

Dynamic Weighted Hybrid Recommender Systems

Hong-Quan Do*, Tuan-Hiep Le**, Byeongnam Yoon***

*VNU Information Technology Institute, Vietnam National University, Hanoi, Vietnam

**FPT University, Hanoi, Vietnam

***Computer Science Department, Kyonggi University, Suwon Korea

quandh@vnu.edu.vn, hiepltgch16527@fpt.edu.vn, tomayoon@kgu.ac.kr

Abstract— Recommender Systems (RSs) have emerged since the mid-90s for dealing with the problem of information overload. They are commonly defined as software tools and techniques that serve as an assistant providing suggestions to the users. The two most familiar recommendation techniques are probably Collaborative filtering (CF) and Content-based filtering (CBF). Whereas CF computes recommendations based on past ratings of users with similar preferences, CBF assumes that each user operates independently, thus exploits only information derived from item features. Technically speaking, the performance of every single recommendation algorithm is limited and each has its own strengths and weaknesses, so recently more attentions are paid to the hybrid recommendation algorithms. In this work, we focus on the weighted hybridization and rather than using fixed weighted for the combination, we aim to propose a simple method to dynamic weight a combination of CF and CBF. Our experimental results on one of the most popular public datasets in the field of RSs - MovieLens have verified the effectiveness of our strategy against the traditional CF, and CBF approaches. It not only boost the prediction performance, but also alleviate the problem of new item cold start.

Keywords— Hybrid Recommender, Dynamic Weighted, Collaborative filtering, Content-based filtering, Cold start

I. INTRODUCTION

With the exponential growth of the digital information in the recent decades, recommender systems (RSs) are now an integral significant part of any modern system. They serve as an assistant helping users in searching for various types of items and services. Historically, the two main approaches of building recommender systems are collaborative filtering (CF) and content-based filtering (CBF). The two approaches can also be combined as hybrid recommender systems.

To begin with, Collaborative filtering recommender systems [1] is the most mature and the most commonly implemented in different application areas, for example Amazon since 1998. CBF uses customer's ratings to categorize users in groups according to their similarity. Recommendations are then inferred by taking into account ratings of users' in the same group (Figure 1).

In contrast to CF, Content-based recommender systems [2] computes recommendations according to the features of items that the user preferred in the past (Figure 2). The main difference between this approach and the CF is that CBF offers the recommendation based on similarity not only by rating, but

also the information from the products i.e., the movie title, the year, the actors, the genre. In order to implement this CB, it is necessary to process information describing each item.

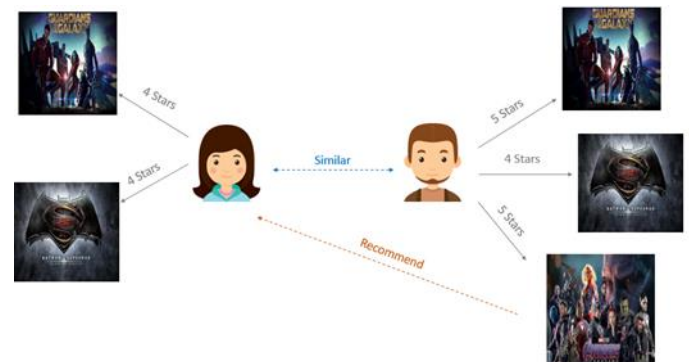


Figure 1. Example of collaborative filtering



Figure 2. Example of content based filtering

The performance of every single recommendation algorithm, however, is limited and each has its own strengths and weaknesses. For example, Collaborative filtering techniques perform well when there is sufficient rating information [3]. Otherwise, their effectiveness is limited when the problems of rating sparsity or cold-start problem for new items occur. CBF, on the other hand, is able to handle the case of new items, but items must be encoded with meaningful features. Therefore, recently more attentions are paid to the hybrid recommendation algorithms – that combines two or more filtering methods to enhance quality of the recommendation. In this study, we particularly emphasize on weighted hybrid recommenders. Using all item's existing information, this approach aims at

combining CF and CBF methods in a dynamic weighting strategy in order to propose a more accurate recommendation. Different from related works, our contribution is in formulation of the weighted factor α based on confidence estimation for each active item, so that, the calculation of prediction takes majority from collaborative filtering or content-based filtering depending on such confidence level. Our dynamic recommendation method will be compared against traditional collaborative filtering, and content based filtering. The experimental results on one of the most popular public datasets in the field of movie recommendation service - MovieLens show that the proposed system has lower Root Mean Squared Error (RSME), meaning a better accuracy. Moreover, the recommendations given by our system is able to alleviate the problem of new item cold start.

The remaining part of this report will start with discussing literature review in Chapter 2. Our contribution is summarized in Chapter 3. Next, Chapter 4 describes the proposed dynamic weighted hybrid technique. The implementation and evaluation details will be depicted in Chapter 5. The report ends with the description of conclusion and future work in Chapter 6.

II. LITERATURE REVIEW

A. Recommender Systems

Recommender systems handle the problem of information overload that users normally encounter by providing them with personalized, exclusive content and service recommendations. According to [4], a recommender system can be distinguished into two types: memory-based and model-based approaches [1]. (1) memory-based approaches generate recommendations through user-item rating matrix. This is the case of Liang et. al., 2016 [21]; 2) model-based approaches design a model to learn users' preference patterns by leveraging observed user ratings as training data, then predict unobserved rating with the help of that model. Types of model-based approaches include clustering based algorithms, matrix factorization based algorithms, and deep learning. It is noticeable that model-building can be accomplished offline prior to online recommendation. Thereby, model-based CF is typically faster in terms of recommendation time using the pre-computed model. A number of model-based studies have attempted include Sarwar, B. et. al., 2001 [18], and Deshpande, M., 2004 [19].

Furthermore, various approaches for building recommendation systems have been developed, which can mainly utilize either Collaborative Filtering (CF), or Content-based Filtering (CBF). The two approaches can also be combined as hybrid recommender systems.

B. Collaborative Filtering

Research on recommender systems gained importance after the emergence of collaborative filtering. In the early 1990s, Tapestry [5] was the first manual collaborative filtering system allowing the user to harness the reactions of the previous readers to determine relevance to them. Automated collaborative filtering systems soon followed. GroupLens [6, 7] used this technique to identify articles that are likely to be

interesting to a particular user. Users only needed to provide ratings or perform other observable actions. For many year, it has still represented the most commonly adopted technique in crafting academic and commercial recommender systems.

In the memory based approach of CF, the filtering can use the user-item ratings directly and proceed by aggregating the ratings of similar users (user-based CF) and/or similar items (item-based CF). A user-item filtering takes a particular user, find users that are similar to that user based on similarity of ratings, and recommend items that those similar users liked. In contrast, item-item filtering will take an item, find users who liked that item, and find other items that those users or similar users also liked. It takes items and outputs other items as recommendations. Item-item collaborative filtering was invented and used by Amazon.com in 1998 [17]. It was first published in an academic conference in 2001 [8].

Two weak points are recognised in CF systems: “data sparsity”, and “cold start”. The former occurs when the number of ratings from users is small in comparison with the total number of items, so that the probability of finding users that rate the same items is too low to make good estimations. The latter happens when new users or new items arrive. The traditional collaborative filtering assumes that each user or item has some interactions so that they can infer ratings of similar users/items. However, in case of new user/new items with no or very few interactions are available, then a pure collaborative algorithm cannot recommend the item.

C. Content-based Filtering

Unlike collaborative methods that only rely on the user-item interactions, content based approaches use additional information about users and/or items. For example, some characteristics of a movie that could be observed includes movie title, release date, video release date, genre, to mention it but a few. The CBF is based on the concept that items with similar characteristics will be rated similarly. The process of a content-based recommender, therefore, requires to create an item profile (or feature vector) for each item. Then, from the item profile and the input user ratings, the content based algorithm tries to establish user's models in which each model, named θ_u , can be represented for a user u . Finding θ_u for each user could be considered as a regression problem if ratings are in a range of values or classification problem if ratings are in some specific cases (e.g. like/dislike). Choosing regression model or classification model depends on the type of application. The training data set, that is used to create θ_u , is pairs of item profile, and user ratings. Finding missing values in the utility matrix is predicting values for unrated items when applying θ_u on them. CBF, in contrast to collaborative filtering, is able to handle the case of new items, but items must be encoded with meaningful features.

D. Hybrid Filtering

In order to mitigate some of the problems identified, the combination of different recommendation techniques is a widespread practice these days. A hybrid approach is expected to avoid disadvantages of existing approach besides combining

advantages of them. In the survey of Borràs et. al., 2014 [16], it can be observed an increasing trend in the exploitation of recommendation filtering techniques since 2012, mainly in hybrid systems. From 2008 to 2011 only 25% of systems used such method, whereas since 2012 the percentage has increased to 75%. Hybrid systems can integrate recommendation techniques in different ways. Three approaches can be distinguished:

- **Switching:** The system switches between recommendation techniques depending on the particular situation. For instance, the Daily Learner [9] system uses a content/collaborative hybrid in which a content-based recommendation method is applied, then if the content-based system cannot make a recommendation with sufficient confidence, a collaborative recommendation is attempted.
- **Sequential use:** The features produced by a specific recommendation technique are fed into another recommendation technique. This is the case of Good et al. [10] in which content-Based characteristics are added to collaborative Models, and Basu C. et. al. [11] in which collaborative filter's ratings are used in a content-based system as a feature for recommending movies.
- **Weighted use:** The score of several recommendation techniques are combined to produce a single recommendation using either a linear combination of ratings [12] or a voting scheme [13]. Another instances are Borràs et al. (2011) [14], Lucas et al. (2013) [15].

In view of the disadvantages of the content-based and collaborative filtering, many researchers have proposed hybrid approaches that utilizes the two techniques. Very often the two algorithms are combined using a simple combining strategy, e.g. In Salehi M. et. al., 2013 [23], the predictions of the two filtering systems are combined linearly (weighted) using the formula:

$$p = \alpha_u * p_{algorithm_1} + (1 - \alpha_u) * p_{algorithm_2}$$

Other examples include Claypool et al. [20], which worked on combining collaborative and content-based filtering through a linear combination for an online newspaper recommender system, and P-tango [12], which gives equal weights for each recommendation technique at first, then weights are adjusted as predictions are confirmed or otherwise. The value of α , in most cases, just only is given a fixed value but after a huge number of tries of different values. This value does not depend on the user or item rated. Another work in Jiali Wang et. al., 2016 [22] remarkably defined and minimized an error function with the use of Lagrange theory and minimum theory to find an optimal value for α . Although the performance of their proposed method is improved, the computation of prediction takes time. (That work in fact reduced execution time by memory-based computing and parallel operations.)

III. OUR CONTRIBUTION

The work aims to improve the quality and accuracy of recommendation and address the new item problem by linearly combining content-based and collaborative filtering techniques. However different from related works mentioned in chapter II,

our contribution is in formulation of the linear weight α that will be calculated from a confidence factor for each item, so that, the calculation of prediction takes more parts from collaborative filtering or content-based filtering depending on such confidence level. The proposed method is also straightforward enough to be implemented in a real application.

IV. PROPOSED METHODOLOGY

A. Preliminaries

We use a consistent mathematical notation for various elements of a recommender system model. Firstly, we consider U as a set of n users, while I is a set of m items. I_u is the set of items rated by user u , and U_i is the set of users who have rated item i . Secondly, the user-item matrix (or utility rating) is denoted by R (or $R^{n \times m}$), with $r_{u,i}$ being the rating of user u provided for the item i . If $r_{u,i}$ is “observed”, it will be associated with a positive integer value, i.e. $r_{u,i} \in \{1, 2, 3, 4, 5\}$, otherwise $r_{u,i}$ is set by zero, implying an “unobserved” rating. We also consider r_u is the vector of all ratings provided by user u , and r_i is the vector of all ratings provided for item i . Besides, $\bar{r}_u$ and $\bar{r}_i$ are the average of a user u 's rating and an item i 's rating respectively. Last but not least, we denote estimated rating of the user u to item i is: $p_{u,i}$.

B. Baseline Collaborative Filtering Candidate

In this work, we suggest to use an item – item collaborative filtering as a Baseline Collaborative Filtering Candidate. Given the utility matrix $R^{n \times m}$, with $r_{u,i}$ being the rating of user u provided for the item i , r_u is the vector of all ratings provided by user u . The process (Algorithm 1) is to calculate the similarities between the target item i and all other items, select the top k similar items, and take the weighted average of ratings from these k items with similarities as weights.

Algorithm 1. ITEM_ITEM_KNN_RECOMMENDER

```

u: user to recommend items
k: number of neighbours used for ranking  $k \in \mathbb{N}$ 
I: list of all items
Iu is the set of items rated by user u
R: utility matrix  $R^{n \times m}$ , with  $r_{u,i}$ : rating of user u provided for the item i
Begin
1. for each item  $i \in I \setminus I_u$  do
1.1. neighbours  $\leftarrow \emptyset$ ,  $p_{u,i} \leftarrow 0$ , denominator  $\leftarrow 0$ 
1.2. for  $j$  in I do
    neighbours[j]  $\leftarrow Sim(i, j)$  (see E.q. 4.1)
1.3. neighbours  $\leftarrow sort(neighbours)[:k]$ 
1.4. for  $j$  in neighbours do
1.4.1.  $p_{u,i} += Sim(i, j) * r_{u,j}$ 
1.4.2. denominator  $+= Sim(i, j)$ 
1.5. end for
1.6.  $p_{u,i} = p_{u,i} / denominator$  (see E.q. 4.2)
2. end for
3. return p
End

```

Traditional similarity calculation methods include: cosine similarity (Resnick et. al., 1994), correlation-based similarity or adjusted cosine similarity (Sarwar et. al., 2001). Adjusted cosine similarity is used in this work, where the user rated average is subtracted from each co-rated pair. This similarity method can work well with a different scale of ratings.

$$Sim(i, j) = \frac{\sum_{u \in U_i \cap U_j} (r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u \in U_i \cap U_j} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{u \in U_j} (r_{u,j} - \bar{r}_u)^2}} \quad (\text{E.q. 4.1})$$

, where $\bar{r}_u$ is the average of the user u 's ratings. The bigger $Sim(i, j)$ is, the more similar the two items are. When $Sim(i, j) = 0$, two items are completely dissimilar and $Sim(i, j) = 1$ means they are completely similar.

The second step is to produce the prediction. Let neighbours $[i_{new}]$ be the set of N -nearest neighbours for the item i_{new} , we calculate the value for the new item as a weighted average of the N -nearest neighbours:

$$p_{u,i_{new}} = \bar{r}_{i_{new}} + \frac{\sum_{j \in \text{neighbours}[i_{new}]} sim(i_{new}, j) * (r_{u,j} - \bar{r}_j)}{\sum_{j \in \text{neighbours}[i_{new}]} |sim(i_{new}, j)|} \quad (\text{E.q. 4.2})$$

C. Baseline Content Based Filtering Candidate

The work concerns content-based recommender systems have dealt with the choice of ridge regression - a linear regression with regularization since it is able to perform well when a small amount of regularization is required for a large number of predictor variables. Considering a user u_n , suppose that each user can have a corresponding model demonstrated by w_n vector and bias b_n , a user's level of interest in an item can be calculated using a linear function as follows:

$$y_{mn} = x_m w_n + b_n \quad (\text{E.q. 4.3})$$

, where x_m is a row vector and w_n is a column vector.

The ridge regression loss function is given by the equation ...

$$L_n = \sum_{m:r_{mn}=1} (x_m w_n + b_n - y_{mn})^2 + \frac{\lambda}{2s_n} \sum w_n^2 \quad (\text{E.q. 4.4})$$

The second part is L2 regularization term, which is added to solve a common problem in machine learning, called overfitting. Besides, λ is a positive value that represents a hyper-parameter on how much to penalize the size of the w_n coefficients, and s_n is number of items that user u has rated. In matrix form, this can be written as:

$$L_n = \frac{1}{2s_n} \sum_{m:r_{mn}=1} \|w_n x_m w_n + b_n - y_{mn}\|_2^2 + \frac{\lambda}{2s_n} \|w_n\|_2^2 \quad (\text{E.q. 4.5})$$

The solution of (w_n, b_n) can be found by using Stochastic Gradient Descent (SGD), or Mini-batch Gradient Descent to minimize the loss function L_n .

$$\underset{(w,b)}{\operatorname{argmin}} \left(\frac{1}{2s_n} \sum_{m:r_{mn}=1} \|w_n x_m w_n + b_n - y_{mn}\|_2^2 + \frac{\lambda}{2s_n} \|w_n\|_2^2 \right) \quad (\text{E.q. 4.6})$$

Algorithm 2 summarizes the process of the baseline model-based content recommender used in this work.

Algorithm 2. MODEL_BASED_CONTENT_RECOMMENDER

```

u: user to recommend items
I: list of all items
I_u is the set of items rated by user u
R: utility matrix  $R^{n \times m}$ , with  $r_{u,i}$ : rating of user u provided for the item i

--TRAINING PHASE--
1. for each user m in training dataset do
  1.1. Construct vector item profile:  $x_m = \text{TF\_IDF}(I_m)$ 
  1.2.  $y_{mn} = r_{mn}$ 
  1.3. Solve the E.q.4.6 achieve  $(w_n, b_n)$ 
2. end for

--PREDICTING PHASE--
1. for each active user u do
  1.1. Construct vector item profile:  $x_u = \text{TF\_IDF}(I_u)$ 
  1.2. for each item i  $\in I \setminus I_u$  do
    Predict rating  $p_{u,i} = x_u w_i + b_i$ 
  1.3. end for
2. end for
3. return p

```

D. Proposed Dynamic Weighted Hybrid Recommender

In this paper, we suggest a novel method of dynamic weighted hybrid recommender that utilize content-metadata and well-known item-based CF methods. The contributions of the proposed recommendation system are depicted in Algorithm 3. A linear combination of different techniques is given by the following formula:

$$p_{u,i} = \sum_f \alpha_f * p_{u,i}^f \quad (\text{E.q. 4.7})$$

, where α_f denotes weight of algorithm f .

Since we are interested in combining 2 recommendation techniques, the computation of prediction score can be written as:

$$p_{u,i} = \alpha_u * p_{u,i}^{\text{CF}} + (1 - \alpha_u) * p_{u,i}^{\text{CBF}} \quad (\text{E.q. 4.8})$$

Different from some related works, our contribution is that we form a simple calculation for α with respect to a confidence factor. Thereby, α will be given by this equation:

$$\alpha_u = \frac{t_u}{k} * 0.9 \quad (\text{E.q. 4.9})$$

, where k : number of neighbours used for ranking $k \in \mathbb{N}$, and t_u is the number of items $\in I_{u_i}$. Note that, in case that t_u is greater than k , set $t_u = k$. We called the ratio between t_u and k as a confidence factor for each user u .

V. IMPLEMENTATION AND EVALUATION

A. Evaluation Index

We can evaluate the accuracy of the model by comparing the estimated ratings with the real ones. One of the most famous evaluation metric to calculate the accuracy of predicted ratings is Root Mean Squared Error (RMSE) [24] in which the lower the RMSE result, the better the predictions are. It is defined as:

$$RMSE = \sqrt{\frac{1}{|T|} \sum_{u,i \in T} (R_{ui} - \hat{r}_{ui})^2}$$

, where R_{ui} and $\hat{r}_{ui}$ is the rating and hybrid result of user u on item i , respectively. $|T|$ denotes the number of $\hat{r}_{ui}$. The RMSE aims to impose a penalty over the larger error, then increases when the errors magnitudes increase. Thus, it is very helpful to compare the performance of different models on the same data (Hahsler, 2014 [25]).

Algorithm 3. DYNAMIC_WEIGHTED_RECOMMENDER

Input

N : number of items to be recommended $N \in \mathbb{N}$
 k : number of neighbours used for ranking $k \in \mathbb{N}$
 I : list of all items
 R : utility matrix $R^{n \times m}$, with r_u is the vector of all ratings provided by user u .

Output

N items to be recommended to user u

Begin

1. **for each** active user u **do**

1.1. Construct vector item profile: $x_u = \text{TF_IDF}(I_u)$

1.2. $\alpha_u = \frac{t_u}{k} * 0.9$ ($t_u = \text{size}(I_u)$, $t_u = k$ if $t_u > k$)

1.3. **for each** item $i \in I \setminus I_u$ **do**

1.3.1. $p_{u,i}^{\text{CF}} \leftarrow \text{ITEM_ITEM_KNN_RECOMMENDER}$

<Algorithm 1>

1.3.2. $p_{u,i}^{\text{CBF}} \leftarrow \text{MODEL_BASED_CONTENT_RECOMMENDER}$

<Algorithm 2>

1.3.3. $p_{u,i} = \alpha_u * p_{u,i}^{\text{CF}} + (1 - \alpha_u) * p_{u,i}^{\text{CBF}}$

1.4. **end for**

2. **end for**

3. **return** p

End

B. Experimental Setup

Dataset. MovieLens is one of the most popular dataset widely used in education, research, and industry. It was collected by GroupLens research group [26] and can be considered as a standard dataset in the domain of movie recommender systems. In order to evaluate the prediction models and validate the proposed recommendation method, a stable benchmark version from MovieLens, named MovieLens 100K Dataset will be used. This dataset was released on April 1998, consisting of 100000 ratings from 943 users on 1682 movies. All selected users have rated at least 20 movies.

There are two main files of the dataset used in our experiments. Firstly, “*u.data*” contains 100000 ratings by 943 users on 1682 items. Each user has rated at least 20 movies. This is a tab separated list of user id, item id, rating, and timestamp. Secondly, “*u.item*” shows information about the movies. This is a tab separated list of movie id, movie title, release date, video release date, IMDB URL, and the last 19 fields representing the genres (Table 1), in which 1 indicates the movie is of that genre, and 0 indicates it is not (movies can be in several genres at once). Besides, no demographic information is used in our below experiments.

TABLE 1. A LIST OF THE GENRES

Index	Genres
0	unknown
1	Action
2	Adventure
3	Animation
4	Children's
5	Comedy
6	Crime
7	Documentary
8	Drama
9	Fantasy
10	Film-Noir
11	Horror
12	Musical
13	Mystery
14	Romance
15	Sci-Fi
16	Thriller
17	War
18	Western

Analysing the dataset in details reveals some remarkable things as follows.

1. The “observed” ratings are on a scale from 1 to 5. The value 0 can represent a missing value, then it is not displayed in the analysis (Figure 3). Most rating values are cantered at 3-4. Average ratings is 3.52986.
2. It is noticeable that just the 17% of the movie ratings are negative (1 or 2) and the 83% remaining gives a relatively positive feedback. This phenomenon can be explained as when a user is watching a movie that they do not like then they commonly just leave without rating the movie.

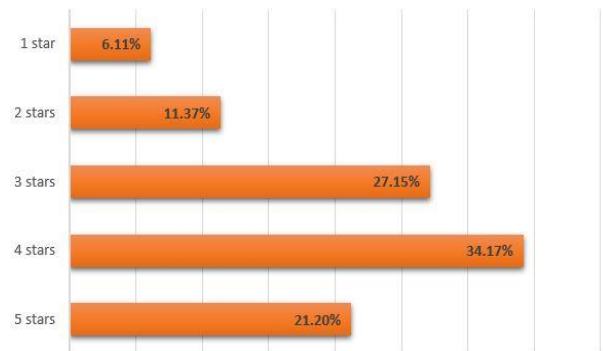


Figure 3. Rates distribution of 100000 ratings from 943 users on 1682 movies

3. The sparsity level of the dataset is high (sparsity = 93.695%) as calculated following the below equation:

$$\text{Sparsity} = 1 - \frac{\text{Number of nonzero entries}}{(\text{Number of Users} \times \text{Number of Movies})}$$

The high sparsity level indicates that most of the values in the utility matrix are unknown as users have not rated the majority of movies. This leads to a long-standing challenge in building recommender systems.

4. Figure 4 depicts the distribution of ratings per movie. As can be seen, the majority of movies (75%) are much less known with 80 ratings or less. While the

most rated movie has 583 ratings, there are 141 movies received only 1 rating in the whole dataset. Since there aren't enough user interactions for a particular item, this MovieLens 100K Dataset, therefore, faces the problem of new item cold start.

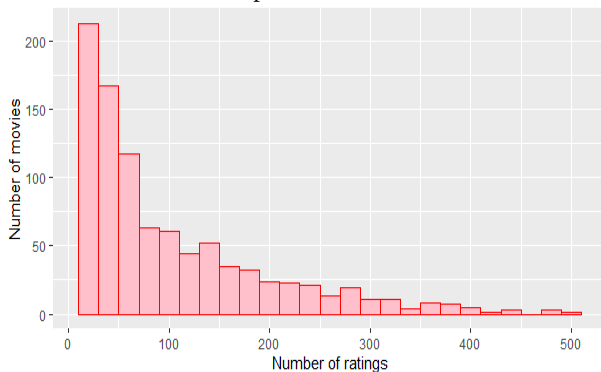


Figure 4. Distribution of ratings per movie

Experiment 1 – Performance comparison

We implement 3 recommendation algorithms: Item-based collaborative filtering (Algorithm 1: ITEM-ITEM-KNN-RECOMMENDER), Content-based filtering (Algorithm 2: MODEL-BASED-CONTENT-RECOMMENDER) and our proposed Dynamic weighed hybrid filtering (Algorithm 3: DYNAMIC-WEIGHTED-RECOMMENDER). We perform them in training sets and test sets to compute predicting ratings. Applying 5-fold cross validation strategy, we divide the dataset into 2 training sets and a test set randomly in which 80% of the data is trained; 20% is tested. Additionally, the value of k , the number of neighbours used for ranking $k \in \mathbb{N}$, is differentiated from 10 to 50.

The experimental results (Table 2) show that the proposed hybrid filtering has lower RSME in most cases of k , meaning a better accuracy. The results in content-based filtering seem not to be affected by the value of k and show the worst performance. Most importantly, it can be seen that our proposed hybrid filtering compares favourably against item-based CF for small values of k . For large values of k , both algorithms will converge since as the k values increase, they tend to recommend the most popular items. However, the RSME from our hybrid filtering is somewhat a bit higher due to the fact that the poor results from content-based filtering contribute negative effect to the hybrid system.

TABLE 2. THE RECOMMENDATION RESULTS OF ABOVE THREE RECOMMENDATION ALGORITHMS

	k= 10	k= 20	k= 30	k= 40	k= 50	
RMSE	0.957	0.953	0.954	0.956	0.957	ITEM_ITEM_KNN_RECOMMENDER
	1.120	1.117	1.120	1.124	1.122	MODEL_BASED_CONTENT_RECOMMENDER
	0.946	0.944	0.949	0.958	0.965	Our proposed DYNAMIC_WEIGHTED_RECOMMENDER

Experiment 2 – Address the new item cold start issue

The quality of collaborative filtering recommendations is significantly affected by the number of ratings due to the fact that they rely on the item's interactions to make recommendations. Therefore, when new items arrive with no or very little interactions, it leads to the problem of item cold start, in which a pure collaborative algorithm cannot recommend the item or the quality of those recommendations will be poor. To validate the effectiveness of our proposed hybrid technique against the collaborative filtering, this conducted experiment, firstly, explore new items that receives no rating previously. There are all 141 movies received only 1 rating in the whole MovieLens 100K Dataset. Since k -cross validation is applied, we selected only items among those 141 items, and also appeared in testing data set. By doing so, we have a list of new items which presented no interactions in training data set. Secondly, we perform the three recommendation techniques: CF, CBF and our proposed hybrid filtering to compute predicting ratings. All cases of k demonstrates a lower RMSE in our algorithm compared to both other filterings (Table 3). Besides, averagely 4.56% lower RMSE compared to CF indicates that by using additional information of item features, our method alleviates the problem of new item cold start.

TABLE 3. THE RECOMMENDATION RESULTS ON 141 MOVIES WHICH RECEIVED ONLY 01 RATING

	k= 10	k= 20	k= 30	k= 40	k= 50	
RMSE	1.195	1.195	1.196	1.197	1.199	ITEM_ITEM_KNN_RECOMMENDER
	1.204	1.247	1.149	1.163	1.205	MODEL_BASED_CONTENT_RECOMMENDER
	1.132	1.135	1.144	1.147	1.151	Our proposed DYNAMIC_WEIGHTED_RECOMMENDER

VI. CONCLUSIONS

The main conclusion reached during this project was the powerful linearly combination of collaborative filtering and content-based filtering. The experimental results on the public datasets MovieLens show that the proposed system has lower Root Mean Squared Error (RSME), meaning a better accuracy. As observed in the described experiments, 4.56% lower RMSE in case of the conduct experiment on items that receive no previous ratings indicates that by using additional information of item features, our method addresses the problem of cold start. In our future work, since another weak point recognised in CF systems is “data sparsity” - Sparser the dataset is, the worse the performance of CF will be [27], we will consider the fusion of such filtering methods if it can alleviate this kind of problem.

REFERENCES

- [1] D. Goldberg, D. Nichols, B.M. Oki, D. Terry. "Using collaborative filtering to weave an information tapestry". Communications of the ACM 35 (1992) 61–70.
- [2] M. Pazzani, J. Muramatsu, D. Billsus. "Syskill and webert: Identifying interesting web sites". In: Proceedings of the Thirteenth National Conference on Artificial Intelligence, vol. 1, AAAI'96, AAAI Press, 1996, pp. 54–61.
- [3] Xiaoyuan Su and Taghi M. Khoshgoftaar. "A Survey of Collaborative Filtering Techniques". Advances in Artificial Intelligence, vol. 2009, Article ID 421425, 19 pages, 2009. <https://doi.org/10.1155/2009/421425>.
- [4] J.S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," Proc. 14th Conf. Uncertainty in Artificial Intelligence, July 1998.
- [5] Goldberg, D., Nichols, D., Oki, B. and Terry, D. (1992). "Using collaborative filtering to weave an information tapestry". Communications of the ACM, 35(12), pp.61-70.
- [6] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. "GroupLens: an open architecture for collaborative filtering of netnews". In Proceedings of the 1994 ACM conference on Computer supported cooperative work (CSCW '94). ACM, New York, NY, USA, 175-186. DOI=<http://dx.doi.org/10.1145/192844.192905>.
- [7] Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L. and Riedl, J. (1997). "GroupLens: applying collaborative filtering to Usenet news". Communications of the ACM, 40(3), pp.77-87.
- [8] Sarwar, Badrul & Karypis, George & Konstan, Joseph & Riedl, John. (2001). "Item-based Collaborative Filtering Recommendation Algorithms". Proceedings of ACM World Wide Web Conference. 1. 10.1145/371920.372071.
- [9] D. Billsus and M. Pazzani, "User Modeling for Adaptive News Access". User Modeling and User-Adapted Interaction, vol. 10, nos. 2–3, pp. 147–180, 2000.
- [10] N. Good, J.B. Schafer, J.A. Konstan, A. Borchers, B. Sarwar, J.L. Herlocker, and J. Riedl, "Combining Collaborative Filtering with Personal Agents for Better Recommendations". Proc. Conf. Am. Assoc. Artificial Intelligence (AAAI-99), pp. 439-446, July 1999.
- [11] Basu C, Hirsh H, Cohen W. "Recommendation as classification: using social and content-based information in recommendation". In: Proceedings of the 15th national conference on artificial intelligence, Madison, WI; 1998, p. 714–20.
- [12] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin, "Combining Content-Based and Collaborative Filters in an Online Newspaper," Proc. ACM SIGIR '99 Workshop Recommender Systems: Algorithms and Evaluation, Aug. 1999.
- [13] M. Pazzani, "A Framework for Collaborative, Content-Based, and Demographic Filtering", Artificial Intelligence Rev., pp. 393-408, Dec. 1999.
- [14] Borràs, J., de la Flor, J., Pérez, Y., Moreno, A., Valls, A., Isern, D., et al. (2011). "SigTur/Edestination: A system for the management of complex tourist regions". In R. Law, M. Fuchs, & F. Ricci (Eds.), Information and communication technologies in tourism 2011 (pp. 39–50). Springer.
- [15] Lucas, J., Luz, N., Moreno, M., Anacleto, R., Figueiredo, A., & Martins, C. (2013). "A hybrid recommendation approach for a tourism system". Expert Systems with Applications, 40(9), 3532–3550.
- [16] Joan Borràs, Antonio Moreno, Aida Valls. "Intelligent tourism recommender systems: A survey". Expert Systems with Applications, Volume 41, Issue 16, 2014, Pages 7370-7389.
- [17] Linden Gregory D and Jacobi Jennifer A and Benson Eric A. "Collaborative Recommendations Using Item-to-item Similarity Mappings". Granted Patent, US 6266649 B1, 2001-07-24.
- [18] Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: "Item-based Collaborative Filtering Recommendation Algorithms". In: Proc. of the 10th Int. Conf. on World Wide Web (2001)
- [19] Deshpande, M., Karypis, G.: "Item-based top-N recommendation algorithms". ACM Transactions on Information Systems 22, 143–177 (2004).
- [20] Claypool M, Gokhale A, Miranda T, Murnikov P, Netes D, and Sartin M, 1999. "Combining content-based and collaborative filters in an online newspaper". SIGIR99: Proceedings of the ACM SIGIR Workshop on Recommender Systems.
- [21] Liang, Xijun, Zhonghang Xia, Liping Pang, Liwei Zhang, Hongwei Zhang (2016). "Measure prediction capability of data for collaborative filtering". In: Knowledge and Information Systems 49.3, Springer Verlag London 2016; pp. 975–1004.
- [22] Jiali Wang, Hang Zhuang, Changlong Li, Hang Chen, Bo Xu, Zhuocheng He, Xuehai Zhou. "A Fast and Better Hybrid Recommender System Based on Spark". Network and Parallel Computing. 13th IFIP WG 10.3 International Conference, NPC 2016 Xi'an, China, October 28–29, 2016.
- [23] Salehi, M., Pourzaferani, M., Razavi, S.A. "Hybrid attribute-based recommender system for learning material using genetic algorithm and a multidimensional information model". Egyptian Informatics Journal (2013).
- [24] Asela Gunawardana, Guy Shani. "A Survey of Accuracy Evaluation Metrics of Recommendation Tasks". Journal of Machine Learning Research 10 (2009) 2935-2962.
- [25] Hahsler, Michael (2014)". *Recommenderlab: Lab for Developing and Testing Recommender Algorithms*". R package version 0.1-5. URL:<http://CRAN.R-project.org/package=recommenderlab>.
- [26] F. Maxwell Harper, F. Maxwell Harper. "The MovieLens Datasets: History and Context". ACM Transactions on Interactive Intelligent Systems (TiIS) - Regular Articles and Special issue on New Directions in Eye Gaze for Interactive Intelligent Systems (Part 1 of 2) archive. Volume 5 Issue 4, January 2016.
- [27] Grčar M., Mladenčić D., Fortuna B., Grobelnik M. (2006). "Data Sparsity Issues in the Collaborative Filtering Framework". Advances in Web Mining and Web Usage Analysis. WebKDD 2005, pp.58-76.



related to E-government, and E-Commerce Recommendation applications.



Summer 2018. His research interests are in the area of recommender systems.



Byeongnam Yoon (M'97) He became a Member of IEEE in 1997. He was born in Seoul Korea 1949. He got the PhD in computer science, Chungnam National University, Korea, 1997. He worked for the Sperryrand UNIVAC as a Computer Specialist 1974 -1978, Samsung as a Manager of Telecommunications Section 1978 - 1982, Electronics & Telecommunications Research Institute (ETRI) as a Principal Researcher 1982 – 1999, National Information society Agency (NIA) as a Senior Executive Director General 1999 – 2010, Kyonggi University (KGU) as an Associate Professor, Faculty of Computer Science 2010 - 2016. Global IT Research Institute GIRI) as a President 1999 – current. His research area include a Telecommunications, Internet, Software, Web programming & security, e-Government, Enterprise Architecture, Work Flow, Information Control Nets. System Work Method. SPiCE, CMMI, BPM, etc.