2012 Ninth IEEE International Conference on e-Business Engineering

Personalized Recommendation Based on Reviews and Ratings Alleviating the Sparsity Problem of Collaborative Filtering

Jingnan Xu
Department of Computer Science
Zhejiang University
Zhejiang, P.R. China
xujingnan308@gmail.com

Xiaolin Zheng
Department of Computer Science
Zhejiang University
Zhejiang, P.R. China
xlzheng@zju.edu.cn

Weifeng Ding
Department of Computer Science
Zhejiang University
Zhejiang, P.R. China
vvkkharry@gmail.com

Abstract- with the development of e-commerce, shopping online is becoming more and more popular. When we need to decide whether to purchase a product or not on line, the opinions of others become important. The convenience of new web technologies enables us to freely express our opinions and reviews for various products we have purchased which leads to a serious problem, information overloading. How to mine these review data to understand customers' preferences and make recommendations is crucial to merchants and researchers. Traditional collaborative filtering (CF) algorithm is one of the most successful recommendation system technologies. The core idea of CF algorithm is to recommend products based on other people who have similar tastes with target users. However, the ability of CF is limited by the sparsity problem, which is very common in reality. The reason derives from the fact that traditional CF method only takes users' ratings into account. In this paper, we propose a new personalized recommendation model, i.e. topic model based collaborative filtering (TMCF) utilizing users' reviews and ratings. We exploit extended LDA model to generate topic allocations for each review and then obtain each user's preference. Moreover, a new metric is designed to measure similarity between users alleviating the sparsity problem to a large extent. Finally, recommendations are made based on similar users' ratings. Experiments on seven data sets indicate better prediction accuracy than other traditional and state-of-the-art methods with substantial improvement in alleviating the sparsity problem.

Keywords: Collaborative Filtering Recommendation, Topic Model, Sparsity Problem

I. INTRODUCTION

With the development of Web 2.0, more and more people are connecting to the Internet and becoming information producers instead of only information consumers in the past, resulting to the serious problem, information overloading. variety of metrics are applied to evaluate the similarity value between users such as Euclidean distance similarity, Pearson correlation similarity and so forth.

However, the availability of traditional CF is limited by the sparsity problem. When it comes to a user who rates few items, user-based CF, e.g. often cannot work due to the difficulty of calculating similarity. Even if the similarity value can be computed, the result is not convincing as there are few co-rated items between two users. Unfortunately, this data sparsity is common in practice [2].

Meanwhile, merchants often ask their customers to express their reviews on the products they have purchased. These reviews are important reference suggestions when people decide whether to buy a product or not. In general, users always show their preferences for a product through their reviews. Table 1 below illustrates a Laptop review example from Amazon. The content in this table describes the user's positive opinions about the computer's design, resolution and battery life. Negative opinion about screen is expressed also.

TABLE 1. An laptops review example

ID: B002C744K6-R3OV0NNVQQRV0O-A2QUTLY11FR5JR Category: Laptops Product: Apple-MacBook-MC118LL-15-4-Inch-Laptop ReviewerID: A2QUTLY11FR5JR Rating: 3.0 Content: … .Im impressed with the design of Macbook pro and how fast it is to start up and shut down. The resolution is amazing, particularly with iphoto, you can see every detail of the picture. I have yet to explore other apps other than itunes. The advertised battary life was 7 hours, I haven't fully tested that yet, although I noticed that the battery generally start at about 5-6 hours after a full charge. I would have given this product a five stars review if it wasn't for the mac freezes. For example, I was importing a CD into itunes, but after importing

understand users' preferences from users' reviews. Moreover, recommendations based on such semantic information can be more useful and more interpretable for customers to accept.

Collaborative information is good at recommending items that target users may interest from other similar users by utilizing ratings, and content-based method can help discover similar users with same taste by exploiting reviews. So our purpose is to make personalized recommendations based on users' reviews and ratings.

This paper seeks to address the following questions:

(1) Make recommendations based on reviews and ratings
- Infer users' preferences
- Define users' similarity
- Recommend based on users' similarity and ratings

(2) Alleviating the sparsity problem to a large extent

In the case of review contents, ASUM model in [3] is adopted, which improves the original LDA model to fit online reviews. Prior to inferring users' preferences, ASUM is employed to model the generation process of reviews. Topic allocations for each review will be obtained from this process. We continue to estimate users' preferences in the following. A new metric is designed to measure each user's most valued features and similar neighbors next. Finally, each user's ratings on unknown items are predicted based on similar neighbors' ratings for those items and those predicted items will be recommended in a descending order.

The rest of this paper is organized as follows. Section 2 gives a brief overview of recent research in opinion analysis and collaborative filtering recommendation. Our topic model based collaborative filtering model (TMCF) will be introduced in the next section. Experimental results on real data sets will be represented in section 4. Finally, conclusion of our work and future work are discussed in section 5.

II. Related Work

In recent years, there has been an increasing amount of literature on review mining/opinion analysis, which focuses on mining useful information from online reviews in the level of features of products.

A major method used in review mining is to extract frequent nouns as product features. Liu et al. [4-7] have designed a framework to resolve this problem. When dealing with features identification, association rule mining algorithm is applied to produce frequent nouns as candidate features. Adjective words near products' features are simply extracted as opinion words. Then a lexicon-based method is and sentiments which is time-consuming and cannot fit for other domains.

A large body of literature has investigated how to utilize topic modeling to perform review mining in the last decade.

Blei et al. [10] first propose Latent Dirichlet Allocation (LDA) to model text corpora. LDA is a three-level Bayesian model, where each document is modeled as a finite mixture over an underlying set of topics. Each topic is represented as a probabilities distribution over all words in the text corpora.

Several researchers make progress in topic modeling [3, 11-13]. These approaches all take sentiments into account to extend original LDA model to suit for online user reviews.

Mei et al. [11] propose a Topic Sentiment Mixture (TSM), in which sentiment is represented as a language model separating from topic model. Each word in documents may come from either topics or sentiments. This separation may cause the difficulty of explaining the relationship between topics and sentiments.

Multi-Aspect Sentiment (MAS) model is proposed by Titov and McDonald [12]. Topics are modeled to fit a set of predefined features that are explicitly rated by customers in reviews. However, supervised data are not always available and it lacks of the flexibility to be fit for other domains.

A Joint Sentiment/Topic Model (JST) is investigated by Lin and He [13]. Individual words in a sentence may be generated from different sentiment models and topic models, which may not always be consistent with true cases.

ASUM model is proposed by Yohan and Alice in [3]. It posits that one sentence in a review tends to represent one feature of a product and one sentiment towards that feature. From Table 1, we know this assumption works in common practice. In this model, each review has its sentiment distribution. And for each sentiment, there is a topic distribution with it. A word distribution is drawn from each pair of sentiment and topic. Words in the same sentence belong to the same sentiment and topic. In a quantitative analysis, ASUM outperforms other generative models, so it is employed to be integrated into our model.

Collaborative filtering recommendation is another domain related to our work. Traditional CF, e.g. user-based CF, assumes that items favored by other similar users may be liked by target user. So it needs to determine the similar neighbors based on different similar metrics. But similarity-based calculation is often difficult to perform or the result is often unconvincing when target user rates few items.

who have rated 20 or more movies, which results more dense data distribution than that of our experiments.

Desrosiers et al. [16] compute global similarities between pairs of items and users, based on the solution to system of linear equations relating user similarities to item similarities. However, a good interpretation of similarity measures in the context of prediction problem is lacking in such a graph theoretic measures based method.

A random walk method that uses finite length random walks to produce item recommendations is presented in [17]. However, it is computationally expensive to measure similarities among users or items on a large scale.

Pan et al. [18] deal with the data sparsity problem in a target domain by transferring knowledge about both users and items from auxiliary data sources. However, it requires that either users or items are shared between domains. Such hypothesis is not commonly encountered in practice.

Relationships between topic model and recommendation are also investigated in [19-23].

Sriurai et al. [19, 20] propose a method to recommend related articles in Wikipedia via a topic-based model. Original LDA algorithm is used to generate topic allocations for each article represented as a probability distribution over different topics. Dot product is applied to calculate articles' similarity. However, our method is based on reviews with sentiments. What's more, users' similarity is redesigned instead of simply using dot product. The last but not the least, not only reviews, but also ratings are taken into account in our model.

A method combining the merits of CF and topic model is applied by Wang and Blei in [21]. But their goal is aimed at recommending scientific articles which is the same as [19, 20] but different from our problem and our rating matrix is much sparser than their rating matrix where each user has 37 ratings in average.

A maximum entropy principle is proposed to integrate ratings and reviews by Jin et al. in [22]. However, as in [19, 20], it uses original LDA algorithm to deal with reviews.

A state-of-the-art method is proposed by Pennacchiotti and Gurumurthy [23] to make friends recommendations. A user-level LDA model is put forward to represent users as mixtures of topics, which is almost the same as ours. However, our model can deal with sentiments in reviews which are not considered in their model. What's more, when calculating users' similarity, users' most valued features are taken into account instead of applying cosine similarity between users. We notice that in their approach data pre-

III. Topic Model based Collaborative Filtering

Prior to introduce our TMCF algorithm, some notations are needed to be defined first.

A. Notation

[illegible] TopicNumber is a parameter in our experiments indicating the topic number of reviews,

[illegible] is another parameter to denote the feature number,

[illegible] describes the neighbor number,

[illegible] is the user set,

[illegible] is the item set,

[illegible] denotes rating matrix. If there is no rating given by [illegible] to [illegible], [illegible]

[illegible] represents the item set rated by [illegible]

[illegible] is used to represent the predictive rating matrix,

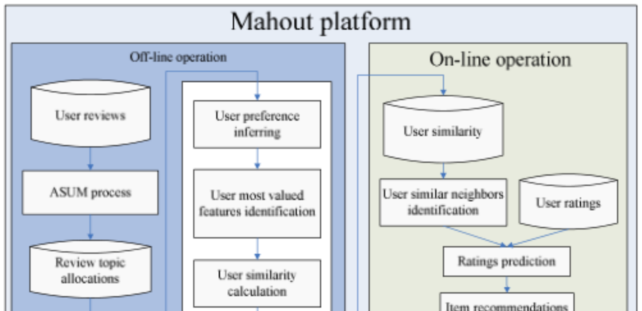
[illegible] indicates the review topic allocation matrix. [illegible] is the topic allocation of review which is given by [illegible] to [illegible]. If there is no review given by [illegible] to [illegible], [illegible] which means there is no operation on this [illegible] in our experiments,

[illegible] denotes the user topic allocation set,

[illegible] represents [illegible]'s most valued feature set.

B. Topic model based collaborative filtering model

The TMCF model consists of two components based on the Mahout platform shown in Figure 1. Offline component accepts users' reviews to generate users' similarity. Online component makes recommendations for target users.



allocation will be a probability distribution over T= TopicNumber*2 topics, which means for each topic, we have two probability distribution values, one for positive sentiment and the other for negative. This is why [illegible].

From the point of semantics, each review's topic allocation makes us understand user's preference better. Table 2 illustrates a review example for an Apple laptop. There are several sentences about the speed feature in this example. After the process of ASUM, these sentences will all be about the same topic, increasing the probability value of speed topic. It is obviously that from this topic allocation, we may know that the user pays more attention to the speed while little attention to other features.

TABLE 2. A review example

It does not feel cheap, and it is very FAST. This is a speedy computer in many ways. I run everything from iWork, iLife, etc. If you want a real computer, this is the one.

Since reviews' topic allocations can express users' preferences for corresponding items, we can extend the review's topic allocation to user's preference from all the reviews given by the user.

We define [illegible] as the topic probability distribution of review given by [illegible] to [illegible] and [illegible] denotes the probability value for the kth feature. User topic allocation, i.e. user preference can be represented with the following formula:

$$[illegible] \quad (1)$$

The same as [illegible], [illegible] is a topic probability distribution of [illegible] from its reviews' topic allocations, [illegible].

Prior to proposing our metric for users' similarity, we need to define users' most valued features. When a user seeks recommendations from other people for an item, other people will often want to know what the user's most valued feature is. In most cases, users will consider several features when they are evaluating an item. Generally, the number of features considered may not be too large. In our experiments, we use the variable F to control the number of features to generate users' most valued features. We define users' most valued features as below:

$$[illegible] \quad (2)$$

TABLE 3. Procedure of MostValuedFeatures

Procedure MostValuedFeatures for [illegible]
Input: [illegible] and F
Results, the set of feature indexes
1: sort [illegible] by the value of [illegible] in a descending order

Once users' most valued features are computed, users' similarity can be obtained with the following formula:

$$[illegible] \quad (3)$$

[illegible] is a function which returns the common feature number of the first k number of features in [illegible]. Detailed implementation is represented in Table 4.

TABLE 4. Procedure of CommonCount

Procedure CommonCount for [illegible] and [illegible]
Input: [illegible]'s most valued features [illegible], [illegible]'s most valued features [illegible], k is the boundary of the computation and [illegible]
Sum is the common feature counts in the first k features of [illegible] and [illegible] and is initialized to be 0
1. for all [illegible]
2. for all [illegible]
3. if([illegible]) then
4. sum++, break
5. end if
6. end for
7. end for
Output: sum

Why not use other simpler similarity metric such as:

$$[illegible] \quad (4)$$

The reason is that such metric can not reflect the users' similarity in most value features. For example, we assume [illegible]. If [illegible] is carried out, we will get [illegible] but we know [illegible] should be more similar as the fact that they all most value feature 1. [illegible] produces more precise results by taking the features order into consideration.

Following is the results of [illegible] and [illegible]

[illegible]

In the end, we use the similar method used in user-based CF to make recommendations as below:

$$[illegible] \quad (5)$$

[illegible] indicates the predictive rating given by [illegible] to [illegible], and [illegible] is a function which returns the nearest N neighbors of [illegible] and these neighbors all have already rated the [illegible]. The nearest neighbors of [illegible] mean the most similar neighbors [illegible] according to the [illegible] value.

Finally, we represent the full pseudo code of TMCF algorithm as below:

TABLE 5. TMCF algorithm

TMCF algorithm

IV. EXPERIMENTS

We will first introduce our data sets and evaluation metric, and then describe our experimental settings. Finally we compare our model with three basic CF algorithms, user-based CF, item-based CF, SlopeOne CF, and the extended state-of-the-art algorithm user-level LDA CF.

A. Data sets and evaluation metric

We conduct our experiments on the open source project, mahout[1], which is a machine learning library under Apache Software Foundation. We have implemented our algorithm and the other four algorithms on the platform.

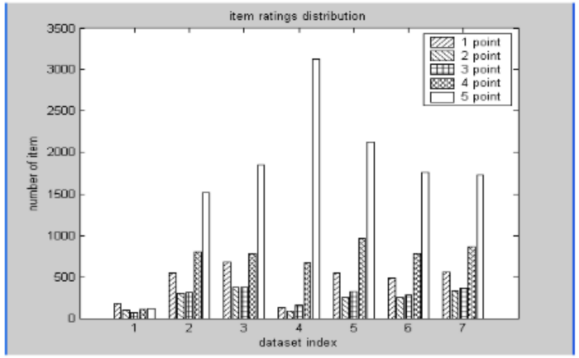
The data sets we use are from the paper [3]. These data sets are a collection of electronic device reviews from Amazon which belong to seven categories. Each file in the data sets has the format as is shown in Table 1.

We perform all algorithms on these data sets separately. For convenience, we will use 1-7 representing these data sets. Table 6 below shows some properties of the data sets.

TABLE 6. Properties of the data sets

Data sets	# of reviews	# of reviewers	# of items	Sparsity
AirConditioners-1	551	507	121	0.991
CanisterVacuums-2	3474	3420	143	0.993
CoffeeMachines-3	4054	4016	284	0.996
DigitalSLRs-4	4014	3150	188	0.993
Laptops-5	4065	3286	623	0.998
MP3Players-6	3504	3239	375	0.997
SpaceHeaters-7	3822	3721	366	0.997

In Figure 2, we show statistical rating distribution in these seven data sets. The horizontal axis represents the seven data sets and each data set has 5 different color bars indicating 1-5 rating points.



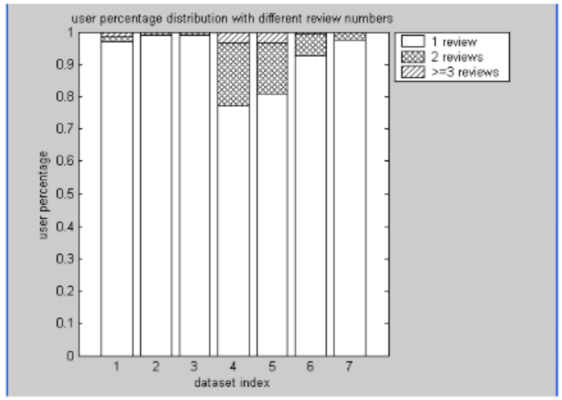


Figure 3. users' percentage distribution

Mean Absolute Error (MAE) is the most widely used metric when comparing actual ratings and predictive ratings. So we adopt it as our metric which computed as follows:

(6)

Where is the absolute error between actual rating and predictive rating

B. Experimental settings

For user-based CF (UCF), Euclidean distance similarity (Euclidean) and Pearson correlation similarity (Pearson) are experimented to find a better similarity metric. As for the neighborhood setting, NearestNUserNeighborhood class from mahout is applied with various neighbor numbers. This parameter N is set to be 2, 4, 8, 16 and 32.

For item-based CF (ICF), Euclidean and Pearson are adopted to test again. As there is no similar user in ICF, no N evaluation experiment is performed.

Because there is no similarity utilized in SlopeOne CF (SCF), no experiments with different similarity metric and N is conducted for SCF.

For user-level LDA CF (ULCF), two experiments are performed with different filtering strategies. One is filtering all users who have less than 5 reviews, just as the process mentioned in [23], and the other is remaining all users even when most users only rate/review one item.

For our TM CF model, various parameters are examined, e.g. TopicNumber and IterationNumber in the process of ASUM. TopicNumber determines the number of topics of each review and IterationNumber decides the iteration

from mahout, also known as a MAE algorithm, is adopted to evaluate the quality of all five algorithms' recommendations.

C. Comparisons

Due to extreme sparsity of our data sets (average larger than 99% for all seven data sets), we often confront the recommendation results of NaN. NaN means "not a number" and no answer could be computed. This problem is mainly caused by the fact that the data sets are too sparse. Not only Euclidean but also Pearson cannot be computed in such a situation, i.e. NaN means the worst recommendation result.

In the following, comparisons between TMCF with other four algorithms are presented separately.

1) UCF vs. TMCF

The result obtained from TMCF can be seen from Table 7 and Table 8. Table 7 gives an intermediate result of TMCF on data set 4 and Table 8 contains the final result. The first row of Table 7 represents various F, and the first column denotes different N and total MAE value for each data set. It is apparent that when F is selected 4, the total MAE achieves the smallest over all N. This column data are chosen as the result of TMCF on data set 4.

TABLE 7. Result of TMCF of data set 4

	1	2	3	4	5
2	0.585	0.591	0.581	0.571	0.596
4	0.58	0.587	0.559	0.576	0.586
8	0.546	0.572	0.572	0.596	0.62
16	0.6	0.562	0.606	0.536	0.57
32	0.551	0.578	0.572	0.576	0.594
Total	2.862	2.890	2.890	2.855	2.966

Results from other 6 data sets are conducted with the same method to form final result of TMCF in Table 8. The average results for each data set are used to compare with that of UCF. The smallest MAE value of TMCF in each data set is marked with bold type which will be chosen to compare with other algorithms with no similar neighbors.

Table 8 also provides the result obtained from UCF with Euclidean and various N over 7 data sets. The outputs of UCF with Pearson are all NaN, so there is no table to show the results. A possible explanation for this is the data sparsity. When two users overlap in only one rated item which is very common in our data sets, no correlation can be computed, due to the definition of Pearson.

It deserves notice that although UCF with Euclidean outperforms UCF with Pearson in data sets 4, 5, 6, there are still 4 data sets all with NaN results. This reason may still derive from sparsity problem in data sets. Why results of data sets 4, 5, 6 are better than others? Another explanation is users' percentage distribution in data sets as shown in Figure 3. Fewer ratings are given, more difficult to calculate similarity. It is interesting to note that in these 3 data sets, the MAE value is quite low, i.e. UCF performs pretty well in these data sets. This finding is in agreement with the effectiveness of traditional CF algorithms demonstrated in a great deal of previous works.

Table 8 illustrates the comparison results of UCF and TMCF over all 7 data sets. Although UCF returns NaN on data set 6 when N is 2, we still calculate the average MAE of other four N as UCF's average MAE value on data set 6. Better result for each data set is marked with bold type.

This table is quite revealing in several ways. First, TMCF outperforms UCF in 4 data sets (1, 2, 3, 7) where UCF can not recommend at all. Moreover, the average MAE values over these 4 data sets are acceptable (1.22 in the range of 1-5). Second, UCF can achieve better results in other 3 data sets (4, 5, 6) and the MAE values are quite small. This finding corroborates the effectiveness of traditional CF algorithms in making recommendations. Third, the contradiction may come from the different users' percentage distribution over these 7 data sets. We may assume that in an application where most users only rate and review one item, our approach outperforms traditional CF algorithms. However, in an opposite scenario, traditional CF algorithms can achieve better recommendations. We will continue to prove this hypothesis in following comparisons.

TABLE 8. Comparison of UCF and TMCF

	1		2		3		4		5		6		7	
	UCF	TMCF	UCF	TMCF	UCF	TMCF	UCF	TMCF	UCF	TMCF	UCF	TMCF	UCF	TMCF
2	NaN	1.695	NaN	1.316	NaN	1.112	0.018	0.571	0.091	0.933	NaN	1.119	NaN	1.116
4	NaN	1.205	NaN	1.169	NaN	1.158	0.046	0.576	0.126	0.821	0.067	1.124	NaN	1.316
8	NaN	2.002	NaN	1.113	NaN	1.099	0.11	0.596	0.226	0.81	0.288	1.143	NaN	1.238
16	NaN	1.199	NaN	1.121	NaN	1.055	0.208	0.536	0.345	0.808	0.42	1.129	NaN	1.209
32	NaN	1.325	NaN	1.165	NaN	0.75	0.293	0.576	0.429	0.795	0.672	1.226	NaN	1.093

TABLE 9. Comparison of ICF and TMCF

	ICF-Euclidean	ICF-Pearson	TMCF
1	NaN	NaN	1.199
2	NaN	NaN	1.113
3	NaN	NaN	0.75
4	0	0	0.536
5	0.02	0	0.795
6	0	NaN	1.119
7	NaN	NaN	1.093

3) SCF vs. TMCF

As there is no similarity needed to be exploited in SCF, it can alleviate the sparsity problem to some extent. Data from Table 10 below testifies such improvement when compared with Table 8 and Table 9.

TABLE 10. Comparison of SCF and TMCF

	SCF	TMCF
1	NaN	1.199
2	1.543	1.113
3	1.963	0.75
4	0.059	0.536
5	0.130	0.795
6	0.629	1.119
7	1.234	1.093

It can be seen from Table 10 that SCF makes recommendations in more data sets than UCF and ICF. However, there is still one data set where SCF only returns NaN. It demonstrates that when data set is extremely sparse, SCF still cannot be a good choice to make recommendations. What's more, after comparing results between SCF and TMCF, TMCF still outperforms SCF, not only in the aspect of availability for sparse data sets but also in the recommendation precision in 4 data sets (1, 2, 3, 7). But when we turn to other 3 data sets, SCF can generate better results. Similar results appear and prove our hypothesis again. Now, we can further confirm that our TMCF model outperforms traditional CF algorithms, including UCF, ICF and SCF, especially when most users rate and review only one item in data set. But if users rate more items, traditional CF is still a good choice to make recommendations.

4) ULCF vs. TMCF

In this part, comparison between the state-of-the-art method ULCF with all users and TMCF is discussed. Table 11 compares the results obtained from ULCF and TMCF with various N over 7 data sets. The result of ULCF with users who have at least 5 reviews is not represented because the result are all NaN over all 7 data sets, i.e. the original algorithm in [23] cannot generate meaningful results in our data sets. It is natural because such pre-process will discard almost all users in our data sets which will lead to the difficulty of calculating users' similarity.

From Table 11, we can draw three conclusions. Firstly, when compared with traditional CF, ULCF and TMCF are both topic model based collaborative filtering algorithms which not only consider users' ratings but also take users' reviews' content into account. It is the reviews that help us make recommendations even when most users only rate and review one item. Secondly, with more users' ratings and reviews, topic model based collaborative filtering algorithms can generate better recommendation results. This can be seen from the average results of the 3 data sets (4, 5, 6). Both algorithms produce smaller MAE values in these 3 data sets than that of other 4 data sets. For example, the average MAE values of TMCF on data set 4, 5 are 0.571 and 0.833, which are much better than others. ULCF shows the same trend as TMCF. Finally, our TMCF outperforms ULCF with smaller average MAE values over all 7 data sets. This may be caused by the fact that we adopt ASUM model instead of applying original LDA and users' most valued features are taken into account instead of using simple cosine similarity when calculating users' similarity.

TABLE 11. Comparison of ULCF and TMCF

	1		2		3		4		5		6		7	
	ULCF	TMCF	ULCF	TMCF	ULCF	TMCF	ULCF	TMCF	ULCF	TMCF	ULCF	TMCF	ULCF	TMCF
2	1.822	1.695	1.09	1.316	1.373	1.112	0.591	0.571	0.985	0.933	1.394	1.119	1.351	1.116
4	0.863	1.205	1.469	1.169	1.207	1.158	0.596	0.576	0.978	0.821	1.218	1.124	1.234	1.316
8	1.659	2.002	1.182	1.113	1.284	1.099	0.585	0.596	0.966	0.81	1.175	1.143	1.247	1.238
16	1.656	1.199	1.332	1.121	1.375	1.055	0.604	0.536	0.971	0.808	1.299	1.129	1.285	1.209
32	2.16	1.325	1.353	1.165	1.367	0.75	0.625	0.576	0.965	0.795	1.275	1.226	1.227	1.093
Ave	1.632	1.485	1.285	1.177	1.321	1.035	0.600	0.571	0.973	0.833	1.272	1.148	1.269	1.194

different domains show our model can achieve better

to be performed to testify the effectiveness and efficiency of our model.

ACKNOWLEDGMENT

This work is supported in part by the National Natural Science Foundation of China (Grant No. 61003254), the National Key Technology R&D Program (No. 2012BAH16F02) and the Fundamental Research Funds for the Central Universities.

REFERENCES

[1] D. Billsus and M.J. Pazzani, "Learning collaborative information filters," Proceedings of the Fifteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc, 1998, pp. 46-54, doi:10.1.1.40.4781.

[2] M. Grcar, D. Mladenic, B. Fortuna and M. Grobelnik, "Data sparsity issues in the collaborative filtering framework," Advances in Web Mining and Web Usage Analysis, vol. 4198, 2006, pp. 58-76, doi:10.1007/11891321_4.

[3] Y. Jo and A.H. Oh, "Aspect and sentiment unification model for online review analysis," Proceedings of the fourth ACM international conference on Web search and data mining, ACM Press, 2011, pp. 815-824, doi:10.1145/1935826.1935932.

[4] M. Hu and B. Liu, "Mining and summarizing customer reviews," Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM Press, 2004, pp. 168-177, doi:10.1145/1014052.1014073.

[5] M. Hu and B. Liu, "Mining opinion features in customer reviews," Proceedings of the National Conference on Artificial Intelligence, AAAI Press, 2004, pp. 755-760.

[6] B. Liu, M. Hu and J. Cheng, "Opinion observer: analyzing and comparing opinions on the Web," Proceedings of the 14th international conference on World Wide Web, ACM Press, 2005, pp. 342-351, doi:1-59593-046-9.

[7] M. Hu and B. Liu, "Opinion extraction and summarization on the web," Proceedings Of The 21st National Conference On Artificial Intelligence, AAAI Press, 2006, pp. 1621-1624.

[8] G. Ganu, A. Marian and N. Elhadad, "URSA-User Review Structure Analysis: Understanding Online Reviewing Trends," DCS Technical Report, 2010.

[9] G. Ganu, Y. Kakodkar and A. Marian, "Improving the quality of predictions using textual information in online user reviews," Information Systems, 2012, doi:10.1016/j.is.2012.03.001.

[10] D.M. Blei, A.Y. Ng and M.I. Jordan, "Latent dirichlet allocation," The Journal of Machine Learning Research, vol. 3, 2003, pp. 993-1022.

[11] Q. Mei, X. Ling, M. Wondra, H. Su and C.X. Zhai, "Topic analysis," Proceeding of the 18th ACM conference on Information and knowledge management, ACM Press, 2009, pp. 375-384, doi:10.1145/1645953.1646003.

[13] I. Titov and R. McDonald, "A joint model of text and aspect ratings for sentiment summarization," Proceedings of 46th Annual Meeting of the Association for Computational Linguistics (ACL'08), 2008.

[14] M. Papagelis, D. Plexousakis and T. Kutsuras, "Alleviating the sparsity problem of collaborative filtering using trust inferences," Trust management, vol. 3477, 2005, pp. 125-140, doi:10.1007/11429760_16.

[15] A. Abdelwahab, H. Sekiya, I. Matsuba, Y. Horiuchi and S. Kuroiwa, "Collaborative filtering based on an iterative prediction method to alleviate the sparsity problem," Proceedings of the 11th International Conference on Information Integration and Web-based Applications & Services, ACM Press, 2009, pp. 375-379, doi:10.1145/1806338.1806406.

[16] C. Desrosiers and G. Karypis, Solving the sparsity problem: Collaborative filtering via indirect similarities. 2008, Technical Report.

[17] H. Yildirim and M.S. Krishnamoorthy, "A random walk method for alleviating the sparsity problem in collaborative filtering," Proceedings of the 2008 ACM conference on Recommender systems, ACM, 2008, pp. 131-138, doi:10.1145/1454008.1454031.

[18] W. Pan, E.W. Xiang, N.N. Liu and Q. Yang, "Transfer learning in collaborative filtering for sparsity reduction," Proceedings of the 24rd AAAI Conference on Artificial Intelligence, ACM, 2010.

[19] C. Haruechaiyasak and C. Damrongrat, "Article recommendation based on a topic model for Wikipedia Selection for Schools," Digital Libraries: Universal and Ubiquitous Access to Information, vol. 5362, 2008, pp. 339-342, doi:10.1007/978-3-540-89533-6_39.

[20] W. Sriurai, P. Meesad and C. Haruechaiyasak, "Recommending Related Articles in Wikipedia via a Topic-Based Model," In IICS, 2009, pp. 194-203.

[21] C. Wang and D.M. Blei, "Collaborative topic modeling for recommending scientific articles," Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM Press, 2011, pp. 448-456, doi:10.1145/2020408.2020480.

[22] X. Jin, Y. Zhou and B. Mobasher, "A maximum entropy web recommendation system: combining collaborative and content features," Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, ACM Press, 2005, pp. 612-617, doi:10.1145/1081870.1081945.

[23] M. Pennacchiotti and S. Gurumurthy, "Investigating topic models for social media user recommendation," Proceedings