An Improved Dynamic Collaborative Filtering Algorithm Based on LDA

MENG DI-FEI¹, LIU NA¹, LI MING-XIA¹,²,³, AND SU HAO-LONG¹

¹School of Information Science and Engineering, Dalian Polytechnic University, Dalian 116001, China
²School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China
³Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang 110168, China

Corresponding author: Liu Na (liuna@dlpu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61402069, in part by the National Key Research and Development Plan under Grant 2017YFC0821003, and in part by the Natural Science Foundation of Liaoning Province under Grant 20180550395.

ABSTRACT Currently, available collaborative filtering (CF) algorithms often utilize user behavior data to generate recommendations. The similarity calculation between users is mostly based on the scores, without considering the explicit attributes of the users with profiles, as these are difficult to generate, or their preferences over time evolve. This paper proposes a collaborative filtering algorithm named hybrid dynamic collaborative filtering (HDCF), which is based on the topic model. Considering that the user’s evaluation of an item will change over time, we add a time-decay function to the subject model and give its variational inference model. In the collaborative filtering score, we generate a hybrid score for similarity calculation with the topic model. The experimental results show that this algorithm has better performance than currently available algorithms on the MovieLens dataset, Netflix dataset and la.fm dataset.

INDEX TERMS Collaborative filtering, LDA, topic model, time tag.

I. INTRODUCTION

With the advent of the Internet, the availability of information continues to rapidly increase. This abundance makes it challenging for users to effectively find the information they need in the high volume of data they can access. To address this problem, recommendation systems have been proposed to help users quickly obtain useful information depending on their past preferences or other sources.

Most established recommendation algorithms start with finding a set of customers. This set of customers has purchased and rated products that overlap with the products that the current user has purchased and rated. The algorithm aggregates the products from these similar customers, excludes the products that the user has purchased or rated, and recommends the remaining products to the user.

The purpose of collaborative filtering (CF) is to suggest new items or to predict the utility of a certain item for a particular user based on the user’s previous preferences and the opinions of other like-minded users. Researchers have proposed many CF algorithms that can be divided into two main categories: user-based and item-based. In this research, we focus on user-based methods.

In recent years, an increasing number of studies have used topic models and review texts to generate recommendations. In the recommendation algorithm, to find the nearest neighbor of the target user, we need to measure the similarity between users. However, the number of users and items in the recommendation system is very large, users often only score a small number of items, and the user rating data are extremely sparse, which makes the nearest neighbor set obtained by the traditional similarity measurement method not accurate enough, resulting in a reduction in the recommendation quality of the algorithm. In addition, the time factor is very important context information in recommendation systems. Users’ interests vary greatly in different periods of time. The topics in a document evolve over time. We compare the inaugural addresses of two presidents in different periods and present them in the form of word clouds in Figure 1. The higher the word frequency, the larger the display size.

It can be seen in the figure that although they were all inaugural addresses, they used different words, and the use of high-frequency words was also quite different. Therefore, when the time span of the corpus is large, the topic model obviously needs to consider the time factor and the dynamic change in language. However, most methods ignore the time factor. Therefore, we propose a hybrid dynamic collaborative filtering algorithm (HDCF) that can capture the evolution of topics in a collaborative filtering algorithm.
Equation (1) the traditional CF algorithm assigns the same weight to all items of interest, ignoring the influence of user time on interest when calculating the similarity. In fact, the user’s attention to the item will change over time, and the attention will affect the user’s interest. Therefore, the algorithm should consider the time dimension. Aiming at the above problems, this paper introduces a time-decay function in the LDA model that gives different weights to items according to the time users look at them. The corpus is divided into time slices according to the time attribute of the document. The distribution of current topic words is determined by the distribution of previous time, the time-decay rate and the weight of words. We also give the approximate variational posterior of this model.

Equation (2) in the processing of similarity computation, we calculate a hybrid user similarity score, that is, the integration of the topic model and traditional similarity.

In this way, our approach is different from the established recommendation algorithms to improve the recommendation quality.

This paper is organized as follows. Section II describes related work, including CF algorithms and topic models in the literature. Section III describes the proposed algorithm, and Section IV presents the results of applying this algorithm to different datasets. In Section V, we conclude and discuss further research directions.

II. RELATED WORK

Recommender systems are often based on CF, which relies only on past user behavior. For example, their previous transactions or product ratings do not require the creation of explicit profiles. Notably, CF techniques do not require domain knowledge and avoid the need for extensive data collection. In addition, relying directly on user behavior allows the discovery of complex and unexpected patterns that would be difficult or impossible to profile using known data attributes. As a consequence, CF has attracted much attention in the past decade, resulting in significant progress and adoption by some successful commercial systems.

Along with the deep study of recommendation systems, researchers introduced the context environment into recommendation systems, such as time, location, mood, active state, and network condition. Traditional CF techniques cannot track users’ preferences over a period of time [1]–[9]. Therefore, temporal dynamics emerged in recommendation systems. Considering the time interval between purchases, Wang et al. proposed an opportunity model. The model determines the items to be recommended and the best time to recommend a specific product [10]. Mustansar Ali Ghazanfar et al. presented a novel structure learning technique called the kernel-mapping recommender (KMR) to make reliable recommendations under sparse, cold-start, and long-tail scenarios. In 2011, they presented a fast incremental algorithm for building the model. In 2012, they constructed user-based and item-based KMR, combining user-based and item-based KMR and kernels from feature information [11], [12]. Cheng JiuJun proposed a user spatiotemporal behavior pattern method combining mobile personalized attributes and context information. Considering the popularity of users’ interests, the popularity of subjects and the impact of users, they built a model based on users’ interests [13]. Nour EI Islam Karabadij considered the recommended range when selecting the appropriate group. This helped to make accurate and diverse recommendations simultaneously [14]. Niu Z proposed a new knowledge-based topic model that combined the Dirichlet tree and integrated the must-links into topic modeling for object discovery. In particular, to better deal with the polysemy phenomenon of visual vocabulary, a must-link was redefined such that it only constrained one or more specific topics rather than all topics, thus significantly improving topic coherence [15]. Zhang Xiong proposed a method that further improved recommendation accuracy. Within a semantic environment in a document, entities of different topics were present. Another entity that
appeared in the same document at the same time was used to help disambiguate the referenced content to a certain extent [16], Zhang and Liu presented a cross-domain recommender system based on kernel-induced knowledge transfer. This method effectively transfers knowledge through overlapping entities and alleviates data sparsity issues [17], [18]. Das J proposed a CF method based on clustering, which used two kinds of hierarchical space to divide the data structure: K-d tree and quadtree. They clustered or partitioned the user space according to the user’s location and then used the generated clustering results to predict the target user’s score [19]. Zhang, Pengfei, et al. proposed decomposing the matrix into two nonnegative matrices and then integrating the time weighting into the evaluation matrix in the collaborative filtering algorithm [20]. Yu X et al. proposed two auxiliary domains, i.e., user-side domain and item-side domain, to solve the sparsity problem. In this method, both the user side and item side can not only share information but also infer domain-independent user and item features. They also proposed another cross-domain collaborative filtering algorithm to alleviate the sparsity problem. This method first formulates the recommendation problem as a classification problem in the target domain. Then, Funk-SVD decomposition is used to extract extra user and item features. Finally, the C4.5 decision tree algorithm is used to predict missing ratings [21], [22]. Wang joined LDA and a listwise model to generate collaborative filtering results [23]. Zhang used a multichannel feature vector to calculate the similarity between items [24].

III. PROPOSED APPROACH

The basic idea of CF algorithms is to provide item recommendations or predictions based on the opinions of other like-minded users. The opinions of users can be obtained explicitly from the users or by using implicit measures. In a typical CF scenario, there is a list of m users U={u1,u2,…,um} and a list of n items I={i1,i2,…,in}. Each user u has a list of items rui, which the user has expressed their opinions about. CF algorithms represent the entire m×n user-item data as a ratings matrix. Each entry auij represents the preference score (ratings) of the i-th user on the j-th item. Each individual rating is within a numerical scale; the value zero indicates that the user has not yet rated the item.

A key step in CF algorithms is to calculate the similarity between users. It is typically based on a cosine approach or a correlation in similarity computation. However, they cannot find deeper relations between words. A topic model is another option. In this section, we first introduce the basic idea of LDA, then introduce our proposed method, give its variational inference model, then introduce the similarity computation of the HDCF model, and finally the collaborative filtering prediction model.

A. UNDERLYING EXISTING LDA

LDA is an unsupervised machine learning technique that can be used to identify potentially hidden topic information in a large-scale corpus. This method assumes that each word is extracted from a hidden topic.

The LDA model can be expressed as a probability graph model, as shown in Figure 2. The shadow circle in the figure represents the observed variable, the nonshadow circle represents the potential variable, the arrow represents the conditional dependency between the two variables, the box represents the replicate sampling, and the number of replicates is in the lower right corner of the box.

The LDA assumes that the prior distribution of the document topics is the Dirichlet distribution. That is, for each document d, its topic distribution θd is

\[ \theta_d = \text{Dirichlet}(\alpha) \]  

(1)

where α is the proportion parameters.

The LDA assumes that the prior word distribution in a topic is a Dirichlet distribution. That is, for each topic k, its word distribution ϕ is:

\[ \phi_d = \text{Dirichlet}(\beta) \]  

(2)

where β is a vector of the topic parameters.

For each word Wd,n in document d, we can obtain its topic number Zdn distribution from the topic distribution θd as follows:

\[ Z_{dn} = \text{multi}(\theta_d) \]  

(3)

For this topic number, we obtain the probability distribution of word Wd,n as follows:

\[ w_{d,n} = \text{multi}(\phi_{z_{dn}}) \]  

(4)

A k-dimensional Dirichlet random variable θ can take values in the (k − 1)-simplex. It can be expressed as the following probability density on this simplex:

\[ p(\theta | \alpha) = \frac{\Gamma \left( \sum_{i=1}^{k} \alpha_i \right)}{\prod_{i=1}^{k} \Gamma (\alpha_i)} \theta_1^{\alpha_1-1} \cdots \theta_k^{\alpha_k-1} \]  

(5)

where the parameters α1, α2,…,αk > 0 and Γ(x) are the gamma functions.

Given the parameters α and β, the joint distribution of a topic mixture is:

\[ p(\theta, z, w | \alpha, \beta) = p(\theta | \alpha) \prod_{n=1}^{N} p(z_n | \theta) p(w_n | z_n, \beta) \]  

(6)

Integrating over θg and summing over z, the distribution of a document is:

\[ p(w | \alpha, \beta) = \int p(\theta | \alpha) \left( \prod_{n=1}^{N} p(z_n | \theta) p(w_n | z_n, \beta) \right) d\theta \]  

(7)
Taking the product of the probabilities of single documents, the probability of a corpus is:

\[
p(D | \alpha, \beta) = \prod_{d=1}^{M} p(\theta_d | \alpha) \\
\times \left( \prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn} | \theta_d) p(w_{dn} | z_{dn}, \beta) \right) d\theta_d
\]

(8)

Each topic \( z \) corresponds to a distribution over an item, and each user has a distribution \( \theta \) on a potential topic. The approach for rating items can be described as follows: the users choose a topic based on their interests and then choose an item according to the distribution over this topic. More specifically, the process is as follows:

1. For each document \( d \), choose \( \theta \sim \text{Dir}(\alpha) \).
2. For each word \( w \) for document \( d \):
   a. Choose a topic \( z_n \sim \text{multinomial}(\theta) \).
   b. Choose a word \( w_n \) from \( p(w_n|z_n, \beta) \), a multinomial probability conditioned on the topic \( z_n \).

### B. GENERATING THE HDCF ALGORITHM

Most current topic models are based on the assumption that documents in a corpus are not sequential; in other words, documents in a corpus are interchangeable. In fact, this simplified assumption is inappropriate and inconsistent with the actual situation. 1) A large number of corpora, such as scientific and technological literature databases and news databases, are temporal corpora, and the documents in them have the time attribute. Some specific text information can only appear in a certain time period. In addition, many corpora span hundreds of years; for these corpora, it is obviously inappropriate to ignore the time order attribute. 2) Language changes over time, and the topic is bound to evolve over time.

The traditional CF algorithm assigns the same weight to all items of interest, ignoring the influence of user time on interest when calculating the similarity. This is obviously unreasonable. Because the attention of users to items will change with the retention time of items, the attention affects the interest of users. Therefore, the algorithm should consider the time dimension to improve the accuracy of the recommendation prediction. Generally, the recently followed items with high attention of users show their recent interests and hobbies, so users have a high degree of interest in such items, whereas users have a low degree of interest in items.

Aiming at the above problems, this paper introduces the HDCF model on the basis of reference [25] and gives different weights to items according to the time users look at the items. Based on the weight of the time attribute, the changes in users’ interests can be reflected to realize the time-based similarity calculation of interests. The shorter the time for users to consume items, the higher their corresponding interests will be, which should have a higher weight, and vice versa. The difference between our model and reference [25] lies in: The process of HDCF algorithm is divided into 4 phases as shown in Figure 3. This method not only uses time-delay function, but also gives the derivation process of Variational Inference in phase 2. The nearest neighbor set is used instead of all data sets, so as to reduce the sparsity of data in phase 3. To verify the reliability and accuracy of the algorithm, the number of baselines is increased in the experiment, and the experimental results std. deviation (with error bars) are given.

According to the time attribute of the document, the corpus is divided into \( T \) time slices. Time slice 1 is the oldest time slice, time slice \( T \) is the latest time slice, and a subset of documents on each time slice is \( D = \{D_1, D_2, \ldots, D_T\} \). Introducing the weight of time attribute \( r(t) > 0 \). Let the last browsing time of the topic be \( t_i \) the current time be \( t \), and the distribution of current topic words is determined by the distribution of previous times, the time-decay rate and the weight of words.

\[
\varphi_t = \varphi_{t-1} \cdot \omega \cdot r(t) \\
\omega = \frac{W_t}{W_{t-1}} \\
r(t) = \frac{1}{1 + e^{\alpha(t_i - t)}}
\]

(9)

where \( W(t) \) is the number of words of time slice \( t \) and \( r(t) \) is the rate of time decay.
\[ Z_{t,N} \]

\[ \Phi_{t,i} \]

\[ \beta_{t-1} \sim N(\beta_t, \sigma^2 I) \]

\[ q(\beta, z, \theta | \lambda, \phi, \gamma) \]

\[ = q(\beta_1, \beta_{t,1}, \theta_1, \theta_t, \phi_{t}^d, \gamma_{t}^d) \]

\[ = \prod_{k=1}^{K} q(\beta_{t} | \beta_{t-1}) \prod_{d=1}^{D_t} q(\theta_{t}^{d} | \theta_{t-1}^{d}, \gamma_{t}^{d}, \phi_{t}^{d}) \]

\[ = \prod_{k=1}^{K} q(\beta_{t} | \beta_{t-1}) \prod_{d=1}^{D_t} \left( q(\theta_{t}^{d} | \theta_{t-1}^{d}, \gamma_{t}^{d}, \phi_{t}^{d}) \prod_{n=1}^{N_d} q(\phi_{n,t}^{d} | \phi_{n,t-1}^{d}) \right) \]

\[ \times \prod_{t=1}^{T} \left( \prod_{d=1}^{D_t} \left( q(\theta_{t}^{d} | \theta_{t-1}^{d}, \gamma_{t}^{d}, \phi_{t}^{d}) \prod_{n=1}^{N_d} q(\phi_{n,t}^{d} | \phi_{n,t-1}^{d}) \right) \right) \] (12)

The rest of the iterative and updating process of our model is very similar to that of the traditional LDA, but the two are obviously different. The distribution of topics in our method is related to the degree of time decay, which makes the iteration and updating process on each document associated with the previous slice process.

\section*{D. SIMILARITY COMPUTATION OF HDCF MODEL}

An important step in CF algorithms is to calculate the similarity between users. This result is used to establish a proximity-based neighborhood between the target user and some similar users. The main goal of neighborhood formation is to find an ordered list of \( I \) users \( N = \{n_1, n_2, \ldots n_I\} \) for each user \( u \). For example, \( sim(u, N_1) \) is maximum, \( sim(u, N_2) \) is the next maximum, and so on. The proximity between two users is usually measured by a cosine-based similarity. However, both the cosine and correlation approaches utilize a bag of words. They cannot mind the semantic relationships between words.

In our method, we first define the matching degree \( \delta_t \) between document \( d \) and topic over word \( w_{n,t} \) on time slice \( t \) as

\[ \delta_t = \theta_{t}^{d} \cdot Z_{n,t}^{d} \]

\[ = \sum_{j=1}^{W_t} Z_{n,t}^{d} Z_{j,t}^{d} \]

\[ \frac{Z_{n,t}^{d}}{W_t} \]

(13)

where \( Z_{n,t}^{d} \) is the distribution of words on topic and \( W_t \) is the number of words on time slice \( t \).

Then, the \( r_{ij} \) of document \( d_i \) for word \( w_j \) in the CF algorithm is modified by

\[ r_{ij} = \delta_t \cdot \varphi_{ij} \]

(14)

Then, we regard the topic distribution as the rating of the document-word matrix and use the KL (Kullback-Leibler)
divergence to evaluate the similarity of users.

\[
\text{sim}_{\text{DA}} = \exp(-KL(u, u_j))
\]

\[
KL(u_i, u_j) = KL(u_i||u_j) + KL(u_j||u_i)
\]

\[
= \sum_{k \in m} \left( \frac{\theta_i}{\theta_k} \right) \cdot \theta_i + \sum_{k \in m} \left( \frac{\theta_j}{\theta_k} \right) \cdot \theta_j
\]

Considering the advantages of traditional similarity and topic similarity, we use a hybrid similarity for words in this paper.

\[
sim_{ij} = \sum_{u \in U} \frac{r_{ui} \cdot r_{uj}}{\sqrt{\sum_{u \in U} r_{ui}^2 \cdot \sum_{u \in U} r_{uj}^2}}
\]

\[
sim_{ac} = \sum_{u \in U} (r_{ui} - \bar{r}_u)(r_{uj} - \bar{r}_u)
\]

\[
sim_{ij}^p = \sum_{u \in U} (r_{ui} - \bar{r}_i)(r_{uj} - \bar{r}_j)
\]

\[
sim_{ij} = \lambda \left( \frac{1}{3} (sim_{ij}^c + sim_{ij}^p + sim_{ij}^{ac}) \right) + (1 - \lambda) \text{sim}_{ij}^{\text{DA}}
\]  

(16)

where the similarity \(sim_{ij}^c\) is calculated by the cosine measurement, the adjusted cosine similarity \(sim_{ij}^p\) improves the above limitation by subtracting the user’s average score, correlation-based similarity \(sim_{ij}^{ac}\) measures similarity with the Pearson correlation coefficient, \(\bar{r}_j\) is the average of \(ri\), \(\lambda\) is adjusting parameter.

E. PREDICTION COMPUTATION OF HDCF MODEL

Another important step in the CF algorithm is to generate recommendations based on predictions. After calculating the similarity between the items, the user can predict the unrated items based on their similarity.

To solve the problem of data sparsity, we use the nearest target users’ neighbors method for prediction [27]. Setting \(NBS_u\) as the nearest neighbor for user \(u\), the prediction \(\hat{r}_{ui}\) of user \(u\) for item \(i\) are obtained by the ratings of user \(u\) for the nearest neighbors set \(NBS_u\).

\[
r_{ui} = \bar{r}_u + \frac{\sum_{n \in NBS_u} \text{sim}_{u,n} \times (r_{n,i} - \bar{r}_n)}{\sum_{n \in NBS_u} |\text{sim}_{u,n}|}
\]  

(17)

where \(\text{sim}(u,n)\) is the similarity between user \(u\) and user \(n\), \(r_{n,i}\) is the rating of user \(n\) for item \(i\), and \(\bar{r}_n\) is the average rating of user \(u\) and user \(n\).

This method calculates the prediction on user \(u\) for item \(i\) by computing the sum of the user’s ratings for items that are similar to \(i\). For the influence of context time, a rating the user made long ago should be less influential than a rating made now. To accomplish this, a time delay function \(f_{u,i}\) of active user \(u\) for tag \(t_i\) is used in the prediction computation.

In this method, the prediction on user \(u\) for item \(i\) is calculated by the average of the user’s ratings. The ratings are weighted by the similarity between items, which is similar to item \(i\). However, the recently followed items with high user attention show their recent interests and hobbies, so users have a high degree of interest in such items, whereas users have a low degree of interest in items. Therefore, considering the time influence on the prediction results, the time-decay function is introduced into the prediction process in this paper.

\[
f_{u,i} = \sum_{j \in M_i} \text{sim}_{i,j} \cdot \exp\left(-\frac{|t_{u,j} - t_{u,i}|}{\bar{t}_u}\right)
\]  

(18)

Recommendation prediction is modified as

\[
r_{ui} = \frac{\hat{r}_{u,i} \cdot r_{ui}}{\sum_{j \in M_i} f_{u,j} |r_{u,j}|}
\]  

(19)

IV. EXPERIMENTS

Our experiment is divided into two parts. The first part determines all kinds of parameters required by the algorithm, including the number of topics, neighbor size, and hybrid similarity parameter. We compare performance with other traditional methods in the second part.

A. DATASET

We use the MovieLens dataset for our first part of the experiments. This dataset consists of 100,000 ratings (1-5) for 943 users of 1,682 movies. We use 80% of the dataset as the training set and 20% as the test set. In addition to the MovieLens dataset, we also use the Netflix dataset and last.fm dataset in the comparison experiments.

B. EVALUATION METRICS

There are numerous measures available to evaluate the recommendation quality. We use the mean absolute error (MAE) and the root-mean-square error (RMSE) to evaluate our algorithm. They are common measures in recommender systems. MAE is an average of the absolute errors between predictions and eventual outcomes, while RMSE is the square root of the average of the square of all the errors. MAE and RMSE are given by

\[
\text{MAE} = \frac{1}{n} \sum_{u \in U} |r_{u,i} - \hat{r}_{u,i}|
\]  

(20)

\[
\text{RMSE} = \sqrt{\frac{1}{n} \sum_{u \in U} (r_{u,i} - \hat{r}_{u,i})^2}
\]  

(21)

Obviously, the lower the value of MAE or RMSE is, the higher the accuracy of the recommended results.

C. PARAMETER OPTIMIZATION EXPERIMENTS

We started our experiments by dividing the dataset into training and test sections (80% for training and 20% for testing). Then, we measured the sensitivity of some parameters before
TABLE 1. Description of first 5 distribution.

| TOPIC1          | TOPIC2          | TOPIC3          | TOPIC4          | TOPIC5          |
|-----------------|-----------------|-----------------|-----------------|-----------------|
| Beauty and the Beast (1991) | Fargo (1996)     | Mission: Impossible (1996) | Titanic (1997) | Monty Python and the Holy Grail (1974) |
| 0.02240325      | 0.025057762     | 0.025577339     | 0.052849563    | 0.023877744     |
| Alice in Wonderland (1951) | Frighteners, The (1996) | Independence Day (ID4) (1996) | Good Will Hunting (1997) | Sleepless in Seattle (1993) |
| 0.02055515      | 0.024494687     | 0.02316641      | 0.046646279    | 0.023290162     |
| Mary Poppins (1964) | Beavis and Butt-head Do America (1996) | Rock, The (1996) | Apt Pupil (1998) | Young Frankenstein (1974) |
| 0.018112542     | 0.024025458     | 0.022799529     | 0.03609264     | 0.022362829     |
| Fantasia (1940)  | Heat (1995)      | Eraser (1996)    | L.A. Confidential (1997) | Mrs. Doubtfire (1993) |
| 0.017270137     | 0.023931613     | 0.021803711     | 0.031762285    | 0.022435497     |
| Mask, The (1994) | Twelve Monkeys (1995) | Twister (1996)   | Conspiracy Theory (1997) | Groundhog Day (1993) |
| 0.016764694     | 0.022617772     | 0.021174772     | 0.030936664    | 0.021527415     |

There are three main parameters in the LDA: $\alpha$, $\beta$, and $K$. Parameter $\alpha$ denotes the distribution of topics for each document, $\beta$ denotes the distribution of words for each topic, and $K$ denotes the number of topics. We set $\alpha = 0.1$ and $\beta = 50/K$ in our experiment. We set the number of topics $K$ to range from 10 to 100, incrementing by 10 each time. The MAE results are shown in Figure 5.

It is clear in Figure 5 that MAE performance fluctuates slightly with different numbers of topics. The minimum value of MAE occurs when $K=50$. As a result, we set the number of topics $K=50$ in the following experiment.

We also give the first five distributions of topics and words over the topics in TABLE 1 when the number of topics running the main experiment. These parameters include the number of topics $K$, neighborhood size, number of clusters, and the hybrid user similarity parameter. Based on the results, we fixed the optimum values of these parameters and used them for the rest of the experiments.
k = 20. It is interesting that even though the topic is latent and there is no direct real-world explanation, the top item corresponding to each topic allows us to have some form of interpretation of the item found by the model. The first line is the first 5 topics with their probability. The remaining line is the word distribution over the corresponding topic. The topic name and the word is in bold.

The neighborhood size and the number of clusters have a significant influence on the prediction quality. To determine these parameters, we change the neighborhood size from 10 to 160 and the cluster number from 5 to 50. Figure 4 depicts our experimental results. The neighborhood size and the number of clusters affect the prediction quality.

As shown in Figure 6, the MAE is lower when the number of clusters is larger. Therefore, we choose 50 clusters to compare our proposed algorithm with the other algorithms. Our proposed algorithm uses the topic model and some traditional similarity approach to calculate a hybrid user interaction model. The results are shown in Table 2 to Table 5.

### Table 2. Performance results with MAE on movielen dataset.

| Method  | Neighbor Number |
|---------|-----------------|
|         | 10   | 20   | 30   | 40   | 50   | 60   | 80   | 100  | 130  | 160  |
| CF-cos  | 0.8741| 0.8903| 0.8923| 0.8572| 0.8941| 0.8862| 0.8825| 0.8893| 0.8690| 0.9033|
| CF-pos  | 0.8840| 0.8534| 0.8594| 0.8669| 0.8708| 0.8681| 0.8726| 0.8721| 0.8889| 0.8599|
| CF-ac   | 0.8984| 0.8903| 0.8788| 0.8669| 0.8786| 0.8862| 0.8811| 0.8784| 0.8499| 0.8652|
| CF-LDA  | 0.7697| 0.7641| 0.7598| 0.7569| 0.7547| 0.7530| 0.7517| 0.7509| 0.7500| 0.7495|
| CF-MCFV | 0.8010| 0.8420| 0.8440| 0.8380| 0.8360| 0.8370| 0.8340| 0.8350| 0.8210| 0.8070|
| ROST    | 0.7507| 0.7533| 0.7402| 0.7503| 0.7469| 0.7417| 0.7315| 0.7562| 0.7430| 0.7432|
| HDCF    | 0.7559| 0.7228| 0.7427| 0.7134| 0.7199| 0.7015| 0.6817| 0.6718| 0.6539| 0.6923|

### Table 3. Performance results with MAE on netflix dataset.

| Method  | Neighbor Number |
|---------|-----------------|
|         | 10   | 20   | 30   | 40   | 50   | 60   | 80   | 100  | 130  | 160  |
| CF-cos  | 0.8722| 0.9643| 0.9588| 0.9552| 0.9746| 0.9753| 0.9391| 0.9610| 1.0027| 0.9296|
| CF-pos  | 0.8799| 0.8119| 0.7851| 0.8080| 0.8239| 0.8203| 0.8622| 0.8403| 0.9102| 0.8593|
| CF-ac   | 0.8707| 0.9453| 0.9069| 0.9326| 0.8816| 0.9427| 0.9246| 0.9407| 0.9017| 0.9598|
| CF-LDA  | 0.7583| 0.8093| 0.7799| 0.7968| 0.8400| 0.7610| 0.8101| 0.8139| 0.8594| 0.7931|
| CF-MCFV | 0.8676| 0.8665| 0.8128| 0.7762| 0.8140| 0.8248| 0.7782| 0.8645| 0.8526| 0.8763|
| HDCF    | 0.7874| 0.7873| 0.7730| 0.7807| 0.7402| 0.7596| 0.7548| 0.7141| 0.7763| 0.7363|

### Table 4. Performance results with MAE on fa.lm dataset.

| Method  | Neighbor Number |
|---------|-----------------|
|         | 10   | 20   | 30   | 40   | 50   | 60   | 80   | 100  | 130  | 160  |
| CF-cos  | 0.8840| 0.7821| 0.8037| 0.8439| 0.8080| 0.8373| 0.7986| 0.8492| 0.8459| 0.7969|
| CF-pos  | 0.8857| 0.7923| 0.7857| 0.7922| 0.8010| 0.8054| 0.7831| 0.8047| 0.8006| 0.8223|
| CF-ac   | 0.8090| 0.7926| 0.8013| 0.8515| 0.8645| 0.8044| 0.7837| 0.8599| 0.8626| 0.8995|
| CF-LDA  | 0.7888| 0.8160| 0.8370| 0.8210| 0.8023| 0.7792| 0.8082| 0.7904| 0.7940| 0.7813|
| CF-MCFV | 0.6940| 0.7046| 0.7808| 0.7856| 0.7857| 0.8014| 0.7964| 0.8187| 0.8058| 0.7599|
| HDCF    | 0.8142| 0.7641| 0.8241| 0.7101| 0.8152| 0.7372| 0.7798| 0.6963| 0.8283| 0.7905|

### Table 5. Performance results with RMSE on movielen dataset.

| Method  | Neighbor Number |
|---------|-----------------|
|         | 10   | 20   | 30   | 40   | 50   | 60   | 80   | 100  | 130  | 160  |
| CF-cos  | 1.3000| 1.3000| 1.2870| 1.2610| 1.3330| 1.2630| 1.2230| 1.3330| 1.2070| 1.3270|
| CF-pos  | 1.2610| 1.1910| 1.2400| 1.2870| 1.2390| 1.2520| 1.1890| 1.3230| 1.2630| 1.2010|
| CF-ac   | 1.2640| 1.3710| 1.3240| 1.2630| 1.2930| 1.3250| 1.2870| 1.3350| 1.2030| 1.2850|
| CF-LDA  | 1.1340| 1.1090| 1.2040| 1.1660| 1.1280| 1.1320| 1.1620| 1.1840| 1.2050| 1.1250|
| CF-MCFV | 1.2200| 1.2250| 1.2180| 1.2160| 1.2080| 1.2120| 1.1890| 1.1600| 1.1650| 1.1400|
| ROST    | 1.203 | 1.1981| 1.213 | 1.1690| 1.1210| 1.2320| 1.1802| 1.1604| 1.2351| 1.1050|
| HDCF    | 1.0980| 1.1210| 1.1860| 1.1570| 1.2830| 1.1020| 1.1620| 1.1240| 1.1110| 1.0610|

As shown in Figure 6, the MAE is lower when the number of clusters is larger. Therefore, we choose 50 clusters to compare our proposed algorithm with the other algorithms. Our proposed algorithm uses the topic model and some traditional similarity approach to calculate a hybrid user interaction model.
M. D.-Fei et al.: Improved Dynamic Collaborative Filtering Algorithm Based on LDA

TABLE 6. Performance results with RMSE on netflix dataset.

| Method  | Neighbor Number |
|---------|-----------------|
|         | 10   | 20   | 30   | 40   | 50   | 60   | 80   | 100  | 130  | 160  |
| CF-cos  | 1.3332 | 1.4433 | 1.5078 | 1.3457 | 1.2363 | 1.5444 | 1.5270 | 1.4238 | 1.4086 | 1.4771 |
| CF-pos  | 1.4734 | 1.4166 | 1.4984 | 1.6300 | 1.3877 | 1.6571 | 1.6086 | 1.3675 | 1.5533 | 1.4638 |
| CF-ac   | 1.3923 | 1.2748 | 1.4063 | 1.4108 | 1.2769 | 1.3942 | 1.3996 | 1.3833 | 1.3733 | 1.4512 |
| CF-LDA  | 1.2342 | 1.2086 | 1.3037 | 1.2656 | 1.2278 | 1.2317 | 1.2673 | 1.2843 | 1.3045 | 1.2254 |
| CF-MCFV | 1.2268 | 1.4044 | 1.8904 | 1.358 | 1.2337 | 1.3610 | 1.4076 | 1.2398 | 1.6080 | 1.2923 |
| HDCF    | 1.1261 | 1.1923 | 1.2511 | 1.1704 | 1.2288 | 1.1191 | 1.1850 | 1.2353 | 1.2858 | 1.0496 |

TABLE 7. Performance results with RMSE on la.fm dataset.

| Method  | Neighbor Number |
|---------|-----------------|
|         | 10   | 20   | 30   | 40   | 50   | 60   | 80   | 100  | 130  | 160  |
| CF-cos  | 1.5604 | 1.3651 | 1.1636 | 1.2686 | 1.5525 | 1.4698 | 1.1505 | 1.5728 | 1.3119 | 1.3387 |
| CF-pos  | 1.3084 | 1.5191 | 1.3944 | 1.2879 | 1.5517 | 1.4426 | 1.2730 | 1.3798 | 1.4686 | 1.6461 |
| CF-ac   | 1.4091 | 1.3258 | 1.5977 | 1.3414 | 1.4025 | 1.4615 | 1.3559 | 1.3362 | 1.3353 | 1.4635 |
| CF-LDA  | 1.0978 | 1.1212 | 1.1855 | 1.1566 | 1.2832 | 1.1017 | 1.1622 | 1.1243 | 1.1108 | 1.0614 |
| CF-MCFV | 0.9476 | 1.2277 | 1.2734 | 1.0699 | 1.4882 | 1.1114 | 1.1710 | 1.2269 | 1.4527 | 1.4599 |
| HDCF    | 1.2769 | 0.9886 | 1.0379 | 1.1583 | 0.9560 | 0.9476 | 0.9978 | 0.9938 | 1.0116 | 0.9314 |

TABLE 8. T-test with MAE.

| Dataset | Test Value | t       | Sig.(2-tailed) | Mean difference | 95% Confidence interval of difference |
|---------|------------|---------|----------------|-----------------|--------------------------------------|
|         |            |         |                |                 | Lower                                | Upper                                |
| Movielens | .70559     | 4.797   | .005           | .12135834       | .0563273                            | .1863894                            |
| Netflix | .76097     | 3.812   | .019           | .10894600       | .0295908                            | .1883012                            |
| la.fm    | .77598     | 3.093   | .036           | .03207400       | .0032798                            | .0608682                            |

similarity. It is calculated as Equation (16). We set the parameter $\lambda = 0.1, 0.2, 0.3, 0.4, 0.5$. Figure 5 shows the results.

From Figure 7, we determine that the hybrid parameter $\lambda$ is 0.5.

We present our experimental results on the MovieLens 10 M dataset, Netflix dataset and last.fm dataset after we obtain the optimal values of the parameters. The last.fm dataset contains social networking, tagging, and music artist listening information from a set of 2K users from Last.fm online music system. The Netflix dataset contains over 100 million ratings. The ratings were collected between October 1998 and December 2005. Each rating has a customer id, a movie id, the date of the rating, and the value of the rating. In these two dataset, we generate topic model for “title” and “tag” while the number of topic is $K = 5$. We use these three baselines to demonstrate the ability to semantically interpret LDA. CF-cos represents CF with cosine as the similarity measure. CF-pcc represents CF with the Pearson correlation coefficient as the similarity measure. CF-ac represents CF with adjusted cosine as the similarity measure. We also compare three methods [23], [24], [28], CF-LDA, CF-MCFV and ROST to verify the time attribute and hybrid attribute. We set time slide $T = 10$. The results of MAE on different datasets are shown in TABLE 2, TABLE 3 and TABLE 4. The RMSE results on different datasets are shown in TABLE 5, TABLE 6 and TABLE 7.

In addition, we use the t-test to test the significance and give one-sample test results in Table 8. We also give std. deviation (with error bars) with MAE in Figure 8, Figure 9 and Figure 10. We conclude from these results. The MAE and
RMSE of the proposed method are significantly lower than those of the other methods. Experimental results show that the proposed method has high recommendation accuracy.

V. CONCLUSION

Collaborative filtering (CF) is a popular recommendation algorithm that makes predictions and recommendations based on the ratings or behaviors of other users in the system. The traditional CF algorithm assigns the same weight to all items of interest, ignoring the influence of user time on interest when calculating the similarity. This paper introduces the HDCF model and gives different weights to items according to the time users look at the items. In the future, we will continue to study how to improve the dynamic topic model to improve the quality of recommendations.

REFERENCES

[1] T.-M. Chang and W.-F. Hsiao, “LDA-based personalized document recommendation,” in Proc. PACIS, 2013, p. 13.
[2] Q. Liu, E. Chen, H. Xiong, C. H. Q. Ding, and J. Chen, “Enhancing collaborative filtering by user interest expansion via personalized ranking,” IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 42, no. 1, pp. 218–233, Feb. 2012.
[3] F. Ortega, J. Bobadilla, A. Hernando, and A. Gutiérrez, “Incorporating group recommendations to recommender systems: Alternatives and performance,” Inf. Process. Manage., vol. 49, no. 4, pp. 895–901, Jul. 2013.
[4] J. Wang and Y. Zhang, “Opportunity model for e-commerce recommendation: Right product; right time,” in Proc. 36th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr., Jul. 2013, pp. 303–312.
[5] Ç. Bakır and S. Albayrak, “User based and item based collaborative filtering with temporal dynamics,” in Proc. 22nd Signal Process. Commun. Appl. Conf. (Sius), Apr. 2014, pp. 252–255.
[6] Z. Wang, J. Liao, Q. Cao, H. Qi, and Z. Wang, “Friendbook: A semantic-based friend recommendation system for social networks,” IEEE Trans. Mobile Comput., vol. 14, no. 3, pp. 538–551, Mar. 2015.
[7] H. P. Xuan, J. J. Jung, B. K. H. Nam, and T. T. Nguyen, “User timeline and interest-based collaborative filtering on social network,” in Proc. Int. Conf. Context-Aware Syst. Appl. Springer, 2015, pp. 132–140.
[8] L. Tang, “Thresholding for top-k recommendation with temporal dynamics,” 2015, arXiv:1506.02190. [Online]. Available: https://arxiv.org/abs/1506.02190
[9] F. Yang, J. Mou, J. Liu, C. Ma, and H. Yan, “Characteristic analysis of the fractional-order hyperchaotic complex system and its image encryption application,” Signal Process., vol. 169, Apr. 2020, Art. no. 107373.
[10] J. Wang, B. Sarwar, and N. Sundaresan, “Utilizing related products for post-purchase recommendation in e-commerce,” in Proc. 5th ACM Conf. Recommender Syst. (RecSys), 2011, pp. 329–332.
[11] M. A. Ghazanfar, S. Szedmak, and A. Prügel-Bennett, “Incremental kernel mapping algorithms for scalable recommender systems,” in Proc. IEEE 23rd Int. Conf. Tools Artif. Intell., Nov. 2011, pp. 1077–1084, doi: 10.1109/ICTAI.2011.183.
[12] M. A. Ghazanfar, A. Prügel-Bennett, and S. Szedmak, “Kernel-mapping recommender system algorithms,” Inf. Sci., vol. 208, pp. 81–104, Nov. 2012.
[13] J.-J. Cheng, L.-F. Hu, S. Gao, J.-J. Liu, and Y. Yang, “An optimized collaborative filtering method to construct spatial-temporal behavior pattern-based user interest model,” IEEE Trans. Electr. Electron. Eng., vol. 12, no. 2, pp. 221–227, Mar. 2017.
[14] N. E. Karabadjaj, S. Beldjoudi, H. Seridi, S. Aridhi, and W. Dhifli, “Improving memory-based user collaborative filtering with evolutionary multi-objective optimization,” Expert Syst. Appl., vol. 98, pp. 153–165, May 2018.
[15] Z. Niu, G. Hua, L. Wang, and X. Gao, “Knowledge-based topic model for unsupervised object discovery and localization,” IEEE Trans. Image Process., vol. 27, no. 1, pp. 50–63, Jan. 2018.
[16] Z. Xiong, C. Fu-Cai, and H. Rui-Yang, “Semi-supervised entity disambiguation method research based on biterm topic model,” Chin. J. Electron., vol. 46, no. 3, pp. 607–613, 2018.
[17] J. Lu, J. Xuan, G. Zhang, and X. Luo, “Structural property-aware multilayer network embedding for latent factor analysis,” Pattern Recognit., vol. 76, pp. 228–241, Apr. 2018.
[18] Q. Zhang, J. Lu, D. Wu, and G. Zhang, “A cross-domain recommender system with kernel-induced knowledge transfer for overlapping entities,” IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 7, pp. 1998–2012, Jul. 2019.
[19] J. Dai, S. Majumder, P. Gupta, and K. Mali, “Collaborative recommendations using hierarchical clustering based on K-d trees and quadtrees,” Int. J. Uncertainty, Fuzziness Knowl.-Based Syst., vol. 27, no. 4, pp. 637–668, Aug. 2019.
[20] P. Zhang, Z. Zhang, T. Tian, and Y. Wang, “Collaborative filtering recommendation algorithm integrating time windows and rating predictions,” Appl. Intell., vol. 49, pp. 3146–3157, Aug. 2019.
[21] X. Yu, Y. Chu, F. Jiang, Y. Guo, and D. Gong, “SVMs classification based two-side cross domain collaborative filtering by inferring intrinsic user and item features.” Knowl.-Based Syst., vol. 141, pp. 80–91, Feb. 2018.
[22] X. Yu, F. Jiang, J. Du, and D. Gong, “A cross-domain collaborative filtering algorithm with expanding user and item features via the latent factor space of auxiliary domains,” Pattern Recognit., vol. 94, pp. 96–109, Oct. 2019.
[23] W. Han and X. Hong-Bin, “Collaborative filtering recommendation algorithm mixing LDA model and list-wise model,” Comput. Sci., vol. 46, no. 9, pp. 216–222, 2019.
[24] H.-R. Zhang, F. Min, Z.-H. Zhang, and S. Wang, “Efficient collaborative filtering recommendations with multi-channel feature vectors,” Int. J. Mach. Learn. Cybern., vol. 10, no. 5, pp. 1165–1172, May 2019.
[25] L. Na et al., “A hybrid user-based collaborative filtering algorithm with topic model,” Appl. Intell., Mar. 2021, doi: 10.1007/s10489-021-02207-7.
[26] D. M. Blei and J. D. Lafferty, “Dynamic topic models,” in Proc. 23rd Int. Conf. Mach. Learn. (ICML), 2006, pp. 113–120.
[27] A.-L. Deng, Y.-Y. Zha, and B.-L. Shi, “A collaborative filtering recommendation algorithm based on item rating prediction,” J. Softw., vol. 14, no. 9, pp. 54–65, 2003.
[28] B. Li, X. Zhu, R. Li, and C. Zhang, “Rating knowledge sharing in cross-domain collaborative filtering,” IEEE Trans. Cybern., vol. 45, no. 5, pp. 1068–1082, May 2015.