Time-aspect-sentiment Recommendation Models Based on Novel Similarity Measure Methods

GUOHUI LI, School of Software, Huazhong University of Science and Technology, China
QI CHEN and BOLONG ZHENG, School of Computer Science and Technology, Huazhong University of Science and Technology, China
NGUYEN QUOC VIET HUNG, School of Information and Communication Technology, Griffith University, Australia
PAN ZHOU, School of Cyber Science and Engineering, Huazhong University of Science and Technology, China
GUANFENG LIU, Department of Computing, Macquarie University, Australia

The explosive growth of e-commerce has led to the development of the recommendation system. The recommendation system aims to provide a set of items that meet users' personalized needs through analyzing users' consumption records. However, the timeliness of purchasing data and the implicity of feedback data pose severe challenges for the existing recommendation methods. To alleviate these challenges, we exploit the user's consumption records from the perspectives of user and item, by modeling the data on both item and user level, where the item-level value reflects the grade of item, and the user-level value reflects the user's purchase intention. In this article, we collect the description information and the reviews of the items from public websites, then adopt sentiment analysis techniques to model the similarities on user level and item level, respectively. In particular, we extend the traditional latent factor model and propose two novel methods—**I**tem **L**evel Similarity **M**atrix **F**actorization (ILMF) and **U**ser **L**evel Similarity **M**atrix **F**actorization (ULMF)—by introducing two novel similarity measure methods. In ILMF and ULMF, the consistency between latent factors and explicit aspects is naturally incorporated into learning latent factors of the users and items, such that we can predict the users' preferences on different items more accurately. Moreover, we propose **I**tem-**U**ser **L**evel Similarity **M**atrix **F**actorization (IULMF), which combines these two methods to study their contributions on the final performance. Experimental evaluations on the real datasets show that our methods outperform the baseline approaches in terms of both the precision and NDCG.

CCS Concepts: • **Information systems and Social recommendation**

Additional Key Words and Phrases: Recommendation system, time, aspect, sentiment analysis, matrix factorization

This research is partially supported by the NSFC (Grant Nos. 61902134, 61572215, 61872278, 61972448) and the Fundamental Research Funds for the Central Universities (HUST: Grant Nos. 2019kfyXKJC021, 2019kfyXJJS091).

Authors' addresses: G. Li, School of Software, Huazhong University of Science and Technology, Wuhan, China; email: guohuili@hust.edu.cn; Q. Chen and B. Zheng (corresponding author), School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China; emails: {chenqijason, bolongzheng}@hust.edu.cn; N. Q. V. Hung, School of Information and Communication Technology, Griffith University, Gold Coast, Australia; email: henry.nguyen@griffith.edu.au; P. Zhou, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan, China; email: panzhou@hust.edu.cn; G. Liu, Department of Computing, Macquarie University, Sydney, Australia; email: guanfeng.liu@mq.edu.au.

1559-1131/2020/02-ART5 $15.00
https://doi.org/10.1145/3375548

ACM Reference format:
Guohui Li, Qi Chen, Bolong Zheng, Nguyen Quoc Viet Hung, Pan Zhou, and Guanfeng Liu. 2020. Time-aspect-sentiment Recommendation Models Based on Novel Similarity Measure Methods. *ACM Trans. Web* 14, 2, Article 5 (February 2020), 26 pages.
https://doi.org/10.1145/3375548

1 INTRODUCTION

The recommendation system has been extensively studied by the data mining community in recent years and is now widely deployed for a variety of e-business services, location-based services, news feeds, and so on. As one of the most important applications, the recommendation system provides a user with a personalized list of suitable items based on his/her purchase records. It is effective in relieving the information overload caused by massive volume of items and stimulating the user's shopping desire with customized suggestions.

For many modern e-commerce websites such as Amazon,[1] Taobao,[2] Ebay,[3] and so on, the item recommendation plays a crucial part in improving the user's experience and boosting sales, which has gained a lot of attention in both business and academia. To better meet a user's personal need, a series of approaches [1, 2, 4, 10] have been proposed for accurately predicting his/her interest on different items, which mainly focus on modeling the user's historical behaviors (e.g., browsing history, rating, comment).

As one of the most popular approaches for personalized recommendation, the matrix factorization characterizes both items and users by vectors of factors inferred from user-item rating patterns, which has been studied in recent years by combining good scalability with predictive accuracy [5, 16, 24]. However, we observe that the user's preference and item profile always change over time in a dynamic manner; it is thus inappropriate to treat them as static information in real-world applications. To solve this problem, many time-aware methods have been proposed. Ding et al. [9] compute the time weights for historic data on the principle that the impact of recent ratings are more important than old ones. Koren et al. [15] consider the temporal dynamics in matrix factorization recommendation models by involving time-dependent bias. Liang et al. [37] study the variation of the user's preferences over times. Zhang et al. [41] propose a feature-level time series model for predicting user's daily-aware preference by exploiting the seasonal variance derived from purchasing data. However, these methods show limited performance on the long time span recommendation task, especially the recommendation on electronic items, due to the following reasons:

(1) Timeliness. The content-based models [24] only aim at recommending the items that are similar with the ones in a user's purchase history, therefby ignoring the fact that electronic items (e.g., *laptop* or *smartphone*) always update frequently. For instance, when recommending a smartphone, it is obviously inappropriate to suggest a similar one to the user's smartphone that was bought several years ago;

(2) Availability. The time-aware models [9] generally rely on an assumption that the recent information (e.g., the last purchased items) is more useful, and thus the recent purchase records have higher weights. As a result, they assign small weights for the purchase records of a long time ago or even directly discard them, which causes the low availability of the historical purchase records in recommendation.

[1] https://www.amazon.com/.
[2] https://www.taobao.com/.
[3] http://www.ebay.com/.

Fig. 1. Description of a laptop on Amazon.

(3) Relevance. The latent factor models [5] fail to exploit the relationships between the latent factors and explicit features, which cause deviation on the evaluation of latent factor matrix.

To this end, we study how to construct an effective model to extract useful information from the historical purchase records that are not recently produced. Figure 1 is a laptop description page on Amazon, and it shows some aspects (i.e., features) of the laptop that jointly decide its competitive strength in the market. In general, the competitive strength of a laptop always declines over time, since the consumers' increasing performance requirements give rise to the configuration update. As the purchasing cycles of electronic items are usually more than one year, the latest purchased items are not always useful due to the significant changes over time.

To characterize the nature of temporal dynamics, we propose to model on two different perspectives, i.e., ***item level*** and ***user level***. The former aspect reflects the item grade while the latter captures the stability of the user's purchase intention. Based on these two characteristics, novel similarity measure methods are adopted in our models. To the best of our knowledge, this article is the first to address the issue of timeliness and propose two novel methods—**I**tem **L**evel Similarity **M**atrix **F**actorization (ILMF) and **U**ser **L**evel Similarity **M**atrix **F**actorization (ULMF)—by extending the latent factor model. We combine these two methods and further propose **I**tem-**U**ser **L**evel Similarity **M**atrix **F**actorization (IULMF) to study which of these two methods contributes more to the final performance improvement. Our contributions can be summarized as follows:

- This article studies the problem of long time span recommendation, which is not well supported by the conventional recommender systems, since most of them do not consider the specific nature of the items, which can have a major impact on recommendations. Hence, we investigate the patterns of item aspect variances and user consumption behaviors, then propose the item-level and user-level recommendation models.
- We design efficient methods to handle the timeliness problem by mining users' consumption levels from their purchase records and propose novel similarity measure methods.
- We exploit the consistency between the latent factors and the explicit factors, then design an optimization function as an auxiliary part to improve the quality of item recommendation.
- Finally, we evaluate the proposed recommendation methods on real datasets. Experimental results show that our proposed approaches exhibit superior performance against other baselines in terms of both the precision and NDCG.

The rest of this article is organized as follows: Section 2 reviews the related work on existing recommendation methods. Section 3 presents the analysis on the real datasets. Section 4 proposes the framework of our methods. In Section 5, a set of experiments is conducted on real datasets from Amazon, and then we give an analysis on the experimental results. Section 6 gives the conclusion and future work.

2 RELATED WORK

The recommendation system is a subclass of the information filtering system that seeks to predict the users' interests on the items. Senecal et al. [32] investigate the influence of online recommendation on consumers' product choices. It shows that there exists a close correlation between the item recommendation and the users' subsequent consumption behaviors, which proves the value of recommendation system in sales promotion.

Nowadays, many websites collect a lot of useful information such as item description, feedbacks, and so on, which provides support for building highly efficient recommendation systems. A probabilistic recommendation model is proposed by Reference [10], which exploits item descriptions to model commonalities, variants, and cross-category features. Hai et al. [11] propose a supervised probabilistic graphical model to select the most helpful reviews at a fine granularity, in which they jointly model the aspects and sentiments of the reviews. Some existing work considers the specific nature of the items. Reference [41] studies the cyclicity and seasonality of the products in the cosmetic domain. Different from electronic products, the cosmetics are bought frequently and they do not depreciate in general. Reference [35] exploits the temporary features of the movies to quickly select the high-quality candidates. Reference [19] considers the geographical position in location recommendation. Up to now, many efforts have been dedicated to this topic. Next, we review the related work in the following three main categories:

Content-based Filtering. The content-based approach recommends items with an in-depth analysis on the item features. The early work is proposed in Reference [14], which recommends new web pages through analyzing the pages that the user has browsed. Mooney et al. [24] propose a bag-of-word–based naive Bayes text classifier for recommendations with insufficient ratings. However, these approaches suffer a common drawback: The newly added items usually are not used for recommendation. To alleviate the problem, Chen et al. [12] study an automatic recommendation method based on the proposed concept "influent set."

Collaborative Filtering Model. The Collaborative Filtering (CF) techniques can be roughly classified into two categories: *memory*-based CF and *model*-based CF. The memory-based CF [2] methods can be further divided into user-based CF [4, 29] and item-based CF [31]. The user-based CF makes recommendations for the user based on the similar users' purchasing choices. In contrast, the item-based CF directly recommends the items that are similar with the item that the user has purchased. As one of the most representative model-based methods, matrix factorization approaches have gradually become prevalent techniques for recommendation [16, 26]. The Latent Factor Model (LFM) [34] predicts the preference of a user on a given item based on two latent low rank matrices factorized from the explicit user-item rating matrix. Many variants such as the Non-negative Matrix Factorization (NMF) [17], Extended Tensor Factorization Model (ETFM) [23], Probabilistic Matrix Factorization (PMF) [30] and so on, are proposed and all achieve good results. In addition, Zhang et al. [39] propose a Singular Value Decomposition (SVD)–based method to handle the sparsity problem. Koren et al. [15] propose timeSVD++ to extend SVD matrix factorization by modeling the time changing behavior throughout the life span of data. timeSVD++ models the change over time, while our work aims to model the stability over time. Different from timeSVD++,

we exploit the relationships of items purchased in different periods and study the stability of the user's preference level, which paves a way to recommend with few historical records.

Comprehensive Model. Motivated by the increasing temporary information in commercial websites and apps, some comprehensive models are proposed. Tang et al. [35] investigate the effect of temporary features of the movies and mainly consider the production year of the movie in recommendation. Yoon et al. [38] consider that a user's preference change dynamically over time and recommend music based on the time-weighted clustering technique. With the continuous growth of feedback resources, the problem of how to use textual reviews in the recommendation systems has received increasing attention over the past few years. HFT [21] combines the latent factor (hidden factor) model with topic model, which assumes that the number of latent factors is the same as the number of topics. However, it is possible in some real situations that the number of latent factors is larger than the number of topics. For instance, the advertisement is usually a latent factor that stimulates a user to choose an item, but it is rare to see that the user mentioned the advertisement in the review. In addition, more implicit information can be captured if a larger number of latent factors is available, which is beneficial to improve the recommendation quality. Therefore, we set the number of latent factors larger than the number of topics in our extended latent factor model. FPMC [28] integrates matrix factorization with Markov chain based on personalized transition matrix. FPMC shows the predicting superiority on long-sequence data, since long-sequence data are helpful for learning an accurate transition matrix, which has an important influence on the prediction result. Different from FPMC, our model can also deal with the short-sequence data. Zhang et al. [40] present an Explicit Factor Model (EFM) to generate explainable recommendation according to the phrase-level sentiment analysis on the aspects of items. EFM incorporates both user-feature and item-feature relations as well as user-item ratings into a new unified hybrid matrix factorization framework. Different from EFM, we consider the relationship between explicit factors and implicit factors and take the user similarity and item similarity into latent factorization model. Diao et al. [8] propose a hybrid approach to jointly model the aspects, ratings, and sentiments for movie recommendation. Zhao et al. [42] take the sentiments on aspects of category-specific POIs (Points-Of-Interest) into consideration. Bauman et al. [3] use sentiment utility logistic model to recommend the items on a fine-grained aspect level.

However, all the previous work fails to consider the variation of the item features and the stability of the user's preference over time. To this end, this article first exploits the user's purchase behaviors from the perspective of user level and item level and proposes a unified framework based on two similarity measure methods.

3 DATA ANALYSIS

In this section, we first introduce our motivation of modeling the item level and user level. Then, we analyze the characteristics of item aspect variances and user consumption behaviors. Finally, we briefly present a general framework of matrix factorization.

3.1 Empirical Data Analysis

As is well-known, the electronic items such as smartphones, laptops, cameras, and so on upgrade frequently but the consumers update their items infrequently. For example, Apple launches a new iPhone every year, but a consumer usually updates his smartphone more than every two years. Consequently, due to the relatively long purchase cycle, the enterprises are difficult to acquire the up-to-date information from the users' purchase records for recommending new items. In contrast to the frequently purchased commodities, the construction of a recommendation system on these infrequently purchased items has to meet more timeliness requirements.

Table 1. An Example of User's Purchase History

Brand	CPU	RAM	Hard Disk	Weight	OS	Purchase Date
HP DV9334US	Intel T5300 1.7 GHz	2 GB	160 GB	7.7 pounds	Windows Vista	2008/05/22
Acer Nitro VN7	Intel Pentium B960 2.2 GHz	4 GB	500 GB	7.0 pounds	Windows 8	2013/06/22

Traditional feature-based methods aim at recommending items that share similar aspects with the ones bought in the past by the user, which makes it difficult to recommend suitable items based on the purchase records from a long time ago. Table 1 shows a real example that a consumer purchased two laptops within an interval of nearly five years. As we can see, the Acer Nitro VN7 has much improvements over some aspects, such as CPU, RAM, Hard Disk, and so on, compared to the HP DV9334US. However, the feature-based methods would not recommend the Acer Nitro VN7 to the consumer, since it is not similar to the HP DV9334US in many aspects. In general, it is inappropriate to recommend a new laptop that shares similar aspects with the old one as the five-years-ago configuration is obviously outdated. However, due to the fact that the consumer did not update his laptop frequently, there are usually only a few historical records available for analyzing the consumer preference.

In this article, we deal with the long time span recommendation problem when only a limited number of purchase records from a long time ago are available. The key to solving this problem is to seek the relevance between these records to improve the availability of the past purchase records in recommendation. To achieve this goal, we first attempt to exploit the inner links between the items purchased in different periods.

In general, the electronic items can be divided into different levels according to their configurations. Normally, the higher-level items have better configurations in different aspects. Therefore, the expert knowledge can be utilized to compare the aspects of different items to determine the aspect level for each item, which benefits the analysis on the consumer's preference. For instance, if a consumer chooses a laptop equipped with a top-level i7 CPU, it reflects that he prefers strong computation power. Through analyzing the data collected from Amazon, we obtain the observations with the help of expert knowledge on laptop: (1) the new generation laptops show different degrees of improvements on different aspects; (2) the consumers' purchase records show the stabilities of their consumption habits to some extent. These characteristics are summarized in general, which are independent of any particular item and user. Here, we take RAM as an example to illustrate our observations.

Figure 2 depicts the RAM size distribution of the laptops on the market from 2012 to 2015. X axis is the RAM size and Y axis is the corresponding ratio of laptops in a one-year period. We observe that the laptops equipped with 4 GB RAM account for the largest proportion during 2012 to 2014. As time goes on, the ratio of laptops with 4 GB RAM continues to rise until it reaches the top in 2014 and then begins to drop. During 2012 to 2015, the ratio of laptops with 8GB RAM continues to rise. Conversely, the ratio of laptops with 2 GB RAM shows a continuous decrease and reduces to a quite low value in 2015. The ratio variances of laptops with different RAM sizes clearly show the improvements of the laptop RAM over time.

By analyzing the variation of RAM size, we aim to find the connection between the two laptops in Table 1. When comparing the two laptops with the other items in the same period, we find that they are both equipped with the mainstream configurations at the time they are purchased. Assume that the laptops are roughly divided into three levels, i.e., low-end, middle-end, and high-end; the two laptops are both considered as middle-end items. That is to say, although there exists a significant difference on the common aspect of two items, they can still be regarded as the items at the same level, which may well explain why the consumer purchased them in different periods.

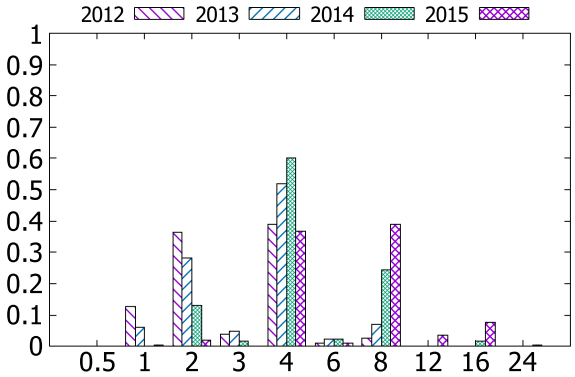


Fig. 2. Distribution of laptop RAM size.

In References [22, 33], the concept "consideration set" is proposed to model the purchase alternatives that a consumer considers positively, which reflects his purchase intention. Although the consideration set could be affected by some external factors, such as brand, discount, and advertising, it is still stable, since a user has a relatively stable purchase intention. For example, it is unsuitable to recommend a new iPhone to a user who always chooses cheap smartphones in the past. Combined with the analysis of some users' purchase records, we make a reasonable presumption that a user generally tends to buy items at the same level or similar level. Motivated by this presumption, we try to incorporate the above characteristics into a traditional latent factor model to improve the recommendation accuracy.

3.2 Preliminaries: Latent Factor Model

Latent Factor Model (LFM) is a recommendation method based on matrix factorization. It predicts each user's potential preference on the unpurchased items by only considering the existing ratings that the user gave on his purchased items. The recommendation task can be summarized as follows: given the purchased records of $|\mathcal{U}|$ users over $|\mathcal{P}|$ items, for a target user, it recommends a list of items of interest that has not been purchased before. For better understanding, we next introduce the details of LFM.

A preference matrix $R \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{P}|}$ is used to describe users' preferences on items, where each element R_{ij} denotes a user u_i's rating on an item p_j. If u_i does not buy p_j, then R_{ij} is set to 0. It is worth noting that $R_{ij} = 0$ does not mean that u_i is not interested in p_j. The reason may be that u_i does not know p_j. The basic principle of latent factor model is mapping both users and items into a shared space with dimension d, i.e., the latent factor number is d. $V \in \mathbb{R}^{|\mathcal{U}|\times d}$ and $Q \in \mathbb{R}^{|\mathcal{P}|\times d}$ are used to denote the latent factor matrices for the users and the items, respectively. For simplicity, we directly use V_i and Q_j to denote the ith row of V and the jth row of Q, respectively. The preference of u_i on p_j can be fitted using

$$\widehat{R}_{ij} = V_i Q_j^T. \tag{1}$$

The model parameters V and Q can be learned by minimizing a weighted regularized square error loss,

$$\min_{V,Q} L = \frac{1}{2}\sum_{i,j}(R_{ij} - \widehat{R}_{ij})^2 + \frac{\lambda}{2}(\|V\|_F^2 + \|Q\|_F^2), \tag{2}$$

where $\|.\|_F$ is the Frobenius norm of a matrix and λ is the regularization parameter. In the training process, we select the same number of positive and negative samples according to Reference [25]. The sampling process for a user u_i is as follows:

(1) Choose an item p_j as a positive sample when the corresponding rating $R_{ij} > 0$.
(2) Choose an item p_c as a negative sample when the corresponding rating $R_{ic} = 0$ and it obeys the rule: p_c is not similar with p_j, and p_c is popular.
(3) Repeat steps (1) and (2) until all the positive elements are sampled.

In step (2), in addition to R_{ij}, we also use the number of reviews of each item in the negative sampling process. We define the popularity of each item by the number of reviews it has and use a threshold to determine whether a sample is similar with the given item p_j. To define the similarity, we use a purchased vector to record the users that have purchased p_j in the past. For each element i, 1 represents that u_i has purchased p_j, otherwise 0 represents that u_i has not purchased p_j. The similarity between any two items is calculated by the cosine similarity of their purchased vectors. We test several parameter combinations in our experiment and observe that a suitable negative sample p_c should not be one of the top-20 most similar items w.r.t. p_j. Meanwhile, p_c is popular if it has more than a constant number of reviews. In our experiments, we choose 100 as this constant number in both the Laptop and Smartphone datasets. Once the latent factors V and Q have been learned, the preference value associated with any user-item pair (u_i, p_j) can be predicted by using Equation (1).

Although LFM utilizes the purchased records in personalized recommendation, it does not make full use of the available information in these records. A purchase record usually contains the item ID, the rating, and the review. However, LFM only considers the ratings (i.e., it directly uses the user's rating to represent his/her preference on the item), while neglecting other useful information, such as the reviews. To overcome this issue, we extend LFM by incorporating the review information in Section 4.

4 FRAMEWORK

In our model, we incorporate multi-source information into building the preference matrix: (1) purchase records; (2) item description pages. In the purchase record, in addition to ratings, we also use the review information. For the item description pages, we use them to extract the item aspect information.

4.1 Notation and Problem Definition

Let $\mathcal{U} = \{u_i\}_{|\mathcal{U}|}$ and $\mathcal{P} = \{p_j\}_{|\mathcal{P}|}$ denote user set and item set, where $|\mathcal{U}|$ and $|\mathcal{P}|$ are the user number and item number. The review set is denoted by $\mathcal{G} = \{g^i_{jm}\}_{|\mathcal{G}|}$ ($|\mathcal{G}|$ is the number of reviews), where g^i_{jm} refers to the review written by user u_i when he/she purchases the item p_j at time t_m. Accordingly, $\mathcal{G}_{u_i}$ denotes the set of the reviews released by the user u_i and $\mathcal{G}_{p_j}$ denotes the set of the reviews on item p_j. Suppose all the items in the same domain share K common aspects and we use a K-dimensional vector to denote the aspects of item p_j, where p_{jk} denotes kth aspect a_k of p_j. The purchase history of user u_i is denoted by $\mathcal{P}_{u_i} = \{p_{u_i,t_1}, p_{u_i,t_2} \ldots, p_{u_i,t_m} \ldots, p_{u_i,t_{|\mathcal{P}_{u_i}|}}\}$. Different from LFM that only considers the ratings information, our model also considers the reviews and the item aspect information when it recommends new items to each user u_i. In our model, the goal is to estimate the final latent factor matrix V and Q by minimizing optimization function L as follows:

$$\min_{V,Q} L = \frac{1}{2}\Gamma(V, Q) + \frac{\lambda}{2}(\|V\|_F^2 + \|Q\|_F^2) + \frac{\beta}{2}\Phi(Q). \tag{3}$$

$\Gamma(V, Q)$ is the main part of our optimization function that models the item level and user level, respectively. $\Phi(Q)$ is the auxiliary part that captures the consistency between explicit and implicit features. $\frac{\lambda}{2}(\|V\|_F^2 + \|Q\|_F^2)$ is used for regularization. The major notations used in this article are summarized in Table 2. In the following, we introduce the details of our extended models based

Table 2. Notations in the Framework

$\mathcal{U}$	User set with $\lvert\mathcal{U}\rvert$ users
$\mathcal{P}$	Item set with $\lvert\mathcal{P}\rvert$ items
$\mathcal{G}$	Review set with $\lvert\mathcal{G}\rvert$reviews
$\mathcal{T}_q$	qth time bin
$\mathcal{P}_{\mathcal{T}_q}$	A subset that all the items are purchased in $\mathcal{T}_q$
$X \in \mathbb{R}^{\lvert\mathcal{U}\rvert\times K}$	The user attention matrix
$Y \in \mathbb{R}^{\lvert\mathcal{P}\rvert\times K}$	The item sentiment rating matrix
$R \in \mathbb{R}^{\lvert\mathcal{U}\rvert\times\lvert\mathcal{P}\rvert}$	The rating matrix

Table 3. An Example of User's Review

ASIN: B00111SB80	**Text:** Perfect for email, web browsing, basic photos, basic videos, music, and word processing. Probably not powerful enough for high end graphics, video editing, or gaming. But, plenty capable for everyday computing.
Author: kc_keg	I've used it to edit and create a home video DVD and it worked very well. What else can I say? For under $500 (Home Depot) this is a great value.
Star: 5	The screen looks great and aspect set is the best I've seen for the price. Update (7/21/08): I've had this for about six months now... no problems.
Date: 2008/12/22	It's still a nice computer for the money. I made a great 1 hour DVD for my daughter's one year birthday using a ton of high quality pics, videos, and songs and it handled it like a champ.

on novel similarity measure methods. First, we will introduce the details of sentiment analysis in the preprocessing phase.

4.2 Sentiment Analysis

To incorporate sentiment into our models, two matrices based on the reviews are generated by means of sentiment analysis techniques, i.e., item sentiment rating matrix and user attention matrix. Table 3 shows the type of a user's real review, where ASIN is the ID number of the item on Amazon. Next, we introduce how to mine users' opinions from their reviews and propose relevant definitions.

The aspect extracting has been well studied [18, 27, 36]. In this article, we only focus on a small number of aspects that appear in item description, so we use the basic LDA [6] model to acquire the word distributions of different topics. Those words expressing the same facet are automatically grouped together under the same topic. An aspect can be expressed by a set of words characterizing a topic in the given domain, which can be features of items or attributes of services. We manually select the appropriate words under the corresponding topics as aspect words. The aspect extracting process is as follows:

(1) We put all the review texts into a document and then preprocess the document by removing the stop words and low-frequency words (appear less than three times).
(2) We use LDA model to generate the topic and the topic words. A topic is represented by a probability distribution over the topic words. The higher probability means the word is more relevant to the topic.
(3) Through analyzing the topic word distribution for each topic, we manually select the aspect words and remove the bad topic words that are highly irrelevant.

Table 4. Example of Aspects

Aspect	Aspect word
CPU	processor, cpu, computing, clockspeed, compute, performance
Screen	screen, display, inch, size
RAM	memory, ram
Operation system	system, os, windows, vista, win7, window,
Graphic	graphic, game, gaming, fan, photo
Harddisk	harddisk, disk, storage
WiFi	wi-fi, wifi, wireless
Weight	weight, pounds
Battery	battery, endurance, duration
Heat	ventilator, overheat, heat
...	...

In step (2), a real word distribution on a topic generated by LDA in gensim toolkit is as follows: 0.079*"screen"+0.048*"issue"+0.047*"inch"+0.046*"spot"+0.043*"efficiency"+0.039*"display" +0.038*"pixel"+0.013*"problem"+0.012*"cam"+0.011*"shape." In step (3), observing that most of the words characterize the topic "screen," we consider "screen" as the aspect, then, we manually remove the words "issue," "efficiency," "problem," and "cam," which are not relevant to the aspect "screen." The reason why some irrelevant words appear in the topic is that the LDA utilizes the concurrence of the words. For instance, many users complain that there exists a problem with their laptop screens, which causes "screen" and "problem" appear in the same topic. Table 4 shows some aspects and the corresponding aspect words, and we use $\mathcal{H}_k$ to denote the set of aspect words mapping to aspect a_k.

Next, we construct a sentiment lexicon $\mathcal{S}$ from textual user reviews by using the method of Reference [20], and the general-purpose sentiment lexicon that is used to construct $\mathcal{S}$ is SentiWordNet 3.0.[4] Then, we analyze users' opinions by following Reference [13] and first extract the aspect-opinion entry (h, o) from the users' reviews, where $h \in \mathcal{H}_k$ is an aspect word to express a_k and o is an opinion word. For example, from the review text "The screen from the seller is excellent, but the processor is weak," the entries (screen, excellent) and (processor, weak) could be extracted. Through consulting the lexicon $\mathcal{S}$, we can acquire the sentiment polarity of entry (h, o) according to the opinion word o. The value +1 represents the entry is positive, −1 for negative, and 0 for neutral. Then, negation words detection is conducted to check whether the sentiment of each entry is reversed. In this example, we acquire the aspect-sentiment entries (screen, excellent, +1) and (processor, weak, −1).

The user's overall sentiment on aspect a_k of item p_j is calculated by summing up the sentiment of all the entries extracted from the reviews on p_j, i.e., $s_{jk} = \frac{1}{N_{p_j}} \sum_{h \in \mathcal{H}_k} (h, o)$, where N_{p_j} is the number of entries. We use Y to denote the synthetic rating matrix, and each element Y_{jk} is defined as follows:

$$Y_{jk} = \frac{r_j}{1 + e^{-s_{jk}}}, \tag{4}$$

where r_j is the average rating of p_j, which captures the user's overall evaluation and s_{jk} reflects the user's opinion on the specific aspect a_k of p_j. To model a user's attention, we use a K-dimension vector to describe his attention degree on K aspects of the items. In accordance with Reference [40],

[4] http://sentiwordnet.isti.cnr.it/.

we use X_{ik} to denote the attention of user u_i on aspect a_k as follows:

$$x_{ij,k} = \begin{cases} 1 & \text{if } f_{ij,k} = 0 \\ 1 + (R_{ij} - 1)\left(\frac{2}{1+e^{-f_{ij,k}}} - 1\right) & \text{if } f_{ij,k} \neq 0 \end{cases}, \tag{5}$$

$$X_{ik} = \frac{\sum_{p_j \in \mathcal{P}_{u_i}} x_{ij,k}}{|\mathcal{P}_{u_i}|}, \tag{6}$$

where $f_{ij,k}$ counts the times that u_i mentions a_k in the review on p_j published by u_i, i.e., the number of aspect-opinion pairs (h, o) when $w \in \mathcal{H}_k$, and $|\mathcal{P}_{u_i}|$ denotes the number of items purchased by u_i. R_{ij} is the rating u_i gives on p_j, which is always a fivestar reviewing system in the real-world, such as Amazon and Yelp. For an aspect that is not mentioned by the user, it is still assigned a basic value 1. Each user's attention vector is normalized, i.e., $\sum_{k=1}^{K} X_{ik} = 1$.

4.3 Item Aspect Level

To better compare the items in different periods, we follow Reference [19] to partition the continuous time into discrete bins—namely, ***time bin***—to study the user's visiting behavior according to a metric (e.g., *week*-based, *month*-based).[5] We test different time bin lengths, and the length of time bin is denoted by b. Consequently, the continuous time is divided into a set of time bins, i.e., $\mathcal{T} = \{\mathcal{T}_q\}_{|\mathcal{T}|}$, where $\mathcal{T}_q$ denotes the time bin $[t_{q-1}, t_q]$ and $|\mathcal{T}|$ is the number of time bins. Accordingly, we divide the items into $|\mathcal{T}|$ groups according to the time they appear on the market, and $\mathcal{P}_{\mathcal{T}_q}$ denotes the set of items that appear on the market during $\mathcal{T}_q$.

Next, we select the aspects that are suitable for comparison, i.e., RAM, hard disk, and weight. With the help of expert knowledge about laptop,[6] any two items can be preliminarily compared on these aspects. The traditional measure method simply compares the aspect values between items; however, the time effect on aspect is not considered. For instance, the level of a 4 GB RAM laptop declines over time from 2013 to 2015, since more laptops are equipped with larger RAM sizes in 2015. Therefore, the traditional measure method cannot capture such information, and we propose a novel comparison strategy based on the aspect level. Instead of using a fixed value, we employ a pair of variational ratios to better show the level of aspect. Given an item $p_j \in \mathcal{P}_{\mathcal{T}_q}$, we use two boundary values to describe its relative *aspect level*, where lower boundary p^l_{jk} denotes the probability of items in $\mathcal{P}_{\mathcal{T}_q}$ that are inferior to p_j on a_k, and upper boundary p^u_{jk} denotes the probability of items in $\mathcal{P}_{\mathcal{T}_q}$ that are superior to p_j on a_k, respectively:

$$\begin{aligned} p^l_{jk} &= Pr(p_{ik} \prec p_{jk} | p_i \in \mathcal{P}_{\mathcal{T}_q}) = \frac{N^{inf}(p_j)}{|\mathcal{P}_{\mathcal{T}_q}|} \\ p^u_{jk} &= Pr(p_{ik} \succ p_{jk} | p_i \in \mathcal{P}_{\mathcal{T}_q}) = \frac{N^{sup}(p_j)}{|\mathcal{P}_{\mathcal{T}_q}|} \end{aligned}. \tag{7}$$

In Equation (7), $N^{inf}(p_j)$ is the number of the items in $\mathcal{P}_{\mathcal{T}_q}$ that are inferior to p_j on a_k, and $N^{sup}(p_j)$ is the number of the items in $\mathcal{P}_{\mathcal{T}_q}$ that are superior to p_j on a_k. $|\mathcal{P}_{\mathcal{T}_q}|$ is the total number of the items in $\mathcal{P}_{\mathcal{T}_q}$.

Table 5 presents an example of comparing different items at different times from the perspective of item level, where $p_1 \in \mathcal{P}_{2012}$, $p_2 \in \mathcal{P}_{2015}$, and $p_3 \in \mathcal{P}_{2015}$. The lower boundary and upper boundary of RAM size are listed in column 3 and column 4, respectively. Traditional content-based methods directly recommend the items with similar aspect configurations. If we directly calculate the difference values of these items by comparing their RAM sizes, then p_2 is more

[5]For ease of presentation, *time* and *time bin* are used interchangeably unless noted otherwise in this article.
[6]Expert knowledge obtained from professional website PassMark https://www.passmark.com/.

Table 5. An Example of Item-level Comparison

Item	RAM(GB)	Lower (p^l_{jk})	Upper (p^u_{jk})
$p_1 \in \mathcal{P}_{2012}$	8	0.73	0.10
$p_2 \in \mathcal{P}_{2015}$	8	0.41	0.32
$p_3 \in \mathcal{P}_{2015}$	16	0.62	0.11

similar to p_1 than p_3. Let us take Euclidean distance, for example: The difference value between p_1 and p_2 is $\sqrt{(8-8)^2} = 0$ and the difference value between p_1 and p_3 is $\sqrt{(16-8)^2} = 8$. However, if we use boundary tuple to recalculate the difference values, then the difference value between p_1 and p_2 is $\sqrt{(0.73-0.41)^2 + (0.10-0.32)^2} \approx 0.388$ and the difference value between p_1 and p_3 is $\sqrt{(0.73-0.62)^2 + (0.10-0.11)^2} \approx 0.110$. The former comparison strategy is not suitable for time-sensitive item recommendation, and it by default considers that the close configuration information is similar. In real-world experience, there are not many laptops equipped with 8 GB RAM in 2012, and it outperforms most of the laptops at that time. However, in 2015, the 8 GB RAM becomes a common configuration that is no longer treated as a high-end item. Considering the dynamic nature of the laptop market, the latter one adopts a new standard that better reflects the characteristics of items and lays the foundation of modeling users' consumption intention.

4.4 Item-level Similarity

Based on the definition of item aspect level, we define the item-level similarity. Assume we have two items $p_j \in \mathcal{P}_{\mathcal{T}_{q_1}}$ and $p_c \in \mathcal{P}_{\mathcal{T}_{q_2}}$. We use $\hat{\mathcal{T}}_{q_1}$ and $\hat{\mathcal{T}}_{q_2}$ to denote the midpoints of time bins $\mathcal{T}_{q_1}$ and $\mathcal{T}_{q_2}$, respectively, and use $|\hat{\mathcal{T}}_{q_1} - \hat{\mathcal{T}}_{q_2}|$ to approximatively express the difference of the launch dates of p_j and p_c. The similarity between p_j and p_c is inversely proportional to the difference of their launch time. Moreover, the reviews contain the feedback information on the aspects of items provided by the consumers, and the user's attention reflects his personal preference on the aspects, which both affect the item-level similarity from the user's point of view. To better calculate the similarity, the user's attention and synthetic ratings on the aspects derived from the reviews are also considered. Therefore, we use weighted cosine similarity to calculate the item-level similarity between p_j and p_c under u_i's attention as follows:

The item-level similarity $PSim(p_j, p_c, X_i)$ is inversely proportional to the difference between $\mathcal{T}_{q_1}$ and $\mathcal{T}_{q_2}$. We use exponential function to model the inverse proportion relation instead of factorial function to avoid the case that the denominator is 0. w^k_{jc} is the weight between p_j and p_c on aspect a_k, where X_{ik} is u_i's attention on aspect a_k and $e^{-\varepsilon_1 X_{ik}|Y_{jk}-Y_{ck}|}$ captures the difference of two items' synthetic ratings. $e^{\varepsilon_2 * |\hat{\mathcal{T}}_{q_1} - \hat{\mathcal{T}}_{q_2}|}$ models the time effect based on the fact that the items have high similarity when they appear on the market at close time. $CSL(p_j, p_c, X_i)$ and $CSU(p_j, p_c, X_i)$ calculate the weighted cosine similarity of lower boundary and upper boundary of two items, respectively. Finally, $PSim(p_j, p_c, X_i)$ uses e^{-x} to rescale the similarity in [0,1], and the smaller value of $PSim(p_j, p_c, X_i)$ means that p_j and p_c are more similar:

$$
\begin{aligned}
w^k_{jc} &= e^{-\varepsilon_1 X_{ik}|Y_{jk}-Y_{ck}| - \varepsilon_2 |\hat{\mathcal{T}}_{q_1} - \hat{\mathcal{T}}_{q_2}|}, \\
CSL(p_j, p_c, X_i) &= \frac{\sum_{k=1}^{K} w^k_{jc} p^l_{jk} p^l_{ck}}{\sqrt{\sum_{k=1}^{K} w^k_{jc} p^l_{jk}} \sqrt{\sum_{k=1}^{K} w^k_{jc} p^l_{ck}}}, \\
CSU(p_j, p_c, X_i) &= \frac{\sum_{k=1}^{K} w^k_{jc} p^u_{jk} p^u_{ck}}{\sqrt{\sum_{k=1}^{K} w^k_{jc} p^u_{jk}} \sqrt{\sum_{k=1}^{K} w^k_{jc} p^u_{ck}}}, \\
PSim(p_j, p_c, X_i) &= e^{-CSL(p_j, p_c, X_i) - CSU(p_j, p_c, X_i)}.
\end{aligned}
\tag{8}
$$

4.5 User-level Similarity

Similar with item-level similarity, we also define the user-level similarity based on the user's preference level, which reflects the average level of items bought in the past. For simplicity, we directly use the average aspect level of items bought by the user to calculate his preference level, i.e., *user level*. For a user u_i, we use u^l_{ik} and u^u_{ik} to denote the lower boundary and upper boundary of u_i's preference level as follows:

$$\begin{aligned} w^{t_n}_{ij} &= \frac{e^{-\varepsilon_2(t_n - t^i_j)}}{\sum_{m=1}^{|\mathcal{P}_{u_i}|} e^{-\varepsilon_2(t_n - t_m)}}, \\ u^l_{ik} &= \frac{1}{|\mathcal{P}_{u_i}|} \sum_{p_j \in \mathcal{P}_{u_i}} w^{t_n}_{ij} p^{l,t^i_j}_{jk}, \\ u^u_{ik} &= \frac{1}{|\mathcal{P}_{u_i}|} \sum_{p_j \in \mathcal{P}_{u_i}} w^{t_n}_{ij} p^{u,t^i_j}_{jk}. \end{aligned} \tag{9}$$

Since the recent purchase records have higher weights, we use $w^{t_n}_{ij} = \frac{e^{-\varepsilon_2(t_n - t^i_j)}}{\sum_{m=1}^{|\mathcal{P}_{u_i}|} e^{-\varepsilon_2(t_n - t_m)}}$ to model the weight of the purchased item p_j, where t_n is the time to make recommendation and t^i_j is the time that u_i purchased p_j. We use $\sum_{m=1}^{|\mathcal{P}_{u_i}|} e^{-\varepsilon_2(t_n - t_m)}$ to normalize the weight, where t_m is the time when the mth item is purchansed. The smaller the time difference $t_n - t^i_j$ is, the higher weight the corresponding purchased item p_j has. $p^{l,t^i_j}_{jk}$ and $p^{u,t^i_j}_{jk}$ are the lower boundary and upper boundary of p_j purchased on t^i_j. We use $\mathcal{P}_{u_i}$ to denote the set of items purchased by u_i, and $|\mathcal{P}_{u_i}|$ denotes the number of items. Based on Equation (9), the user-level similarity between u_i and u_c is defined as follows:

$$\begin{aligned} w_{ic,k} &= e^{-\varepsilon_3 |X_{ik} - X_{ck}|}, \\ CSL(u_i, u_c) &= \frac{\sum_{k=1}^{K} w_{ic,k} u^l_{ik} u^l_{ck}}{\sqrt{\sum_{k=1}^{K} w_{ic,k} u^l_{ik}} \sqrt{\sum_{k=1}^{K} w_{ic,k} u^l_{ck}}}, \\ CSU(u_i, u_c) &= \frac{\sum_{k=1}^{K} w_{ic,k} u^u_{ik} u^u_{ck}}{\sqrt{\sum_{k=1}^{K} w_{ic,k} u^u_{ik}} \sqrt{\sum_{k=1}^{K} w_{ic,k} u^u_{ck}}}, \\ USim(u_i, u_c) &= e^{-CSL(u_i, u_c) - CSU(u_i, u_c)}. \end{aligned} \tag{10}$$

Except for the users' preference levels, the users have different attentions on the different aspects. To consider the effect of users' attentions, we use $e^{-\varepsilon_3 |X_{ik} - X_{ck}|}$ to denote the weight of the attention difference between u_i and u_c on aspect a_k. We use $CSL(u_i, u_c)$ and $CSU(u_i, u_c)$ to calculate the weighted cosine similarity of lower boundary and upper boundary of two users, respectively. Finally, $USim(u_i, u_c)$ uses e^{-x} to rescale the similarity in [0,1], and the smaller value of $USim(u_i, u_c)$ means that u_i and u_c are more similar.

4.6 Consistency Optimization Function

The aforementioned matrices X and Y indicate the observed relations of user-item and item-aspect, respectively. Here, we exploit how to integrate these explicit factors with latent factors in our model, which embodies the auxiliary optimization part $\Phi(Q)$. We use aspect information of items extracted from their description pages on Amazon to generate the explicit similarity matrix S by

using weighted cosine similarity. The weight used to rescale aspect a_k is defined as follows:

$$w_k = \frac{1}{a_k^{max} - a_k^{min}}, \tag{11}$$

where w_k captures the value range of a_k, and a_k^{max} and a_k^{min} are the maximum and minimum value of a_k, respectively. The explicit similarity between p_i and p_j is defined as follows:

$$S_{ij} = \frac{\sum_{k=1}^{K} p_{ik} w_k \times p_{jk} w_k}{\sqrt{\sum_{k=1}^{K} (p_{ik} w_k)^2} \sqrt{\sum_{1}^{K} (p_{jk} w_k)^2}}. \tag{12}$$

In this article, we propose implicit similarity that is generated by the latent factors and consider the consistency between explicit similarity and implicit similarity, i.e., if an item is similar to another one on their aspects, their latent factors are also similar and vice versa. The function $\Phi(Q)$ captures the difference between the explicit similarity matrix and the latent similarity matrix, which will be minimized in the optimization process. We use M to denote the implicit similarity matrix, and each element M_{ij} is the implicit similarity between p_i and p_j, which is defined as follows:

$$M_{ij} = \frac{Q_i Q_j^T}{|Q_i||Q_j^T|}. \tag{13}$$

Here, we arrange the items in the order of the time they appear on the market to pick out the items in the same period easily. For instance, as is depicted in Equation (14), S is the similarity matrix of four items. E_1 is an auxiliary matrix that picks out items p_1 and p_2, which are both in time period $\mathcal{T}_1$. S_1 is a matrix that records the similarity between p_1 and p_2:

$$E_1 \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \times S \begin{pmatrix} S_{11} & \dots & S_{14} \\ \vdots & \ddots & \vdots \\ S_{41} & \dots & S_{44} \end{pmatrix} \times E_1^T \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \rightarrow S_1 \begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix}. \tag{14}$$

The auxiliary matrix E_q is a $|\mathcal{P}_{\mathcal{T}_q}| \times |\mathcal{P}|$ matrix for generating the similarity matrix of items in $\mathcal{P}_{\mathcal{T}_q}$. And each row of E_q is a vector used to choose an item in $\mathcal{P}_{\mathcal{T}_q}$. The kth row vector of E_q is used to choose the kth item as follows:

$$\overrightarrow{E}_{qk} = (\underbrace{0, 0 \dots 0}_{k-1}, 1, \underbrace{0, 0 \dots 0}_{|\mathcal{P}_{\mathcal{T}_q}|-(k-1)})$$

As the items have been divided into $|\mathcal{T}|$ groups, there are $|\mathcal{T}|$ loss terms in the objective function. To estimate Q in the matrix factorization, a reasonable approach is to minimize the squared error of our objective function $\Phi(Q)$ as follows:

$$\Phi(Q) = \sum_{q=1}^{|\mathcal{T}|} \left\| E_q (M - S) E_q^T \right\|_F^2, \tag{15}$$

where $\|\cdot\|_F^2$ is the Frobenius norm. In our experiments, we only need to calculate the upper triangular part of the similarity matrices. Next, we will formalize our model and present solution methods.

4.7 Algorithms

Through mapping users and items into a shared latent space, the traditional latent factor model can make a preliminary estimation for the relations of items and users. We extend the traditional model by incorporating the item-level similarity and user-level similarity, respectively, and present the final optimization formulation as follows:

$$\begin{aligned}\min_{V,Q} L = \frac{1}{2}\sum_{i,j}(R_{ij} - \widehat{R}_{ij})^2 + \frac{\lambda}{2}(\|V\|_F^2 + \|Q\|_F^2) \\ + \frac{\beta}{2}\sum_{q=1}^{|\mathcal{T}|}\left\|E_q(M-S)E_q^T\right\|_F^2.\end{aligned} \tag{16}$$

The evaluated rating value $\widehat{R}_{ij} = (1-\alpha)V_iQ_j^T + \alpha D_{ij}$, and α is a balance parameter. $\frac{\lambda}{2}(\|V\|_F^2 + \|Q\|_F^2)$ is used to avoid over-fitting. Two extensions based on the item-level similarity and user-level similarity are presented as follows:

Item-level Similarity. To better introduce the extension based on item-level similarity, we use $\widehat{R}_{ij}^p$ and D_{ij}^p to replace $\widehat{R}_{ij}$ and D_{ij}, respectively,

$$D_{ij}^p = \frac{1}{Z_{p_j}}\sum_{p_c \in N(p_j)} PSim(p_c, p_j, X_i)V_iQ_c^T. \tag{17}$$

$N(p_j)$ denotes the set of ***Top-N*** most similar items with the given item p_j on aspect level under u_i's attention. Z_{p_j} is the normalizing factor, where $Z_{p_j} = \sum_{p_c \in N(p_j)} PSim(p_c, p_j, X_i)$.

ALGORITHM 1: ILMF

Input: $\lambda, \delta, \theta, \alpha, \beta, V, Q$
Output: R^p

Initialize V, Q, θ
while $k \le \delta$ **do**
  for *each* i, j **do**
    $B_{ij}^p = R_{ij} - \alpha D_{ij}^p - (1-\alpha)V_iQ_j{}^T$
    $D_{ij}^p = \frac{1}{Z_{p_j}}\sum_{p_c \in N(p_j)} PSim(p_c, p_j, X_i)V_iQ_c{}^T$
    $V_i \leftarrow V_i + \theta(B_{ij}^p(\frac{\alpha}{Z_{p_j}}\sum_{p_c \in N(p_j)} PSim(p_c, p_j, X_i)Q_c + (1-\alpha)Q_j) - \lambda V_j)$
    $Q_j \leftarrow Q_j + \theta((1-\alpha)B_{ij}^pV_i - \lambda Q_j$
    $-\beta\sum_{p_c \in \mathcal{P}_{\mathcal{T}_{qe}}}(Y_{cj} - S_{cj})\frac{Q_c}{|Q_c|}\frac{(Q_jQ_j^T)^{1/2}I_d - (Q_jQ_j^T)^{-1/2}Q_j^TQ_j}{(Q_jQ_j^T)})$
    $\theta = \theta * 0.95$
  end
  $k = k + 1$
end
Return R^p

In Algorithm 1, we take the partial derivative of L with respect to V_i and Q_j, respectively, i.e., $\frac{\partial L}{\partial V_i}$, $\frac{\partial L}{\partial Q_j}$, and obtain the updating rules for V_i and Q_j. E_{q_e} is the auxiliary matrix to choose the items in the same periods for an item $p_j \in \mathcal{P}_{\mathcal{T}_{q_e}}$. $\|E_{q_e}(M-S)E_{q_e}^T\|_F^2$ can be rewritten as $\sum_{p_c \in \mathcal{P}_{\mathcal{T}_{qe}}}(M_{cj} - S_{cj})^2$. I_d is an $d \times d$ identity matrix. The details of derivation steps are given in

the Appendix. We employ stochastic gradient descent to estimate the matrix V and Q, where δ is the iteration times and θ is the learning rate. After all the gradients are calculated, the algorithm returns the new evaluated rating matrix R^p. Then, we select *top-k* items according to the rating ranking and add them to the recommendation list.

User-level Similarity. The other extension is characterizing u_i's level by referring to other similar users. And we use $\widehat{R}^u_{ij}$ and D^u_{ij} to replace $\widehat{R}_{ij}$ and D_{ij}, respectively,

$$D^u_{ij} = \frac{1}{Z_{u_i}} \sum_{u_c \in N(u_i)} USim(u_c, u_i) V_c Q_j^T. \tag{18}$$

$N(u_i)$ denotes the set of ***top-N*** most similar users with the given user u_i. Z_{u_i} is the normalizing factor and $Z_{u_i} = \sum_{u_c \in N(u_i)} USim(u_c, u_i)$. We also use stochastic gradient descent to estimate V and Q by minimizing L. In our experiments, we set N to be 10. Algorithm 2 shows the details of the procedure. After calculation, we also recommend *top-k* items in the ranking list.

Item-User–Level Similarity. Moreover, we combine these two extensions and propose IUMLF to study which of these two parts contributes more to the final performance improvement. In this case, the evaluated rating is defined as $\widehat{R}_{ij} = (1 - \gamma_1 - \gamma_2) V_i Q_j^T + \gamma_1 D^p_{ij} + \gamma_2 D^u_{ij}$, where D^p_{ij} and D^u_{ij} are two extension terms mentioned above.

ALGORITHM 2: ULMF

Input: $\lambda, \delta, \theta, \alpha, \beta, V, Q$
Output: R^u

Initialize V, Q, θ
while $k \leq \delta$ **do**
  for *each* i, j **do**
    $B^u_{ij} = R_{ij} - \alpha D^u_{ij} - (1 - \alpha) V_i {Q_j}^T$
    $D^u_{ij} = \frac{1}{Z_{u_i}} \sum_{u_c \in N(u_i)} USim(u_c, u_i) V_c {Q_j}^T$
    $V_i = V_i + \theta((1 - \alpha) B^u_{ij} Q_j - \lambda V_j)$
    $Q_j = Q_j + \theta(B^u_{ij}(\frac{\alpha}{Z_{u_i}} \sum_{u_c \in N(u_i)} USim(u_c, u_i) V_c + (1 - \alpha) V_i)$
    $-\lambda Q_j - \beta \sum_{p_c \in \mathcal{P}_{\mathcal{T}_{q_e}}} (Y_{cj} - S_{cj}) \frac{Q_c}{|Q_c|} \frac{(Q_j Q_j^T)^{1/2} I_d - (Q_j Q_j^T)^{-1/2} Q_j^T Q_j}{(Q_j Q_j^T)})$
    $\theta = \theta * 0.95$
  end
  $k = k + 1$
end
Return R^u

5 EXPERIMENT

In this section, we evaluate the performances of proposed algorithms. We run our experiments on a computer with Intel core i5-3317U CPU@2.60 GHz and 8 GB RAM. The simulation programs are developed in Python. Then, we first introduce the datasets and the baseline methods.

5.1 Experimental Setting

We use two datasets collected from Amazon, i.e., Laptop dataset and Smartphone dataset. Each dataset contains two parts: user purchased records and item description information. Laptop

Table 6. Statistics of the Two Datasets

	Laptop	Smartphone
#Users	4,960	6,235
#Items	1,096	1,303
#Purchase records	13,296	21,053
#Purchase records per user	2.681	3.377
#Purchase records per item	12.131	16.157

Table 7. Item Aspect Information

Laptop	**Screen**	**CPU**	**RAM**	**OS**	**Weight**
	13 inch	Intel Core i5 6200U	8 GB	Windows 7	5.6 pounds
	Hard disk	**Graphic**			
	500 GB	Intel HD Integrated Graphics			
Smartphone	**Screen**	**CPU**	**RAM**	**ROM**	**Weight**
	5 inch	core 1.2 GHz Cortex-A7	1.5 GB	8 GB ROM	12.8 ounces

dataset consists of 4,960 users, 1,096 laptops, and 13,296 purchase records, and Smartphone dataset consists of 6,235 users, 1,303 laptops, and 21,053 purchase records. The statistics of the two datasets are shown in Table 6, and we remove the users whose number of purchase records is less than two. The laptops appeared on the market from June 2007 to December 2015, and the smartphones appeared on the market from January 2013 to December 2016.

We classify these laptops into groups according to the time that they appear on the market (i.e., available time in the description page). It is worth noting that we do not always acquire available time from the description pages, so we use the time that the item is first purchased to replace it. Sometimes, we cannot divide all the items according to time bins. When we set the length of time bin to be one year on Laptop dataset, the items purchased during June 2007 and December 2009 are divided into one group. The reason is that the number of laptops purchased before 2009 is small.

To extract the aspects by using LDA, we follow the method in Reference [7] to select the number of topics. We test different numbers of topics from 40 to 150 with an increase of 10, and finally choose 70 and 50 for the Laptop and Smartphone datasets that achieve the best performance, respectively. Table 7 shows an example of item aspects provided by the item description pages, and we manually choose seven aspects on Laptop dataset and four aspects on Smartphone dataset in our experiments. The aspects and corresponding aspect words are shown in Table 10 in the Appendix.

To evaluate the prediction performances of different models, we use two metrics: prediction precision (Prec) and normalized discounted cumulative gain (NDCG). Prec is the most important assessment criteria for a useful recommendation, and NDCG is a measure of average recommendation ranking quality. For both Prec and NDCG, a higher value indicates a better prediction performance. We report Prec@5, Prec@10, Prec@15 and NDCG@5, NDCG@10, NDCG@15, in our experiments. We compare our approaches against two conventional baselines and four improved methods proposed in different literature. All the comparison models are subject to the same filtering and preprocessing procedures. One conventional baseline is User-based CF model, which predicts a user's preferences by considering the preferences of other similar users. To calculate the similarity between the users, we use the Pearson correlation coefficient. The other one is LFM. This traditional model has been described in the Section 3.2. The timeSVD++ and FPMC are two

Table 8. Performance of ILMF and ULMF with Varying Time Bin Length b (Year) on Two Datasets

dataset	b	ILMF			ULMF		
		Prec@5	**Prec@10**	**Prec@15**	**Prec@5**	**Prec@10**	**Prec@15**
Laptop	0.5	0.0139	0.0110	0.0079	0.0118	0.0109	0.0080
Laptop	1	**0.0177**	**0.0139**	**0.0097**	**0.0155**	**0.0123**	**0.0085**
Laptop	1.5	0.0144	0.0120	0.0090	0.0136	0.0112	0.0078
Laptop	2	0.0122	0.0103	0.0080	0.0129	0.0110	0.0072
Smartphone	0.5	**0.0123**	0.0091	0.0059	**0.0135**	0.0085	0.0069
Smartphone	1	0.0115	**0.0099**	**0.0069**	0.0130	**0.0109**	**0.0078**
Smartphone	1.5	0.0011	0.0087	0.0061	0.0111	0.0097	0.0075
Smartphone	2	0.0096	0.0080	0.0060	0.0105	0.0091	0.0068

methods that model the time effect in recommendation. The HFT and EFM incorporate users' opinions by analyzing their reviews. All the baselines are presented as follows:

- User-CF: User-based Collaborative Filtering model is a basic model that utilizes user similarity for recommendation, and it only considers users' ratings on the items. Different from User-CF, we synthetically consider ratings, item attributes, and reviews and integrate these characteristics into a uniform model.
- LFM [5]: Latent Factor Model enhances the neighborhood-based approach (k-nearest neighbors) by using neighborhood interpolation weights. Different from LFM, we get the k-nearest neighbors by using our new measure methods—user-level similarity and item-level similarity.
- timeSVD++ [15]: timeSVD++ is a popular matrix factorization by modeling the time changing behavior throughout the life span of the data.
- FPMC [28]: Factorizing Personalized Markov Chains model is a sequential prediction method based on Markov chain. FPMC integrates matrix factorization with Markov chains together based on personalized transition graphs.
- HFT [21]: Hidden Factors as Topics model combines latent dimensions in rating data with topics in review text. HFT directly transforms topic model as a part of optimization function.
- EFM [40]: Explicit Factor Model leverages phrase-level sentiment analysis of user reviews for personalized recommendation. EFM incorporates both user-feature and item-feature relations, as well as user-item ratings, into a new unified hybrid matrix factorization framework.

To fairly evaluate the performances of all the methods, we use the latest 30% of purchase records of each user as test data, the earliest 70% purchase records as training data. It is worth noting that we do not always divide training set and test set by this ratio when some users have few purchase records. For instance, a user has three purchase records. The first two are training samples and the last is used for testing. In our experiments, many users have few training samples, which causes difficulty in recommendation.

5.2 Analysis of Bin Length, Latent Factor Number, Regularization Parameter

There are several parameters in our model, and we first test the effect of time bin length b while fixing the other parameters. Table 8 illustrates the performances of our methods evaluated by precision with varying bin lengths, which provides us a clue on the parameter selection. On both datasets, we set $b = 1$ year. The reason we choose $b = 1$ year for smartphone dataset is that the

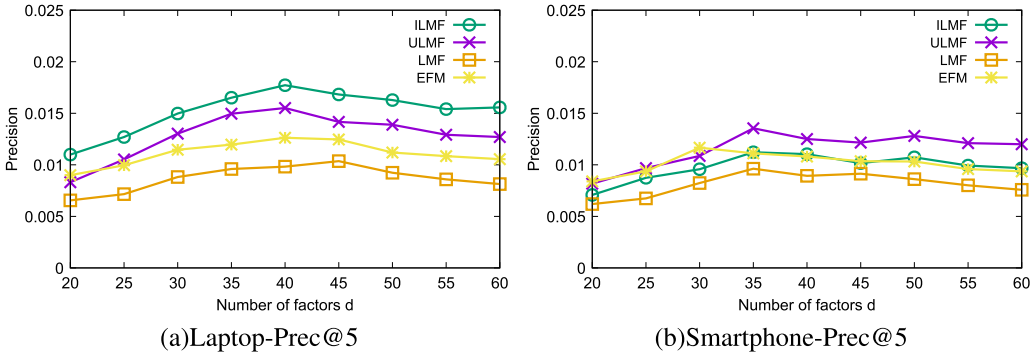


Fig. 3. Precision with varying d.

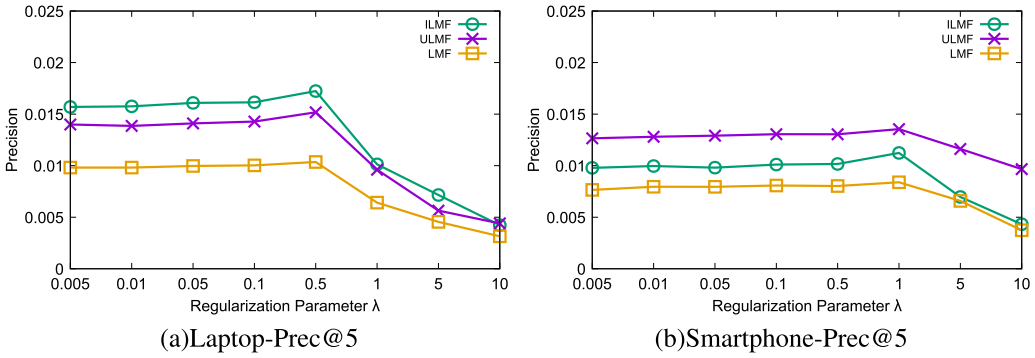


Fig. 4. Precision with varying λ.

model with $b = 1$ year performs better than $b = 0.5$ year in both Prec@10 and Prec@15. In addition, it is just slightly worse than $b = 0.5$ year in Prec@5. Therefore, we make a tradeoff to set $b = 1$ year.

In our model, the suitable length of time bin is decided by the characteristics of the items. More precisely, the update cycle directly decides the length of time bin. For a new item, its item level is stable at the time bin that it launched. Afterwards, the item level starts to depreciate quickly. That is to say, if we choose an improper time bin, the evaluation of the item level may be incorrect, which can lead to a bad recommendation. Therefore, the length of time bin has a strong effect on the recommendation performance. An available approach to quickly find the suitable time bin in a new application is to analyze the update cycle of items, and we will attempt to exploit it in the future.

After identifying the time bin length, we further investigate the impact of latent factor number d. To select the best parameters for ILMF and ULMF, we illustrate their precision performances on two datasets with varying d in Figure 3. As shown in Figure 3, we turn the latent factor number d from 20 to 60 with a step of 5 via setting $\alpha = 0.5$, $\beta = 0.4$, $b = 1$ year, and $\lambda = 0.1$. It shows that the performances of LFM, EFM, ILMF, and ULMF continue to rise with the increase of d until they reach the top when $d = 40$ or 45, and then begin to drop when $d \geq 50$ on Laptop dataset. On Smartphone dataset, the best performances of the models can be obtained at $d = 30$ or 35.

Then, we examine the regularization parameter λ from 0.001 to 10, as depicted in Figure 4, when setting $b = 1$ year, $d = 40$ and 35 on Laptop and Smartphone datasets, respectively. Figure 4(a) shows that the performances of the algorithms increase slowly until $\lambda = 0.5$ when fixing other parameters. On the Smartphone dataset, the performances of models reach the top at $\lambda = 1$ and

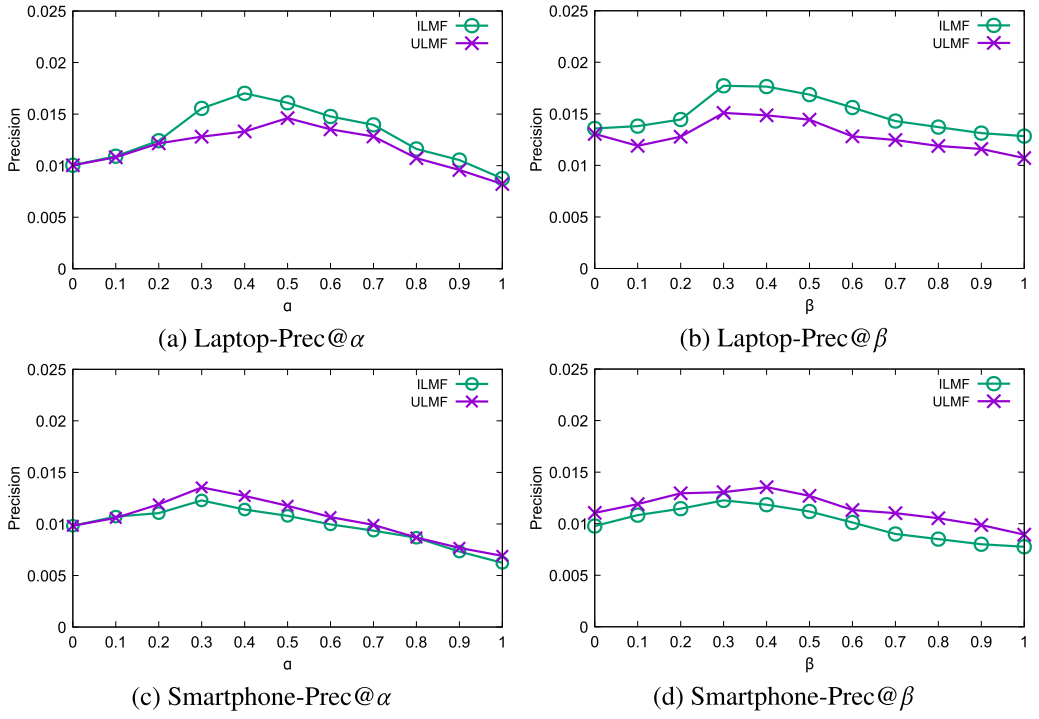


Fig. 5. Precision with varying α and β.

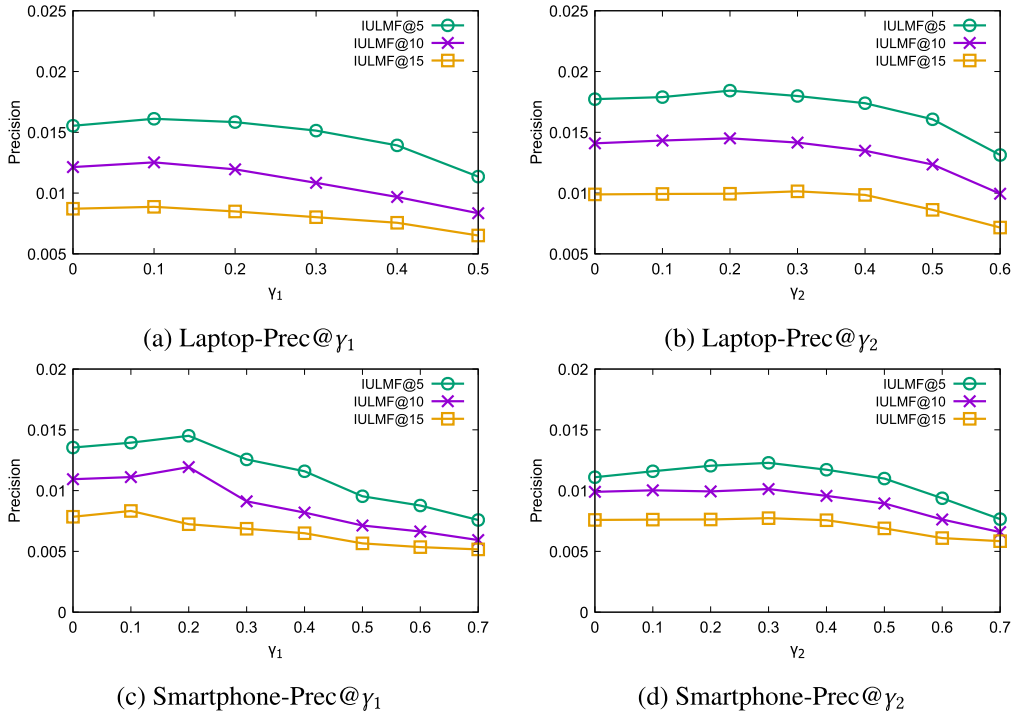
then begin to drop. As our models are based on the latent factor model, they show similar varying trends with LFM when tuning λ.

5.3 Analysis of Balance Parameters

The balance between users' actual rating and alteration caused by other factors is controlled by α, and β is used to weight the loss function of the similarity difference between latent factors and explicit factors. We study the effects caused by different parameters on the methods through a deep analysis of their performances under various parameter settings. Then, we further turn two independent parameters α and β in our models.

On Laptop dataset, we first set $\beta = 0.4$ and tune α from 0 to 1 with a step size of 0.1, and then tune β when fixing the best α. As depicted in Figure 5, ILMF acquires the best performance when $\alpha = 0.4, \beta = 0.3$, and ULMF reaches the peak at $\alpha = 0.5, \beta = 0.4$. We use the same approach to tune the parameters α and β on Smartphone dataset. ILMF acquires the best performance when $\alpha = 0.3, \beta = 0.3$, and ULMF reaches the peak at $\alpha = 0.3, \beta = 0.4$. When we set $\alpha = 0$ and $\beta = 0$, our model is equivalent to LFM. Although ILMF and ULMF cannot always outperform LFM, they can get better results than LFM with series of combinations of α and β. As shown in Figure 5(a), tuning α with fixed β can improve the performance. As shown in Figure 5(b), tuning β with fixed α can also improve the performance. These two results indicate the parameters α and β are both effective in the experiments on the Laptop dataset. Moreover, we can get a similar observation on the Smartphone dataset.

In Figure 6, we examine the performance of IULFM. To reduce the computation complexity, we directly use the values of parameters α and β obtained before. So, we tune γ_2 when setting $\gamma_1 = 0.4$, $\beta = 0.3$, and tune γ_1 when setting $\gamma_2 = 0.5$, $\beta = 0.4$, respectively, on Laptop dataset. The



(a) Laptop-Prec@γ_1 (b) Laptop-Prec@γ_2

(c) Smartphone-Prec@γ_1 (d) Smartphone-Prec@γ_2

Fig. 6. Precision with varying γ_1 and γ_2.

Table 9. Performance of IULMF with Different Similarity Measure Methods

dataset	Cosine	Pearson	Jaccard	User-Item Level
Laptop	0.0141	0.0125	0.0099	0.0184
Smartphone	0.0116	0.0107	0.0082	0.0143

best precision of IULFM is clearly to be $\gamma_1 = 0.4$, $\gamma_2 = 0.2$, and $\beta = 0.3$ on Laptop dataset. Similarly, we tune γ_2 when setting $\gamma_1 = 0.3$, $\beta = 0.2$, and tune γ_1 when setting $\gamma_2 = 0.3$, $\beta = 0.3$, respectively, on Smartphone dataset. The best precision of IULFM on Smartphone dataset is acquired when $\gamma_1 = 0.2$, $\gamma_2 = 0.3$, and $\beta = 0.3$. Compared with item similarity, user-level similarity contributes more to the final performance on Smartphone dataset.

5.4 Analysis of Similarity Measure Methods

To investigate the impact of different similarity measure methods, we illustrate the precision@5 performance of IILMF on both datasets in Table 9. We compared our similarity measure methods with basic Cosine similarity, Pearson Correlation Coefficient, and Jaccard Coefficient. The results show that our defined similarities outperform the others significantly. Compared with basic Cosine similarity, we consider more factors that reflect the user's personality more comprehensively. Jaccard Coefficient can only capture the similarity of items purchased by the user, so it gets the worst performance.

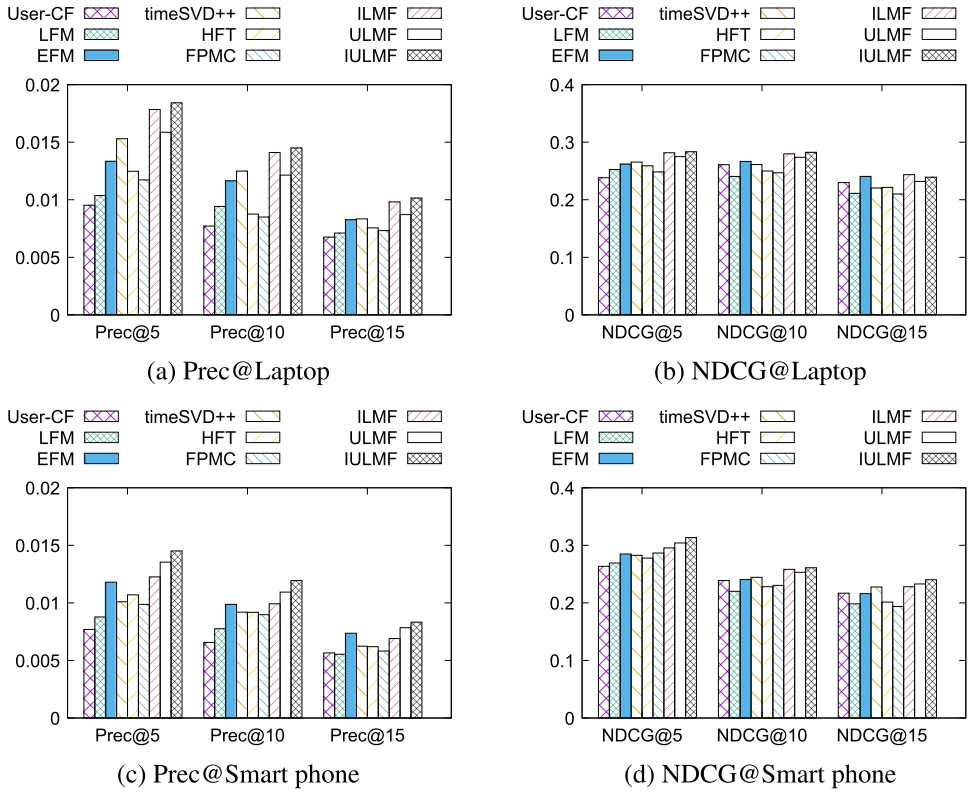


Fig. 7. Experiment comparison.

5.5 Comprehensive Analysis

The Prec and NDCG of all the methods are reported in Figure 7. As shown in Figure 7(a), the experimental results demonstrate the effectiveness of the strategy that recommends the items that have the similar aspect level with the items bought by the user before. Specifically, in terms of Prec@5 and Prec@15, we observe that the ILMF outperforms other methods obviously on Laptop dataset. Compared to the best baseline timeSVD++, the improvements of IULMF are 20.41% on Prec@5, 16.07% on Prec@10, and 21.66% on Prec@15, respectively. The corresponding improvements on NDCG@5, NDCG@10, and NDCG@15 are 6.76%, 8.07%, and 8.51%, respectively. Another baseline FPMC that incorporates time factor fails to achieve the desired effect, which has the similar performance with LFM. A probable reason is that there is not enough serialized purchase data for FPMC, because the laptop is not a high-frequency consumption item. Meanwhile, ILMF outperforms ULMF.

On the Smartphone dataset, EFM and HFT outperform timeSVD++ and FPMC, which reflects that the reviews provide more support than temporal information in Smartphone recommendation. The proposed ULMF outperforms other methods and consistently exceeds the baselines EFM, HFT, timeSVD++, and FPMC, in terms of Prec@5, by 14.79%, 26.60%, 34.07, and 37.22%, respectively. The improvements that IULMF outperforms the baseline EFM on Prec@5, Prec@10, and Prec@15 are 21.31%, 16.79%, and 13.04%, respectively, and the improvements on NDCG@5, NDCG@10, and NDCG@15 are 10.02%, 8.48%, and 11.11%, respectively.

In our analysis, the stability of users' preference levels have effects on the performance of ULMF and ILMF. ILMF may perform better than ULMF in the domain where users have more stable preference levels, since the performance of ULMF is affected by the calculation of users' preference levels. In other words, if a user's preference level is not stable in a domain, the user level computed by our model cannot well reflect his real purchase intention. In Smartphone dataset, ULMF outperforms ILMF. One possible reason is that the consumer prefers to follow the others' choices when buying a smartphone. Another possible reason is that the consumer is more likely to buy the smartphones of different levels compared with the laptops. In real life, some people even have two phones at the same time, where the high-level one is for daily use and the low-level one is for standby application.

In a nutshell, our methods are designed for recommending the items with a relatively long time span purchase cycle. This long time span property can be broadly found when buying the expensive and professional items. A potential reason is that the consumers are usually more cautious when buying expensive items, which better reflects their stable preference levels. However, a consumer may show more randomness when buying cheap daily supplies, which is hard to be captured by our model. For the professional items, the consumers may have known some relevant domain knowledge in advance. Therefore, there is a low chance for them to make impulse decisions. In addition, most of the shopping websites now provide detailed aspect information for the expensive and professional items, which is useful for our methods. In conclusion, the stability of consumers' purchasing intention has an essential influence on the performance of our methods.

6 CONCLUSION AND FUTURE WORK

In this article, we propose to leverage item level and user level for personalized recommendation. To model user level and item level, we extract explicit item aspect information, user sentiment, and user rating, then incorporate them into a new unified framework. Besides introducing more measures and capturing more connections between features, we exploit the relationship between explicit and implicit latent factors. Our experimental results show that IULMF outperforms the baselines on different datasets, which proves that it is reasonable and credible to take item-level similarity and user-level similarity into account for building effective an recommendation system.

Before implementing our method in a new application, we need to analyze the characteristics of the items to determine the feasibility of applying our model. Generally, three conditions are required: (1) aspect information can be extracted; (2) aspects can be easily compared; (3) time information is available. Our article is a preliminary attempt to integrate item level and user level into a matrix factorization model for item recommendations. Several directions for future research are promising: First, we will extend our model to recommend the items that are hard to compare by the explicit attributes, such as the books, music, and daily necessities. Second, we will exploit the recommendations for the cross-domain items to overcome the limitation of model migration. Third, we will explore how to compare the aspects of items without the expert knowledge.

APPENDIX

A PROCESS OF DERIVATION

We give the main procedure of derivation as follows:

$$\frac{\partial \sum_{pc \in P_{\mathcal{T}_{qe}}} (M_{cj} - S_{cj})}{\delta Q_j} = \frac{\partial \sum_{pc \in P_{\mathcal{T}_{qe}}} M_{cj}}{\partial Q_j} = \frac{\partial \sum_{pc \in P_{\mathcal{T}_{qe}}} \frac{Q_c Q_j^T}{|Q_c||Q_j^T|}}{\partial Q_j}$$
$$= \frac{\sum_{pc \in P_{\mathcal{T}_{qe}}} \frac{Q_c}{|Q_c|} \partial \frac{Q_j}{|Q_j|}}{\partial Q_j} = \frac{\sum_{pc \in P_{\mathcal{T}_{qe}}} \frac{Q_c}{|Q_c|} \partial \frac{Q_j}{(Q_j Q_j^T)^{1/2}}}{\partial Q_j}$$

$$= \sum_{p_c \in P_{\mathcal{T}_{qe}}} \frac{Q_c}{|Q_c|} \frac{(Q_j Q_j^T)^{1/2} \partial Q_j / \partial Q_j - Q_j^T \partial (Q_j Q_j^T)^{1/2} / \partial Q_j}{((Q_j Q_j^T)^{1/2})^2}$$
$$= \sum_{p_c \in P_{\mathcal{T}_{qe}}} \frac{Q_c}{|Q_c|} \frac{(Q_j Q_j^T)^{1/2} I_d - (Q_j Q_j^T)^{-1/2} Q_j^T Q_j}{(Q_j Q_j^T)}.$$

B ASPECT EXTRACTION RESULT

Table 10. Result of Aspect Extraction

Dataset	Aspect	Aspect word
Laptop	CPU	processor, cpu, computing, clockspeed compute, performance
	Screen	screen, display, inch, size
	RAM	memory, ram
	Operation System	system, os, windows, vista, win7, window,
	Graphic	graphic, game, gaming, fan, photo
	Harddisk	harddisk, disk, storage
	WiFi	wi-fi, wifi, wireless
Smartphone	Screen	screen, display, inch, size, amoled, oled, led
	CPU	cpu,processor,speed
	RAM	ram
	ROM	rom, capacity, flash, storage, space, memory

REFERENCES

[1] F. Abbattista, M. Degemmis, N. Fanizzi, O. Licchelli, et al. 2007. Learning customer profiles for content-based filtering in e-commerce. *Commun. ACM* 50, 1 (Jan. 2007), 36–44. DOI: http://doi.acm.org/10.1145/1219092.1219093

[2] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* 17, 6 (2005), 734–749. DOI: https://doi.org/10.1109/TKDE.2005.99

[3] Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. 2017. Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In *KDD*. ACM, 717–725. DOI: https://doi.org/10.1145/3097983.3098170

[4] Robert M. Bell and Yehuda Koren. 2007. Improved neighborhood-based collaborative filtering. In *KDD Cup and Workshop at the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 7–14.

[5] Robert M. Bell and Yehuda Koren. 2007. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM*. IEEE Computer Society, 43–52. DOI: https://doi.org/10.1109/ICDM.2007.90

[6] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *J. Mach. Learn. Res.* 3 (2003), 993–1022. Retrieved from http://www.jmlr.org/papers/v3/blei03a.html.

[7] Juan Cao, Xia Tian, Jintao Li, Yongdong Zhang, and Tang Sheng. 2009. A density-based method for adaptive LDA model selection. *Neurocomputing* 72, 7 (2009), 1775–1781.

[8] Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J. Smola, Jing Jiang, and Chong Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In *KDD*. ACM, 193–202. DOI: https://doi.org/10.1145/2623330.2623758

[9] Yi Ding and Xue Li. 2005. Time weight collaborative filtering. In *CIKM*. ACM, 485–492. DOI: https://doi.org/10.1145/1099554.1099689

[10] Horatiu Dumitru, Marek Gibiec, Negar Hariri, Jane Cleland-Huang, Bamshad Mobasher, Carlos Castro-Herrera, and Mehdi Mirakhorli. 2011. On-demand feature recommendations derived from mining public product descriptions. In *ICSE*. ACM, 181–190. DOI: https://doi.org/10.1145/1985793.1985819

[11] Zhen Hai, Gao Cong, Kuiyu Chang, Wenting Liu, and Peng Cheng. 2014. Coarse-to-fine review selection via supervised joint aspect and sentiment model. In *SIGIR*. ACM, 617–626. DOI: https://doi.org/10.1145/2600428.2609570

[12] Chen Jian, Yin Jian, and Huang Jin. 2005. Automatic content-based recommendation in e-Commerce. In *EEE*. IEEE Computer Society, 748–753. DOI: https://doi.org/10.1109/EEE.2005.37

[13] Yohan Jo and Alice H. Oh. 2011. Aspect and sentiment unification model for online review analysis. In *WSDM*. ACM, 815–824. DOI: https://doi.org/10.1145/1935826.1935932
[14] Thorsten Joachims, Dayne Freitag, and Tom M. Mitchell. 1997. Web watcher: A tour guide for the world wide web. In *IJCAI (1)*. Morgan Kaufmann, 770–777.
[15] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. *Commun.* In *KDD*. ACM, 447–456. DOI: https://doi.org/10.1145/1721654.1721677
[16] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *IEEE Comput.* 42, 8 (2009), 30–37. DOI: https://doi.org/10.1109/MC.2009.263
[17] Daniel D. Lee and H. Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. In *NIPS*. MIT Press, 556–562. Retrieved from http://papers.nips.cc/paper/1861-algorithms-for-non-negative-matrix-factorization.
[18] Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers. DOI: https://doi.org/10.2200/S00416ED1V01Y201204HLT016
[19] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *AAAI*. AAAI Press, 194–200. Retrieved from http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11900.
[20] Yue Lu, Malú Castellanos, Umeshwar Dayal, and Cheng Xiang Zhai. 2011. Automatic construction of a context-aware sentiment lexicon: An optimization approach. In *WWW*. ACM, 347–356. DOI: https://doi.org/10.1145/1963405.1963456
[21] Julian J. McAuley and Jure Leskovec. 2013 Hidden factors and hidden topics: Understanding rating dimensions with review text. In *RecSys*. ACM, 165–172. DOI: https://doi.org/10.1145/2507157.2507163
[22] Anusree Mitra. 1995. Advertising and the stability of consideration sets over multiple purchase occasions. *Int. J. Res. Market.* 12, 1 (1995), 81–94.
[23] Samaneh Moghaddam, Mohsen Jamali, and Martin Ester. 2012. ETF: Extended tensor factorization model for personalizing prediction of review helpfulness. In *WSDM*. ACM, 163–172. DOI: https://doi.org/10.1145/2124295.2124316
[24] Raymond J. Mooney and Loriene Roy. 2000. *Content-based book recommending using learning for text categorization*. In *ACM DL*. ACM, 195–204. DOI: https://doi.org/10.1145/336597.336662
[25] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan M. Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *ICDM*. IEEE Computer Society, 502–511. DOI: https://doi.org/10.1109/ICDM.2008.16
[26] Weike Pan, Qiang Yang, Wanling Cai, Yaofeng Chen, Qing Zhang, Xiaogang Peng, and Zhong Ming. 2019. Transfer to rank for heterogeneous one-class collaborative filtering. *ACM Trans. Inf. Syst.* 37, 1 (2019), 10:1–10:20.
[27] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Comput. Ling.* 37, 1 (2011), 9–27. DOI: https://doi.org/10.1162/coli_a_00034
[28] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW*. ACM, 811–820. DOI: https://doi.org/10.1145/1772690.1772773
[29] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An open architecture for collaborative filtering of netnews. In *CSCW*. ACM, 175–186. DOI: https://doi.org/10.1145/192844.192905
[30] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML (ACM International Conference Proceeding Series)*, Vol. 307. ACM, 880–887. DOI: https://doi.org/10.1145/1390156.1390267
[31] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*. ACM, 285–295. DOI: https://doi.org/10.1145/371920.372071
[32] Sylvain Senecal and Jacques Nantel. 2004. The influence of online product recommendations on consumers' online choices. *J. Retail.* 80, 2 (2004), 159–169.
[33] Allan D. Shocker, Moshe Ben-Akiva, Bruno Boccara, and Prakash Nedungadi. 1991. Consideration set influences on consumer decision-making and choice: Issues, models, and suggestions. *Market. Lett.* 2, 3 (1991), 181–197.
[34] Xiaoyuan Su and Taghi M. Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Adv. Artific. Intell.* 2009 (2009), 421425:1–421425:19. DOI: https://doi.org/10.1155/2009/421425
[35] Tiffany Ya Tang, Pinata Winoto, and Keith C. C. Chan.2003. On the temporal analysis for improved hybrid recommendations. In *Web Intelligence*. IEEE Computer Society, 214–220.
[36] Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent aspect rating analysis on review text data: A rating regression approach. In *KDD*. ACM, 783–792. DOI: https://doi.org/10.1145/1835804.1835903
[37] Liang Xiang, Quan Yuan, Shiwan Zhao, Li Chen, Xiatian Zhang, Qing Yang, and Jimeng Sun. 2010. Temporal recommendation on graphs via long- and short-term preference fusion. In *KDD*. ACM, 723–732. DOI: https://doi.org/10.1145/1835804.1835896

[38] Taebok Yoon, Seunghoon Lee, Kwang ho Yoon, Dongmoon Kim, and Jee-Hyong Lee. 2008. A personalized music recommendation system with a time-weighted clustering. *2008 4th International IEEE Conference Intelligent Systems* 2 (2008), 10–48.

[39] Sheng Zhang, Weihong Wang, James Ford, Fillia Makedon, and Justin D. Pearlman. 2005. Using singular value decomposition approximation for collaborative filtering. In *CEC*. IEEE Computer Society, 257–264. DOI: https://doi.org/10.1109/ICECT.2005.102

[40] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *SIGIR*. ACM, 83–92. DOI: https://doi.org/10.1145/2600428.2609579

[41] Yongfeng Zhang, Min Zhang, Yi Zhang, Guokun Lai, Yiqun Liu, Honghui Zhang, and Shaoping Ma. 2015. Daily-aware personalized recommendation based on feature-level time series analysis. In *WWW*. ACM, 1373–1383. DOI: https://doi.org/10.1145/2736277.2741087

[42] Kaiqi Zhao, Gao Cong, Quan Yuan, and Kenny Q. Zhu. 2015. SAR: A sentiment-aspect-region model for user preference analysis in geo-tagged reviews. In *ICDE*. IEEE Computer Society, 675–686. DOI: https://doi.org/10.1109/ICDE.2015.7113324

Received March 2018; revised August 2019; accepted December 2019