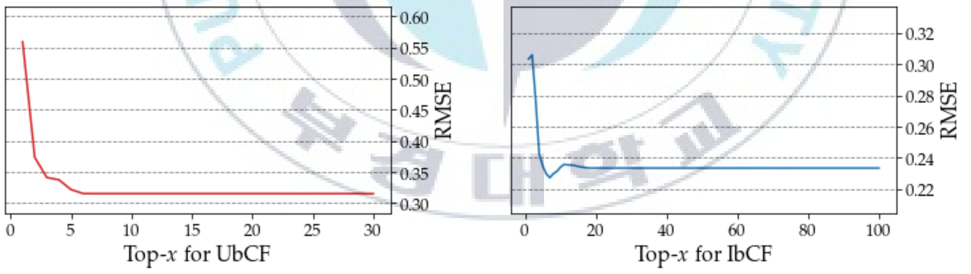


Figure 2 ComCF k-NN with Netflix Prize dataset

Table 2 Comparison of the accuracy of proposed RMSE-minimization approach with traditional CF and existing weighting method[22]

Dataset	Type	ComCF weighting	Top-x for UbCF	Top-x for IbCF	RMSE
MovieLens	UbCF		12		0.5098
	IbCF			31	0.2719
	ComCF	MAPE-based	12	8	0.2838
	ComCF	RMSE-based	12	31	**0.2326**
Netflix Prize	UbCF		7		0.3152
	IbCF			7	0.2273
	ComCF	MAPE-based	7	6	0.1699
	ComCF	RMSE-based	7	7	**0.1609**

Table 3 Improvement of proposed RMSE-minimization approach

(±RMSE%, lesser is better)

Dataset	RMSE-based ComCF		
	UbCF	IbCF	MAPE-based ComCF
MovieLens	-54.37%	-14.45%	-18.04%
Netflix Prize	-48.95%	-29.21%	-5.30%

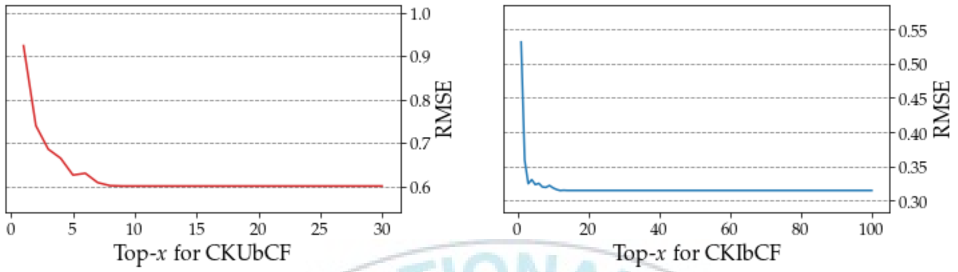


Figure 3 CKComCF k-NN with MovieLens dataset

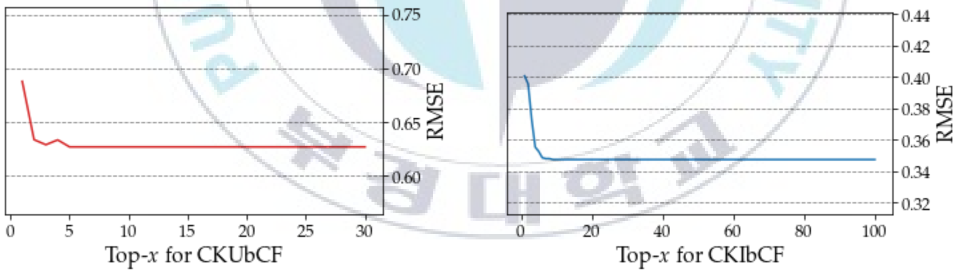


Figure 4 CKComCF k-NN with Netflix Prize dataset

Table 4 Comparison of the accuracy of proposed method "CKComCF" with clustering based traditional CF

Dataset	Type	ComCF weighting	k^U	k^I	Top-x for UbCF	Top-x for IbCF	RMSE
MovieLens	CKUbCF		3		9		0.6016
	CKIbCF			3		1	0.5311
	CKComCF	MAPE-based	3	3	9	12	0.3077
	CKComCF	RMSE-based	3	3	9	12	**0.2614**
Netflix Prize	CKUbCF		5		5		0.6268
	CKIbCF			5		9	0.3472
	CKComCF	MAPE-based	5	5	5	9	0.2972
	CKComCF	RMSE-based	5	5	5	9	**0.2456**

Table 5 Comparison of the calculation time of proposed method "CKComCF" with traditional CF

Dataset	Type	RMSE	Calculation time
MovieLens	UbCF	0.5098	88.191
	IbCF	0.2719	232.97
	ComCF	**0.2326**	321.2
	CKComCF	0.2614	**186.8**
Netflix Prize	UbCF	0.3152	139.3
	IbCF	0.2273	423.79
	ComCF	**0.1609**	563.1
	CKComCF	0.2456	**198.4**

Table 6 Improvement of proposed method “CKComCF” (±RMSE% and ±Time%, lesser is better)

Dataset	CKComCF					
	UbCF		IbCF		ComCF	
	±RMSE%	**±Time%**	**±RMSE%**	**±Time%**	**±RMSE%**	**±Time%**
MovieLens	**-48.72%**	111.87%	**-3.86%**	42.43%	12.38%	**-41.84%**
Netflix Prize	**-22.08%**	-19.82%	8.05%	**-53.18%**	52.64%	**-64.77%**

5. Conclusion

In this thesis, we introduced a weighted RMSE minimized model for ComCF. Our proposed model, CKComCF solved the scalability problem and retained the accuracy rate productively. Finally, we conducted experiments on real-world datasets with our approach and other algorithms for performance evaluation. The results indicated that the proposed RMSE minimized method was effective. Although Canopy–K-means did not guarantee the accuracy of predictions as the clusters were not optimal, the combination approach alleviated it. Canopy–K-means and ComCF well complemented each other. In the future, we will optimize the clustering function for the model as well as consider changes in the preference changes in the of users and the long-tail effect of items on the system.

Appendix

A 1 Four methods to calculate PCC
(CF customized PCC, Excel, Pandas, Spark libraries PCC func.)

Movies / **Users**	***Paw Patrol***	***Moon and Me***	***Peppa Pig***	***Frozen***	***My Little Pony***
Audrey	1	5	2	5	5
Quenby	2		3	5	4

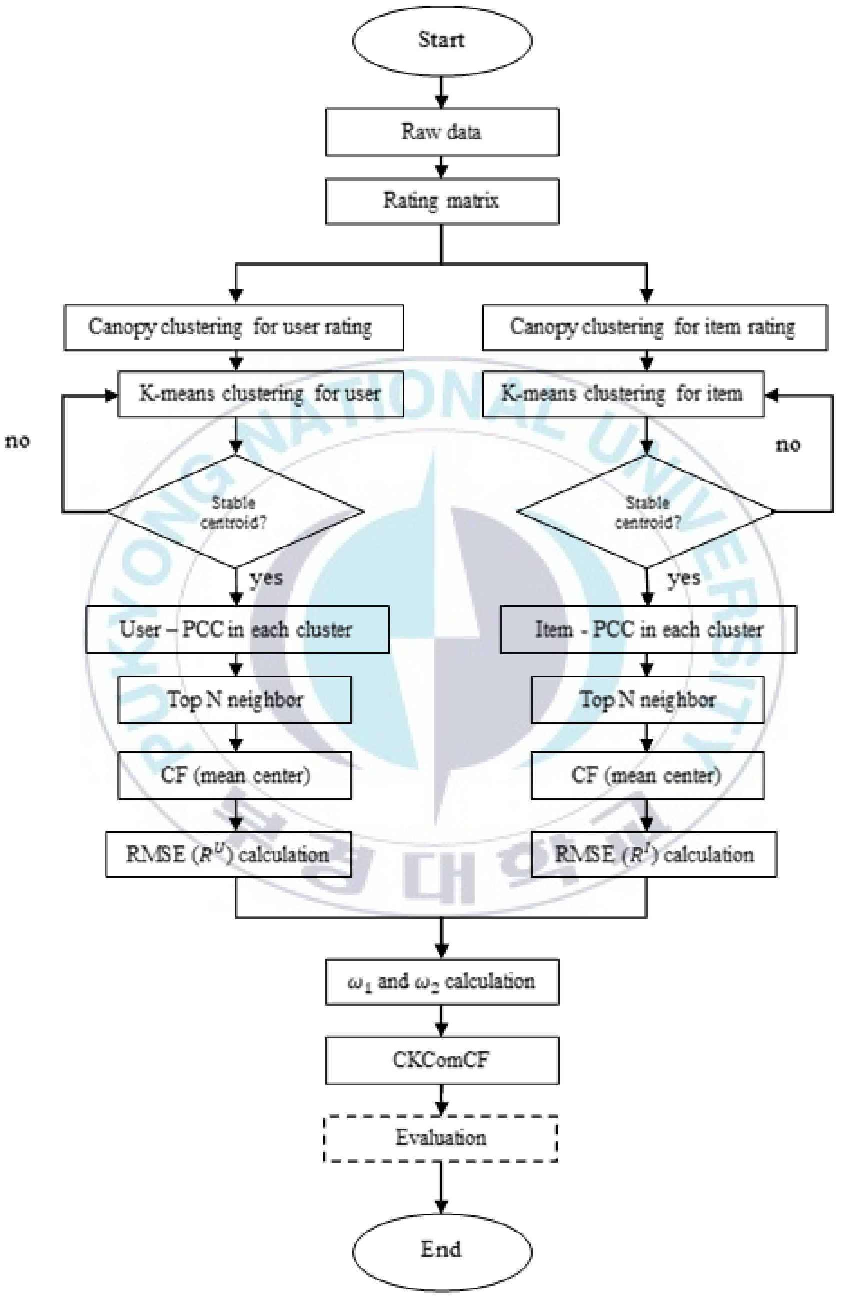
Methods	**Results**
CF required PCC func.	**0.921790586**
Excel PCC func.	0.939336437
Pandas PCC func.	0.939336437
Spark PCC func.	0.173348743

A 1 shows the ratings of each movie by users Audrey and Quenby. The results were calculated by using different methods — CF required PCC, Excel, Pandas, Spark libraries PCC function. The outcomes of PCC functions of Excel, Pandas and Spark are different with the CF required PCC calculation.

A 2 GroupLens Dataset Survey

Dataset Name	Dataset published Date	Dataset record time	#user	#movie	#rating	rating range (min,max,step)	# had rated at least	time	#tags (total/set)	location	#grenres (total/set)
MovieLens Spark 30 Dataset			30	100	1,501	1,5,1		provided			
HetRec 2011, MovieLens + IMDb/Rotten Tomatoes	May, 2011		2,113	10,197	855,598	0.5,5,0.5		provided	47957/13222	47,899	
recommended for new research, MovieLens 25M Dataset	21-Nov-19	Jan 09, 1995 ~ Nov 21, 2019	162,541	62,423	25,000,095	0.5,5,0.5	20+	provided	1,093,360		20809/20
recommended for education and development, MovieLens Latest Datasets Small	26-Sep-18	Mar 29, 1996 ~ Sep 24, 2018	610	9,742	100,836	0.5,5,0.5	20+	provided	3,683		provided
recommended for education and development, MovieLens Latest Datasets Full	26-Sep-18	Jan 09, 1995 ~ Sep 24, 2018	283,228	58,098	27,753,444	0.5,5,0.5	1+	provided	1,108,997		provided
older datasets, MovieLens 100K Dataset	Apr, 1998	Sep 19, 1997 ~ Apr 22, 1998	943	182	100,000	1,5,1	20+	provided			provided
MovieLens 1M Dataset	2000		6,040	3,900	1,000,209	1,5,1	20+	provided		zip code	provided
MovieLens 10M Dataset	2015		71,567	10,681	10,000,054	0.5,5,0.5	20+	provided	95,580		provided
MovieLens 20M Dataset	17-Oct-16	Jan 09, 1995 ~ Mar 31, 2015	138,493	27,278	20,000,263	0.5,5,0.5	20+	provided	465,564		provided
MovieLens Tag Genome Dataset	Sep, 2012			9,734					1,128		

A 3 Flow chart for Canopy–K-means based Combined Collaborative Filtering


Start
Raw data
Rating matrix
Canopy clustering for user rating
Canopy clustering for item rating
K-means clustering for user
K-means clustering for item
no
no
Stable centroid?
Stable centroid?
yes
yes
User – PCC in each cluster
Item - PCC in each cluster
Top N neighbor
Top N neighbor
CF (mean center)
CF (mean center)
RMSE (R^U) calculation
RMSE (R^I) calculation
ω_1 and ω_2 calculation
CKComCF
Evaluation
End

A 4 Notations

Variable	Description
t_1, t_2	Threshold $T1$ and $T2$ in Canopy clustering
k^U, k^I	k in K-means clustering for UbCF or IbCF
x	Number of Top-N neighbors in CF (which is k in k-Nearest Neighbors)
m	Number of users
n	Number of items
$r_{u,i}$	Actual rating of user u for item i
R	User-item Actual rating matrix
$\vec{u}, \vec{v}, \vec{i}, \vec{j}$	Actual rating vector (of user u or v, or item i or j)
U, I	Actual rating matrix for UbCF or IbCF
$p_{u,i}$	Predicted rating of user u for item i
P	User-item Predicted rating matrix
P^U, P^I	Predicted rating matrix from UbCF or IbCF
c	Cluster ID
M_c^U, M_c^I	c-th clustered rating matrix for UbCF or IbCF
$s_{a,b}$	Similarity between a and b
S	Similarity matrix
e^U, e^I	RMSE of UbCF or IbCF predicted rating matrix
ω_1, ω_2	Weight of UbCF or IbCF for ComCF

References

[1] I. M. Dunham, “Big Data: A Revolution That Will Transform How We Live, Work, and Think,” *AAG Rev. Books*, vol. 3, no. 1, pp. 19–21, Jan. 2015, doi: 10.1080/2325548X.2015.985533.

[2] R. PAN, C. GE, L. ZHANG, W. ZHAO, and X. SHAO, “A New Similarity Model Based on Collaborative Filtering for New User Cold Start Recommendation,” *IEICE Trans. Inf. Syst.*, vol. E103.D, no. 6, pp. 1388–1394, Jun. 2020, doi: 10.1587/transinf.2019EDP7258.

[3] F. Ricci, L. Rokach, B. Shapira, P. B. Kantor, and F. Ricci, *Recommender Systems Handbook*. Boston, MA: Springer US, 2011.

[4] N. J. Belkin and W. B. Croft, “Information filtering and information retrieval: Two Sides of the Same Coin?,” *Commun. ACM*, vol. 35, no. 12, pp. 29–38, Dec. 1992, doi: 10.1145/138859.138861.

[5] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, “Using collaborative filtering to weave an information tapestry,” *Commun. ACM*, vol. 35, no. 12, pp. 61–70, Dec. 1992, doi: 10.1145/138859.138867.

[6] S. Jiang, S.-C. Fang, Q. An, and J. E. Lavery, “A sub-one quasi-norm-based similarity measure for collaborative filtering in recommender systems,” *Inf. Sci. (Ny).*, vol. 487, pp. 142–155, Jun. 2019, doi: 10.1016/j.ins.2019.03.011.

[7] T. Arsan, E. Koksal, and Z. Bozkus, “Comparison of Collaborative Filtering Algorithms with Various Similarity Measures for Movie Recommendation,” *Int. J. Comput. Sci. Eng. Appl.*, vol. 6, no. 3, pp. 1–20, 2016, doi: 10.5121/ijcsea.2016.6301.

[8] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005, doi: 10.1109/TKDE.2005.99.

[9] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “GroupLens: An Open Architecture for Collaborative Filtering of Netnews,” in *Proceedings of the 1994 ACM conference on Computer supported cooperative work - CSCW ’94*, 1994, pp. 175–186, doi: 10.1145/192844.192905.

[10] G. Koutrika, B. Bercovitz, and H. Garcia-Molina, “FlexRecs: Expressing and Combining Flexible Recommendations,” in *Proceedings of the 35th SIGMOD international conference on Management of data - SIGMOD ’09*, 2009, p. 745, doi: 10.1145/1559845.1559923.

[11] L. Jiang, Y. Cheng, L. Yang, J. Li, H. Yan, and X. Wang, “A trust-based collaborative filtering algorithm for E-commerce recommendation system,” *J. Ambient Intell. Humaniz. Comput.*, vol. 10, no. 8, pp. 3023–3034, Aug. 2019, doi: 10.1007/s12652-018-0928-7.

[12] T. MA *et al.*, “Social Network and Tag Sources Based Augmenting Collaborative Recommender System,” *IEICE Trans. Inf. Syst.*, vol. E98.D, no. 4, pp. 902–910,

2015, doi: 10.1587/transinf.2014EDP7283.

[13] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, “Item-based collaborative filtering recommendation algorithms,” in *Proceedings of the tenth international conference on World Wide Web - WWW ’01*, 2001, pp. 285–295, doi: 10.1145/371920.372071.

[14] Y. Ding and X. Li, “Time Weight Collaborative Filtering,” *Time*, pp. 485–492, 1935, doi: 1-59593-140-6/05/0010.

[15] Z. M. Feng and Y. D. Su, “Application of Using Simulated Annealing to Combine Clustering with Collaborative Filtering for Item Recommendation,” *Appl. Mech. Mater.*, vol. 347–350, pp. 2747–2751, Aug. 2013, doi: 10.4028/www.scientific.net/AMM.347-350.2747.

[16] E. Aldhahri, V. Shandilya, and S. Shiva, “Towards an effective crowdsourcing recommendation system: A survey of the state-of-the-art,” *Proc. - 9th IEEE Int. Symp. Serv. Syst. Eng. IEEE SOSE 2015*, vol. 30, pp. 372–377, 2015, doi: 10.1109/SOSE.2015.53.

[17] L. Bai, M. Hu, Y. Ma, and M. Liu, “A Hybrid Two-Phase Recommendation for Group-Buying E-commerce Applications,” *Appl. Sci.*, vol. 9, no. 15, p. 3141, Aug. 2019, doi: 10.3390/app9153141.

[18] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, 1967, pp. 281–297.

[19] A. McCallum, K. Nigam, and L. H. Ungar, “Efficient clustering of high-

dimensional data sets with application to reference matching," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '00*, 2000, pp. 169–178, doi: 10.1145/347090.347123.

[20] J. Li *et al.*, "Category Preferred Canopy–K-means based Collaborative Filtering algorithm," *Futur. Gener. Comput. Syst.*, vol. 93, pp. 1046–1054, Apr. 2019, doi: 10.1016/j.future.2018.04.025.

[21] A. Chefrour and L. Souici-Meslati, "AMF-IDBSCAN: Incremental density based clustering algorithm using adaptive median filtering technique," *Inform.*, vol. 43, no. 4, pp. 495–506, 2019, doi: 10.31449/inf.v43i4.2629.

[22] P. Thakkar, K. Varma, and V. Ukani, "Outcome Fusion-Based Approaches for User-Based and Item-Based Collaborative Filtering," in *Technology for Intelligent Systems (ICTIS 2017)*, vol. 84, no. Ictis, S. C. Satapathy and A. Joshi, Eds. Cham: Springer International Publishing, 2018, pp. 127–135.

[23] Z. Wang, *Deep Learning Recommender System*. Electronics Industry Press, 2020.

[24] S. Agarwal, "Data Mining: Data Mining Concepts and Techniques," in *2013 International Conference on Machine Intelligence and Research Advancement*, 2013, pp. 203–207, doi: 10.1109/ICMIRA.2013.45.

[25] G. Xue *et al.*, "Scalable Collaborative Filtering Using Cluster-based Smoothing *," pp. 114–121, 2005, doi: 1-59593-034-5/05/0008.

Acknowledgements

First of all, I would like to honor and thank God, the Almighty, who gave me endless blessings, wisdom, opportunities and support, without His graces, this study would not have been possible.

Immeasurable appreciation and deepest gratitude for the help and support are extended to the following persons who in one way or another have contributed in making this study possible.

I would like to express the deepest appreciation to my advisor, Prof. Ha-Joo Song, your expertise in the formulation of the research subject and technique has been invaluable. I would like to thank you for providing me with the opportunity in this field, supporting my research and helping me to develop as a research scientist.

I would like to thank my IDB Lab members for their amazing partnership and members in our department at PKNU who share the knowledge and useful information with me. You have greatly helped me and have always been able to assist me.

I would also like to thank Dr. Ruth Yeung and Dr. Veronica Lam, my former professors at the Macao Institute for Tourism Studies, for your valuable guidance, all of your encouragement, caring, and support.

I would like to thank my friends who shared the sources with me, offering me a lot of assistance and support for my studies. Thank you for accompanying with me all the time.

Finally, I am especially grateful to my mom, dad, and sister. My parents have sacrificed their lives and given unconditional love and care for my sister and me. Throughout my life, my sister has been my best friend, and I love you dearly and thank you for all your support. I love my family so much, and you are there for me, always.