Collaborative Recommendation with User Generated Content

Yueshen Xu*, Jianwei Yin**

*College of Computer Science and Technology, Zhejiang University, Hangzhou, Zhejiang, 310027, China

Abstract

In the age of Web 2.0, user generated content (UGC), such as user review and social tag, ubiquitously exists on the Internet. Although there exist different kinds of UGC in recommender systems, the existing works only studied a single kind of UGC in each of their papers. Thus, the previous works lose a chance to uncover the similar effects of different kinds of UGC in recommender systems. In this paper, we propose a unified way to utilize various types of UGC to enhance the recommendation accuracy. We build two novel statistical models, which are based on collaborative filtering and topic modeling. Incorporating UGC text, one model focuses on learning user preferences, and the other model aims to learn user preferences and item aspects jointly. With an effective parameter estimation algorithm, our models can not only acquire prediction values of missing ratings, but also produce interpretable topics. We conducted comprehensive experiments on three real-world datasets. The experimental results demonstrate that our proposed models can achieve large improvements compared to several well-known baseline models.

Keywords: Recommender System, User Generated Content, Collaborative Filtering, Topic Modeling, Parameter Estimation

1. Introduction

The recommender system has been an indispensable component in a modern e-commerce system, which aims to gauge a user’s preference. Collaborative filtering (denoted by CF)-based and content-based algorithms are two prevalent approaches to the recommendation problem. Among all the CF-based models, models based on Matrix Factorization (denoted by MF) are widely used in industry and academia (Koren et al., 2009). CF-based algorithms are based on the following assumption: users that had similar preferences before tend to give similar ratings to the same item in the future (Su and Khoshgoftaar, 2009). The key task of CF-based algorithms is to find similar users and infer a user’s preference from the past rating records. The content-based recommender system also plays an important role in many practical scenarios (Pandora,
In such approaches, an item’s and a user’s profiles are collected to predict whether the user likes the item or not (Lops et al., 2011). Meanwhile, user generated content widely exists on the Internet, and has been utilized in recommender systems, such as tag-based and review-based recommendation algorithms (Liang et al., 2010; McAuley and Leskovec, 2013). Formally, user generated content (denoted by UGC) refers to the content created by users on the Internet (Moens et al., 2014), such as social tag, review, blog and tweet.

The utilization of side information to improve the recommendation accuracy is a popular trend in recent years, such as social relationship, implicit feedback and context (Ma et al., 2011; Yang et al., 2012; Liu and Aberer, 2013; Ostuni et al., 2013; Tang et al., 2013; Chen et al., 2014a). They assume that a user’s preference can be inferred through his or her friends’ preferences, or can be influenced by certain context. Although such side information can indeed help improve the recommendation accuracy, many e-commerce web sites do not have the mentioned side information. For example, in Amazon¹, Ebay² and Expedia³, there are no social relationships. However, they do have user reviews. For building a model with high applicability, we have to infer a user’s preference from the available information sources, such as rating records and UGC text.

In recent years, several works tried to utilize an item’s description to learn the item’s associated latent aspects, and make better recommendation (Agarwal and Chen, 2010; Wang and Blei, 2011; Zhang and Wang, 2014). More related works will be discussed in Section 2. The existing models are verified to achieve higher prediction accuracy than the traditional CF-based algorithms that only rely on the past rating records. However, there are still three problems in the existing models.

1. An item’s description lacks distinctiveness since many words in the descriptions of different items tend to overlap. For example, the product descriptions of two cellphones like Samsung Galaxy S5 and iPhone are overlapping largely with each other. The contained words are usually about product features, such as battery, price and size.

2. An item’s description is static, and the description text can only be edited by website editors. Thus, the word space, which is constructed by the item descriptions, is usually very sparse. The high sparsity lowers the accuracy of the latent aspects.

3. An item’s description is independent of a user’s preference, so it is difficult to infer the user’s preference from the description text.

In contrast, UGC text can emphasize an item’s outstanding features. For example, the song You belong with me receives many following tags in Last.fm⁴, including love, country, pop and Taylor Swift, which precisely highlight this song’s aspects. UGC can

¹http://www.amazon.com/
²http://www.ebay.com/
³http://www.expedia.com/
⁴http://www.last.fm
reflect a user’s interested topics as well. For example, if adventure, space and science are frequently used by Bob in his reviews, we can infer that Bob may like the movies Star Trek and Interstellar since the two movies have the related themes. Also, the UGC text opens a gate for us to learn the reason why a user likes or dislikes an item. For example, after purchasing a computer, if Baron writes a review “I like its color and its light weight. It’s easy to carry”, it gives us a hint that we can recommend Baron other electronic devices with the same color and light weight. Besides, the UGC text increases along with time, so the amount of text information is abundant. Also, some kinds of UGC, such as social tag, provide a direct way to acquire meaningful phrases without chunking (for more details about chunking in NLP, please refer to (Manning and Schütze, 1999)). For example, in Last.fm, the song Freezing Moon receives black metal (a tag) for many times, which precisely depicts this song’s genre. Clearly, black metal should be regarded as an integrated phrase. If not, this tag will be mistakenly regarded as a simple combination of a color term (black) and a material term (metal).

Although in some previous works (Liang et al., 2010; Zanardi and Capra, 2011; McAuley and Leskovec, 2013; Ling et al., 2014), UGC has been verified to be helpful to improve the prediction accuracy in recommender systems, there are still three challenging problems:

1. Not every web site allows a user to give tags to products, and some web sites even do not give the chance to write reviews. For example, in Bestbuy and Newegg, users are not allowed to tag. In Movielens, users are not allowed to write reviews.

2. The existing works only study a single kind of UGC in each paper, and leverage different kinds of UGC in very different ways. This makes it difficult to uncover the similar effects which are shared among diverse kinds of UGC in recommender systems.

3. There exist many semantic problems in UGC text, like synonym and acronym. However, some traditional methods ignore these problems without considering the semantic correlation among words.

To deal with the first and second problems, it is necessary to propose an approach that is capable of leveraging different types of UGC in the same way. To deal with the third problem, we need to employ effective techniques, such as topic modeling, to relieve the semantic problems in UGC text.

To tackle the issues mentioned above, in this paper, we study different types of UGC, and incorporate them into the recommender system in a unified manner. Because reviews and tags are two kinds of widely existing UGC, in this paper, we focus on user reviews and social tags. We propose our first model to learn a user's interested topics from his or her UGC text. Most of the previous review-based models only focus on learning item aspects. We want to investigate the function of UGC in learning a user’s
preference. Then, we propose the second model which learns a user’s preference and an item’s aspects jointly. Different from the item description-based algorithms, our proposed model learns the aspects from the collection of the UGC text that an item receives. Both of the proposed models are statistical models and have well-designed generative processes. We design an effective iterative algorithm to learn the model parameters. We conduct comprehensive experiments on three real-world datasets to attest the effectiveness of our models. We also investigate our models’ performance in the “cold-start” scenario, which means a user has limited or even very few ratings. The “cold-start” problem is a very challenging issue since the traditional CF-based models cannot make satisfactory recommendation with insufficient data.

The main contributions of this paper are summarized as follows:

1. It studies the effect of UGC in learning a user’s preference and an item’s aspects. It finds that reviews and social tags are two valuable information sources to learn user preferences and item aspects.

2. It proposes two novel statistical models, both of which can take UGC text and rating records together as input. The first model focuses on learning users’ interested topics. The second model extends the first model, and takes users’ and items’ topics/aspects learning task into consideration jointly.

3. It proposes a parameter estimation algorithm for the proposed models, which is verified to be effective by the experiments.

4. It conducts sufficient experiments on three public real-world datasets, which demonstrate that our proposed models outperform all baseline models. It also expands one of the datasets by crawling more contents. We will release the expanded dataset publicly to facilitate related research in the community.

The rest of this paper is organized as follows: Section 2 summarizes the related work. Section 3 gives a concise explanation of two base models. Section 4 explicates the details of our proposed models. Section 5 shows the experimental results along with a thorough analysis. Lastly, Section 6 concludes the paper and discusses the future work.

2. Related Work

In recent years, in recommender systems, it has been a mainstream to introduce side information to enhance the performance of traditional collaborative filtering models. The typical side information includes social relationship, social tag, item description, etc. Our work is closely related to tag-based recommendation, review-based recommendation and item description-based recommendation (Agarwal and Chen, 2010; Liang et al., 2010; Gemmell et al., 2011; Ling et al., 2014).

**Tag-based Recommendation.** There have been some works that employ social tags to build the connection between users and items (Liang et al., 2010; Gemmell et al., 2011; Zanardi and Capra, 2011). Liang et al. (2010) proposed a tag-based recommendation algorithm. Each item was attached with a tag vector as this item’s profile,
which was used to calculate the similarity between each pair of items. Each user also had a profile, which was constructed from his or her tagging record and used to calculate the similarity between each pair of users. A score for each item in the candidate list was computed for top-N recommendation. Since not every web site has tags, this model’s applicability is limited. The models proposed in (Liang et al., 2010; Gemmell et al., 2011; Zanardi and Capra, 2011) are all traditional memory-based algorithms, so it is hard to interpret how the experimental results are produced. In contrast, in our proposed models, through the generative process (see Section 4.1 and 4.2), we can clearly understand how a rating is generated.

We also notice that there is a line of research which aims to solve the “tag recommendation” problem (Rendle and Schmidt-Thieme, 2010; Feng and Wang, 2012; Wang et al., 2013). However, in this paper, we focus on how to use social tags to better solve the rating prediction problem. Thus, our work is quite different from tag recommendation.

**Review-based Recommendation.** In recent years, there are several works that incorporated the review text into recommendation algorithms (McAuley and Leskovec, 2013; Bao et al., 2014; Diao et al., 2014; Ling et al., 2014). In (McAuley and Leskovec, 2013), the authors proposed a model named Hidden Factors as Topics (HFT), which combined the objectives of matrix factorization and topic modeling together. HFT connected the topic proportion, which was learned from reviews, with the latent features via an exponential transformation function. Although HFT can either generate the topic distribution of users or items, HFT cannot model the topic distributions on users and items jointly. In contrast, our proposed model J-UCR (see Section 4.2) is designed to jointly generate the user-specific and item-specific topic distributions. In (Bao et al., 2014; Diao et al., 2014; Ling et al., 2014), all the proposed algorithms tried to model reviews and ratings in one generative process. Note that, all of the works only investigate the function of reviews in a recommender system, but ignore other kinds of UGC. Besides HFT, all other works only focus on learning item aspects, but ignore to learn user preferences from reviews.

**Item Description-based Recommendation.** Recently, there are a few works that built statistical models to utilize item descriptions in recommender systems (Agarwal and Chen, 2010; Wang and Blei, 2011; Purushotham et al., 2012; Chen et al., 2014a; Wang and Li, 2014; Zhang and Wang, 2014). Wang and Blei (2011) proposed a hybrid prediction model named Collaborative Topic Regression (denoted by CTR), which combined Latent Dirichlet Allocation (LDA) and Matrix Factorization (MF) together. As a Bayesian generative model, CTR integrated an item’s description into the learning process of the item’s latent features. Later, Purushotham et al. (2012) and Chen et al. (2014a) extended CTR with the additional social and context information. Although topic modeling can solve many problems existing in UGC text as stated in Section 1, an item’s description is static, and not related to a user’s consumption behavior and preference. In this paper, we employ UGC text, such as social tags and reviews, since they can depict an item’s aspects, reflect a user’s preference, and increase dynamically. Besides, we aim to solve the recommendation problem in the case of no social relationship like in Amazon and Ebay.
3. Base Models

In this section, we will briefly review two basic models, i.e., Probabilistic Matrix Factorization (PMF) and Collaborative Topic Regression (CTR), which are used as base models in this paper.

3.1. Probabilistic Matrix Factorization

Probabilistic Matrix Factorization (PMF) is a Bayesian generative model, and gives a probabilistic interpretation of Matrix Factorization (MF) model (Salakhutdinov and Mnih, 2008). Its generative process is as follows:

1. For each user \( i \), sample user latent feature vector \( \vec{u}_i \sim \mathcal{N}(0, \lambda_u^{-1} I_K) \)
2. For each item \( j \), sample item latent feature vector \( \vec{v}_j \sim \mathcal{N}(0, \lambda_v^{-1} I_K) \)
3. For each rating \( R_{ij} \), sample \( R_{ij} \sim \mathcal{N}(\vec{u}_i^T \vec{v}_j, c^{-1}_{ij}) \)

where \( \vec{u} \in \mathbb{R}^{K \times M} \), \( \vec{v} \in \mathbb{R}^{K \times N} \) are the user and item latent feature matrices respectively. \( \vec{u}_i \) denotes the \( i \)th column vector of \( \vec{u} \) and \( \vec{v}_j \) denotes the \( j \)th column vector of \( \vec{v} \). \( M, N, K \) are the numbers of users, items and latent features respectively. \( \mathcal{N}(\cdot, \cdot) \) is the probability density function of the Gaussian distribution. \( R = \{ R_{ij} \} (1 \leq i \leq M, 1 \leq j \leq N) \) denotes the rating matrix (a toy example is shown in Table 1), which is usually extremely sparse. The entries filled with grey in Table 1 represent those missing ratings that need to be predicted. \( I_K \) is the identity matrix whose size is \( K \times K \), and \( \lambda_u, \lambda_v \) are both small constants. \( c_{ij} \) is a precision parameter to determine the weight of ratings in different cases as

\[
c_{ij} = \begin{cases} a, & \text{if } R_{ij} \text{ exists} \\ b, & \text{if } R_{ij} \text{ does not exist} \end{cases}
\]

where both \( a \) and \( b \) are constants. We define a matrix \( c = \{ c_{ij} \} \) for the parameter estimation in Section 4.2. We will discuss the value setting of \( a \) and \( b \) in the experiment section (see Section 5.2).

Table 1: A toy example of the user-item rating matrix

<table>
<thead>
<tr>
<th></th>
<th>Titanic</th>
<th>Gravity</th>
<th>...</th>
<th>Interstellar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>5</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Bob</td>
<td></td>
<td>4</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>Baron</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After estimating \( \vec{u}_i \) and \( \vec{v}_j \), the missing \( R_{ij} \) is approximated as the inner product of \( \vec{u}_i \) and \( \vec{v}_j \), which is presented below.

\[
R_{ij} \approx \vec{u}_i^T \vec{v}_j
\]

It can be seen that PMF is only capable of using the information of rating records. For more details, please refer to (Salakhutdinov and Mnih, 2008).
3.2. Collaborative Topic Regression

Collaborative Topic Regression (CTR) was proposed in (Wang and Blei, 2011), and is a hybrid model, which fuses Probabilistic Matrix Factorization (PMF) and Latent Dirichlet Allocation (LDA) together. For more details of LDA, please refer to (Blei et al., 2003). For rating prediction, CTR not only can use rating records, but also can leverage the item descriptions to learn the hidden aspects of items. In CTR, an item is regarded as a document, which contains the words that exist in its description. As a Bayesian generative model, its generative process is depicted as follows:

1. For each user $i$ ($1 \leq i \leq M$), sample user latent feature vector $\vec{u}_i \sim N(0, \lambda_u^{-1}I_K)$
2. For each item $j$ ($1 \leq j \leq N$),
   (a) sample topic proportion vector $\vec{\theta}_j \sim Dir(\alpha)$
   (b) sample item latent offset $\vec{\epsilon}_j \sim N(0, \lambda_v^{-1}I_K)$ and set item latent feature vector $\vec{v}_j = \vec{\epsilon}_j + \vec{\theta}_j$
3. For word $w$ that is in item $j$’s description,
   (a) sample topic assignment $z_w \sim Mult(\vec{\theta}_j)$
   (b) sample word $w \sim Mult(\vec{\beta}_z)$
4. For each rating $R_{ij}$, sample $R_{ij} \sim N(\vec{u}_i^T \vec{v}_j, \sigma^{-1})$

where $Dir(\alpha)$ represents the Dirichlet distribution with the hyperparameter $\alpha$, and $Mult(\cdot)$ represents the multinomial distribution. $\vec{\beta}_k = (\beta_{k1}, \ldots, \beta_{kL})$ ($1 \leq l \leq L$) denotes the topic-word distribution, where $L$ is the vocabulary size and $K$ is the number of topics. $\vec{\theta}_j = (\theta_{j1}, \ldots, \theta_{jk}, \ldots, \theta_{jK})^T$ ($1 \leq j \leq N$) denotes the item-topic distribution, where $N$ is the number of items. As illustrated in the generative process, CTR ignores the abundant UGC text that can be used to learn a user’s interested topics. For more details, please refer to (Wang and Blei, 2011).

4. Proposed Approaches

In this section, we will elaborate our two proposed models, and explicate the parameter estimation algorithm.

4.1. UGC-based Collaborative Recommendation

As discussed in Section 1, a user’s preference can be inferred from his or her UGC text, and is usually specific to certain topics. For another instance, Alice wrote a lot of reviews in IMDB\footnote{http://www.imdb.com/} containing many following words, such as love, action and supernatural. It can be inferred that she may like movies Twilight and Pirates of the Caribbean, since those movies have related themes of love and supernatural. To leverage UGC text to infer user preferences, in a recommender system, we collect all unique words or phrases that exist in the UGC text, which form the term set $T$ as follows:

$$T = \{t_l \mid t_l \text{ is a word or phrase in UGC} \} \ (1 \leq l \leq L)$$
where $L$ is the number of unique terms (words or phrases). Besides a single word, $t_l$ can also be a phrase, especially in the case of social tag. Each user $i$ will correspond to a term count vector $\vec{e}_i = (n_{t_1}, \ldots, n_{t_l}, \ldots, n_{t_L})$, in which $n_{t_l} (1 \leq l \leq L)$ is the times of term $t_l$ that user $i$ wrote before. Specially, if user $i$ has never written $t_l$ in his or her UGC text, $n_{t_l}$ in $\vec{e}_i$ will be 0. Hence, we can construct a user-term co-occurrence matrix, in which each row is $\vec{e}_i$. A toy example is shown in Table 2.

<table>
<thead>
<tr>
<th></th>
<th>space</th>
<th>science</th>
<th>...</th>
<th>love</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>3</td>
<td>0</td>
<td>...</td>
<td>35</td>
</tr>
<tr>
<td>Bob</td>
<td>25</td>
<td>10</td>
<td>...</td>
<td>5</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Baron</td>
<td>35</td>
<td>15</td>
<td>...</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 2: A toy example of the user-term co-occurrence matrix

Treating each user $i$ as a document, through topic modeling technique, we can estimate user $i$’s topic proportion $\vec{\theta}_i$ (see details of $\vec{\theta}_i$ in the following generative process). Since $\vec{\theta}_i$ depicts the topic distribution over a document, as for a user, $\vec{\theta}_i$ depicts the distribution of his or her interested topics. In our model, we propose to approximate $R_{ij}$ via three parts, i.e., latent user feature vector $\vec{u}_i$, topic proportion vector $\vec{\theta}_i$, and latent item feature vector $\vec{v}_j$. $R_{ij}$ is estimated as

$$R_{ij} \approx (\vec{u}_i + \vec{\theta}_i)^T \vec{v}_j$$  \hspace{1cm} (3)

We give the generative process of our model as follows.

1. For each user $i$ ($1 \leq i \leq M$),
   (a) sample topic proportion vector $\vec{\theta}_i \sim Dir(\alpha)$
   (b) sample user latent feature vector $\vec{u}_i \sim N(0, \lambda_u^{-1}I_K)$

2. For each item $j$ ($1 \leq j \leq N$), sample item latent feature vector $\vec{v}_j \sim N(0, \lambda_v^{-1}I_K)$

3. For term $t_{il}$ that user $i$ wrote in his or her UGC text,
   (a) sample topic assignment $z_{il} \sim Mult(\vec{\theta}_i)$
   (b) sample term $t_{il} \sim Mult(\vec{\beta}_{z_{il}})$

4. For each rating $R_{ij}$, sample $R_{ij} \sim N((\vec{u}_i + \vec{\theta}_i)^T \vec{v}_j, c_{ij}^{-1})$

In the above process, $\vec{\theta}_i = (\theta_{i1}, \ldots, \theta_{ik}, \ldots, \theta_{iK})^T$, and we define $\theta = \{\vec{\theta}_i\} (1 \leq i \leq M)$, where $M$ is the number of users. $\vec{\theta}_i$ denotes the user-topic distribution. We title this model UGC-based Collaborative Recommendation Model (denoted by UCR) and give its probabilistic graphical model in Figure 1. We will discuss the parameter estimation algorithm of UCR in the following Section 4.2.
4.2. Joint UGC-based Collaborative Recommendation

As illustrated in Section 1, UGC can also help learn an item’s aspects. For another example, in Amazon\(^9\), the reviews of the novel *A Study in Scarlet* contain many following terms, including *fiction*, *detective* and *deduction*. These terms give an important clue to make a recommendation to a user who is a fan of detective story. Given the term set \( T \) (see Section 4.1), which contains all terms extracted from UGC text in a recommender system, each item \( j \) will also own a term count vector \( \mathbf{e}_j = (n_{t_{j1}}, ..., n_{t_{jL}}) \). In \( \mathbf{e}_j \), \( n_{t_l} \) (\( 1 \leq l \leq L \)) is the times of \( t_l \) that item \( j \) received in its UGC text. If item \( j \) has never received term \( t_l \) before, \( n_{t_l} \) in \( \mathbf{e}_j \) will be 0. Similar to the user-term co-occurrence matrix (see Table 2, in Section 4.1), we construct an item-term co-occurrence matrix, in which each row is \( \mathbf{e}_j \). A toy example is shown in Table 3.

![Probabilistic graphical model of UCR](image)

**Figure 1:** The probabilistic graphical model of UCR

<table>
<thead>
<tr>
<th>space</th>
<th>science</th>
<th>...</th>
<th>love</th>
</tr>
</thead>
<tbody>
<tr>
<td>Titanic</td>
<td>0</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>Gravity</td>
<td>25</td>
<td>10</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Interstellar</td>
<td>30</td>
<td>15</td>
<td>...</td>
</tr>
</tbody>
</table>

Table 3: A toy example of the item-term co-occurrence matrix

In this section, we propose to estimate \( R_{ij} \) via four parts, i.e., latent user feature vector \( \mathbf{u}_i \), user topic proportion \( \mathbf{\theta}_u^i \), latent item feature vector \( \mathbf{v}_j \) and item topic proportion \( \mathbf{\theta}_v^j \). \( \mathbf{\theta}_u^i = (\theta_{u1}^i, ..., \theta_{uk}^i)^T \), and we define \( \mathbf{\theta}_u^i = \{\mathbf{\theta}_u^i\} \) (\( 1 \leq i \leq M \)). \( \mathbf{\theta}_v^j = (\theta_{v1}^j, ..., \theta_{vk}^j)^T \), and we define \( \mathbf{\theta}_v^j = \{\mathbf{\theta}_v^j\} \) (\( 1 \leq j \leq N \)). \( \mathbf{\theta}_u^i \) and \( \mathbf{\theta}_v^j \) denote the user- and item-topic distribution respectively. It can be seen that \( \mathbf{\theta}_v^j \) is defined in the same way as \( \mathbf{\theta}_i \) in Section 3.2, and \( \mathbf{\theta}_u^i \) is defined in the same way as \( \mathbf{\theta}_i \) in Section

\(^9\)http://www.amazon.com/
4.1. The estimation of $R_{ij}$ is shown below.

$$R_{ij} \approx (\vec{u}_i + \vec{\theta}_u) \cdot (\vec{v}_j + \vec{\theta}_v) = \vec{u}_i^T \vec{v}_j + \vec{d}_u^T \vec{\theta}_u \cdot \vec{v}_j + (\vec{\theta}_u)^T \vec{\theta}_v$$ (4)

Eq. 4 clearly illustrates how the two topic proportions $\vec{\theta}_u$ and $\vec{\theta}_v$ affect the rating prediction. $\vec{u}_i^T \vec{\theta}_u$ predicts how user $i$ is interested in item $j$’s aspects. $(\vec{\theta}_u)^T \vec{v}_j$ measures how close user $i$’s interested topics are to item $j$’s features that are learned from rating records. $(\vec{\theta}_v)^T \vec{\theta}_v$ measures how close user $i$’s interested topics are to item $j$’s aspects.

We give its generative process below:

1. For each user $i$ ($1 \leq i \leq M$),
   (a) sample user topic proportion vector $\vec{\theta}_u \sim \text{Dir}(\alpha)$
   (b) sample user latent feature vector $\vec{u}_i \sim \mathcal{N}(0, \lambda_u^{-1}I_K)$
2. For each item $j$ ($1 \leq j \leq N$),
   (a) sample item topic proportion vector $\vec{\theta}_v \sim \text{Dir}(\alpha)$
   (b) sample item latent feature vector $\vec{v}_j \sim \mathcal{N}(0, \lambda_v^{-1}I_K)$
3. For term $t_{il}$ that user $i$ gave in his or her UGC text,
   (a) sample topic assignment $z_{il} \sim \text{Mult}(\vec{\theta}_u)$
   (b) sample term $t_{il} \sim \text{Mult}(\vec{\beta}_u z_{il})$
4. For term $t_{jl}$ that item $j$ received in its UGC text,
   (a) sample topic assignment $z_{jl} \sim \text{Mult}(\vec{\theta}_v)$
   (b) sample term $t_{jl} \sim \text{Mult}(\vec{\beta}_v z_{jl})$
5. For each rating $R_{ij}$, sample $R_{ij} \sim \mathcal{N}((\vec{u}_i + \vec{\theta}_u)^T (\vec{v}_j + \vec{\theta}_v), c_{ij}^{-1})$

In the above process, $\vec{\beta}_u = (\beta_{u1}, \ldots, \beta_{uk}, \ldots, \beta_{ul})$ denotes the topic-term distribution learned from users’ UGC texts, and we define $\vec{\beta}_v = (\beta_{v1}, \ldots, \beta_{v1}, \ldots, \beta_{vl})$ denotes the topic-term distribution learned from items’ UGC texts, and we define $\vec{\beta}_v = (\beta_{v1}) (1 \leq k \leq K)$. Similar to our first model UCR (see Section 4.1), we title this model Joint UGC-based Collaborative Recommendation Model (denoted by J-UCR), and give its probabilistic graphical model in Figure 2.

**Parameters Estimation.** The parameter estimation algorithm of J-UCR is very similar to that of UCR (the first proposed model). Thus, we only present the parameter estimation algorithm for J-UCR. The parameter estimation algorithm of UCR can be naturally derived based on the following algorithm. Let $T$ denote the term set that contains all terms in UGC text. Given parameters $c$, $\lambda_u$, $\lambda_v$, $\beta_u$, $\beta_v$ and the observed $R$, $T$, we aim to learn the maximum a posteriori (MAP) estimate of $u$, $v$, $\theta_u$ and $\theta_v$. The
Inspired by the parameter estimation procedure in CTR, we propose an EM (Expectation-Maximization) style algorithm to estimate the parameters. Maximizing the above posterior (see Eq. 5) is equivalent to maximizing the following log likelihood, which is given as

\[
E = -\frac{1}{2} \sum_{i=1}^{M} \sum_{j=1}^{N} c_{ij} (R_{ij} - (\bar{u}_i + \bar{\theta}_u^i)^T (\bar{v}_j + \bar{\theta}_v^j))^2 
- \frac{\lambda_u}{2} \sum_{i=1}^{M} \bar{u}_i^T \bar{u}_i - \frac{\lambda_v}{2} \sum_{j=1}^{N} \bar{v}_j^T \bar{v}_j 
+ \sum_{l=1}^{L} \sum_{k=1}^{K} \log \left( \sum_{u=1}^{U} \theta_{uk}^l \beta_{k,l}^u \right) + \sum_{j=1}^{N} \sum_{l=1}^{L} \log \left( \sum_{k=1}^{K} \theta_{jk}^l \beta_{k,l}^v \right)
\]

The hyperparameter \( \alpha \) is set to 1, so both \( p(\theta^u|\alpha) \) and \( p(\theta^v|\alpha) \) (see Eq. 5) become constants (remember that both \( p(\theta^u|\alpha) \) and \( p(\theta^v|\alpha) \) follow the Dirichlet distribution). The constants are left out, since they do not make any influence to the final result. Naturally, maximizing Eq. 6 is also equivalent to minimizing the following objective.
Since Eq. 7 is not convex, to achieve its global optima is NP-hard. We propose to use gradient descent algorithm to achieve a local optimum of Eq. 7 in an iterative way, as well as to optimize $u$, $v$, $\theta^u$ and $\theta^v$. Given the estimates of $\theta^u$, $\theta^v$, the partial derivatives of Eq. 7 over $\ddot{u}_i$, $\ddot{v}_j$ are

$$
\begin{align*}
\frac{\partial E}{\partial \ddot{u}_i} &= \sum_{j=1}^{N} c_{ij}(\ddot{u}_i + \ddot{v}_j)^T (\ddot{v}_j + \ddot{\theta}_j^v) - R_{ij}(\ddot{v}_j + \ddot{\theta}_j^v) + \lambda_u \ddot{u}_i \\
\frac{\partial E}{\partial \ddot{v}_j} &= \sum_{i=1}^{M} c_{ij}(\ddot{u}_i + \ddot{v}_j)^T (\ddot{v}_j + \ddot{\theta}_j^v) - R_{ij}(\ddot{u}_i + \ddot{\theta}_j^u) + \lambda_v \ddot{v}_j
\end{align*}
$$

Following the update rule in gradient descent algorithm, in each iteration, $\ddot{u}_i$ and $\ddot{v}_j$ are updated as

$$
\begin{align*}
\ddot{u}_i &= \dddot{u}_i - \rho \times \frac{\partial E}{\partial \ddot{u}_i} \\
\ddot{v}_j &= \dddot{v}_j - \rho \times \frac{\partial E}{\partial \ddot{v}_j}
\end{align*}
$$

where $\dddot{u}_i$, $\dddot{v}_j$ are the updated vectors of $\dddot{u}_i$, $\dddot{v}_j$ respectively, and $\rho$ is the learning rate.

In an iteration, after the update of $\dddot{u}_i$, $\dddot{v}_j$, we use the similar gradient descent algorithm to learn the updated topic proportions $\theta^u$, $\theta^v$. First, we isolate the parts that contain $\dddot{\theta}_i^u$ and $\dddot{\theta}_j^v$ separately from Eq. 7, to construct the objective functions specific to $\dddot{\theta}_i^u$ and $\dddot{\theta}_j^v$ as follows:

$$
\begin{align*}
E(\dddot{\theta}_i^u) &= \frac{1}{2} \sum_{j=1}^{N} c_{ij}(R_{ij} - (\dddot{u}_i + \dddot{v}_j)^T (\dddot{v}_j + \dddot{\theta}_j^v))^2 \\
&\quad - \sum_{l=1}^{L} \log(\sum_{k=1}^{K} \theta^u_{il} \beta_{kl}) \\
\end{align*}
$$

$$
\begin{align*}
E(\dddot{\theta}_j^v) &= \frac{1}{2} \sum_{i=1}^{M} c_{ij}(R_{ij} - (\dddot{u}_i + \dddot{v}_j)^T (\dddot{v}_j + \dddot{\theta}_j^v))^2 \\
&\quad - \sum_{l=1}^{L} \log(\sum_{k=1}^{K} \theta^v_{lj} \beta_{kl})
\end{align*}
$$
Since the inference procedures of $\vec{\theta}^u_i$ and $\vec{\theta}^v_j$ are very similar to each other, in the remainder of this section, we only focus on the estimation of $\vec{\theta}^u_i$. The estimation of $\vec{\theta}^v_j$ can be deduced naturally referring to the following procedure of $\vec{\theta}^u_i$. Because $\log(x)$ function is a concave function, according to Jensen’s inequality, we have

$$\log(\sum_{k=1}^{K} \theta^u_{ik} \beta^u_k) \geq \sum_{k=1}^{K} \phi^i_k (\log \theta^u_{ik} \beta^u_k)$$

where $\phi^i_k = p(z_{il} = k)$ represents the probability of term $t_{il}$ assigned to topic $k$ in user $i$’s UGC text. Each user $i$ owns a matrix $\phi^i = \{\phi^i_{lk}\} (1 \leq l \leq L, 1 \leq k \leq K)$, where $K$ is the number of topics. Considering the above inequality (see Eq. 12) and $E(\vec{\theta}^u_i)$ (see Eq. 10), we have

$$E(\vec{\theta}^u_i) \leq \frac{1}{2} \sum_{j=1}^{N} c_{ij} (R_{ij} - (\vec{u}_i + \vec{\theta}^u_i \vec{v}_j + \vec{\theta}^v_j)^2$$

$$- \sum_{l=1}^{L} \sum_{k=1}^{K} \phi^i_k (\log \theta^u_{ik} \beta^u_k - \log \phi^i_k) = E(\vec{\theta}^u_i, \phi^i)$$

Eq. 13 indicates that $E(\vec{\theta}^u_i, \phi^i)$ gives $E(\vec{\theta}^u_i)$ a tight upper bound. Again, we cannot get a closed-form solution of $\vec{\theta}^u_i$ and $\phi^i$ in $E(\vec{\theta}^u_i, \phi^i)$. We use the gradient descent algorithm to find a minimum of $E(\vec{\theta}^u_i, \phi^i)$. The partial derivative of $E(\vec{\theta}^u_i, \phi^i)$ over $\theta^u_{ik}$ is given in Eq. 14. Note that, we take the natural logarithm (i.e. $\ln(x)$) as the default logarithm function in Eq. 13.

$$\frac{\partial E(\vec{\theta}^u_i, \phi^i)}{\partial \theta^u_{ik}} = \sum_{j=1}^{N} c_{ij} ((\vec{u}_i + \vec{\theta}^u_i \vec{v}_j) - R_{ij}) (v_j + \theta^v_j)$$

$$- \sum_{l=1}^{L} \phi^i_k \theta^u_{ik}$$

where $v_{jk}$ and $\theta^v_{jk}$ are the $k$th elements of $\vec{v}_j$ and $\vec{\theta}^v_j$ separately. $\theta^u_{ik}$ is updated in each iteration in the way analogous to the iterative update of $\vec{u}_i$ and $\vec{v}_j$ (see Eq. 9). As for $\phi^i$, according to the Bayes’ rule (remember that $\phi^i_{lk} = p(z_{il} = k)$), $\phi^i_{lk}$ satisfies

$$\phi^i_{lk} \propto \theta^u_{ik} \beta^u_k$$

After $\phi^i$ is estimated, $\beta$ can be estimated as

$$\beta^u_{kl} \propto \sum_{i=1}^{M} \sum_{j=1}^{L} \phi^i_{lk} I[t_{il} = t]$$

where $I[t_{il} = t]$ is an indicator function. If term $t_{il}$ is in user $i$’s UGC text, and is
exactly the term \( t \), then \( \mathbf{1}[t_{ij} = t] \) is equal to 1. Otherwise, \( \mathbf{1}[t_{ij} = t] \) is equal to 0. The inference process for estimating \( \beta_{k}^{u} \) is the same with that in the EM (Expectation-Maximization) update of LDA. For more details, please refer to the VI-based EM (Variational Inference-based EM) algorithm in (Blei et al., 2003).

5. Experiments

In this section, we evaluate our proposed models’ performance on three real-world datasets.

5.1. Dataset

Dataset. We conduct the experiments on three real-world datasets, which are Movielens dataset, Last.fm dataset and Yelp dataset. Movielens dataset and Last.fm dataset are released by (Cantador et al., 2011), and both can be downloaded in Grouplens\(^{10}\). The MovieLens dataset contains ratings, social tags, and movies’ various attributes. The Last.fm dataset contains social tags and listening counts of users to artists. We expand the Last.fm dataset by crawling every artist’s introduction from the web site of Last.fm, since CTR model (Wang and Blei, 2011) (a baseline) needs item descriptions as input. Meanwhile, we map each listening count into the integer range 1~5 as a rating. The Yelp dataset is publicly released by Yelp\(^{11}\), which contains ratings, reviews and restaurants’ attributes. The details of the three datasets are presented in Table 4.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#user</th>
<th>#item</th>
<th>#term</th>
<th>rating density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movielens</td>
<td>2113</td>
<td>10197</td>
<td>6574</td>
<td>3.97%</td>
</tr>
<tr>
<td>Last.fm</td>
<td>1892</td>
<td>17632</td>
<td>11946</td>
<td>0.29%</td>
</tr>
<tr>
<td>Yelp</td>
<td>23152</td>
<td>11537</td>
<td>28475</td>
<td>0.09%</td>
</tr>
</tbody>
</table>

Table 4: Statistics of the three datasets

In Table 4, \#term represents the number of unique words or phrases in the UGC text. In the Movielens dataset and Last.fm dataset, since there are unigrams and n-grams \( (n \geq 2) \) in social tags, the “term” in Table 4 may refer to a single word, or a phrase. In the Yelp dataset, we remove the stop words\(^{12}\) from the reviews. rating density denotes the proportion of ratings that have been known in the whole user-item rating matrix (see Table 1, in Section 3.1). It can be seen that in the real-world recommendation scenario, the user-item rating matrix is usually extremely sparse.

Training/Testing Set. For the Movielens dataset and Yelp dataset, we partition each user’s rating records into the training set and testing set according to the timestamp. For example, for the training set of 90% density in Section 5.4, we select the latest 10% ratings as testing data, and the remaining 90% old records will be training data. For the Last.fm dataset, since there are no timestamps related with rating records, we randomly select 10% as testing data, and the remaining data will be training data.

\(^{10}\)http://grouplens.org/datasets/hetrec-2011/
\(^{11}\)https://www.kaggle.com/c/yelp-recsys-2013
\(^{12}\)http://www.nltk.org/book/ch02.html
5.2. Parameter Setting

For all three datasets, the default parameter setting is the same. The topic number and latent feature number $K$ is set to 10, and $\lambda_u$, $\lambda_v$ are set to 1 equally. $c_{ij}$ is set to 1 when $R_{ij}$ exists (i.e., $a = 1$, see Eq. 1), and 0 when $R_{ij}$ does not exist (i.e., $b = 0$, see Eq. 1). Note that, in the traditional CTR model (Wang and Blei, 2011), when $R_{ij}$ does not exist, $c_{ij}$ is still set to a non-zero value, for example, $b = 0.01$. However, in this paper, our preliminary experiments found that our models’ performance is not sensitive to the small value of $b$. So for easy implementation, we set $b = 0$. Besides, as mentioned in Section 4.2, the Dirichlet hyperparameter $\alpha$ is set to 1.

5.3. Evaluation Metrics

In the following experiments, Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are chosen as the evaluation metrics, which are defined as follows:

$$RMSE = \sqrt{\frac{1}{|T_s|} \sum_{R_{ij} \in T_s} (R_{ij} - \hat{R}_{ij})^2}, \quad MAE = \frac{1}{|T_s|} \sum_{R_{ij} \in T_s} |R_{ij} - \hat{R}_{ij}|$$

where $T_s$ represents the testing set, and $R_{ij}$, $\hat{R}_{ij}$ are the real and predicted ratings. A smaller RMSE value or MAE value means a better performance.

5.4. Performance Comparison

The compared baseline models are listed below:

1. **ItemCF**: this is the memory-based collaborative filtering algorithm specific to the item side (Sarwar et al., 2001).

2. **UserCF**: this is the memory-based collaborative filtering algorithm specific to the user side (Resnick et al., 1994). UserCF and ItemCF, as well as other models that are extended from UserCF or ItemCF, are widely used in recommender systems (Ricci et al., 2010).

3. **PMF**: this is the Probabilistic Matrix Factorization (PMF) model proposed in (Salakhutdinov and Mnih, 2008). As a base model of our proposed models, a brief introduction of PMF has been given in Section 3.1.

4. **MMMF**: this is the Maximum Margin Matrix Factorization (MMMF) model proposed in (Rennie and Srebro, 2005). As a well-known recommendation algorithm, MMMF has been used or extended successfully in many recommendation tasks (Weimer et al., 2009; Xu et al., 2013; Chen et al., 2014b).

5. **CTR**: this is the Collaborative Topic Regression (CTR) model proposed in (Wang and Blei, 2011). As the other base model, a brief introduction of CTR has been given in Section 3.2.

Tables 5, 6 and 7 show the prediction errors achieved by our models and the baselines on the Movielens dataset, Last.fm dataset and Yelp dataset respectively. We can make several observations from Table 5, 6 and 7 as follows:
Table 5: Accuracy comparison on the Movielens dataset (a smaller value means a better performance)

<table>
<thead>
<tr>
<th>Approach</th>
<th>TD=60%</th>
<th>TD=70%</th>
<th>TD=80%</th>
<th>TD=90%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RMSE</td>
<td>MAE</td>
<td>RMSE</td>
<td>MAE</td>
</tr>
<tr>
<td>ItemCF</td>
<td>0.8885</td>
<td>0.6421</td>
<td>0.8722</td>
<td>0.6309</td>
</tr>
<tr>
<td>UserCF</td>
<td>0.8405</td>
<td>0.6290</td>
<td>0.8303</td>
<td>0.6213</td>
</tr>
<tr>
<td>PMF</td>
<td>0.8312</td>
<td>0.6259</td>
<td>0.8225</td>
<td>0.6191</td>
</tr>
<tr>
<td>MMMF</td>
<td>0.8303</td>
<td>0.6250</td>
<td>0.8213</td>
<td>0.6180</td>
</tr>
<tr>
<td>CTR</td>
<td>0.8211</td>
<td>0.6184</td>
<td>0.8109</td>
<td>0.6103</td>
</tr>
<tr>
<td>UCR</td>
<td>0.8155</td>
<td>0.6145</td>
<td>0.8089</td>
<td>0.6078</td>
</tr>
<tr>
<td>J-UCR</td>
<td>0.8096</td>
<td>0.6089</td>
<td>0.7992</td>
<td>0.6012</td>
</tr>
</tbody>
</table>

Table 6: Accuracy comparison on the Last.fm dataset (A smaller value means a better performance)

<table>
<thead>
<tr>
<th>Approach</th>
<th>TD=60%</th>
<th>TD=70%</th>
<th>TD=80%</th>
<th>TD=90%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RMSE</td>
<td>MAE</td>
<td>RMSE</td>
<td>MAE</td>
</tr>
<tr>
<td>ItemCF</td>
<td>0.5832</td>
<td>0.4641</td>
<td>0.5779</td>
<td>0.4601</td>
</tr>
<tr>
<td>UserCF</td>
<td>0.5471</td>
<td>0.3787</td>
<td>0.5436</td>
<td>0.3749</td>
</tr>
<tr>
<td>PMF</td>
<td>0.5121</td>
<td>0.3941</td>
<td>0.5091</td>
<td>0.3913</td>
</tr>
<tr>
<td>MMMF</td>
<td>0.5119</td>
<td>0.3936</td>
<td>0.5028</td>
<td>0.3867</td>
</tr>
<tr>
<td>CTR</td>
<td>0.4815</td>
<td>0.3702</td>
<td>0.4698</td>
<td>0.3608</td>
</tr>
<tr>
<td>UCR</td>
<td>0.4733</td>
<td>0.3626</td>
<td>0.4617</td>
<td>0.3527</td>
</tr>
<tr>
<td>J-UCR</td>
<td>0.4713</td>
<td>0.3575</td>
<td>0.4601</td>
<td>0.3497</td>
</tr>
</tbody>
</table>

Table 7: Accuracy Comparison on the Yelp dataset (A smaller value means a better performance)

<table>
<thead>
<tr>
<th>Approach</th>
<th>TD=60%</th>
<th>TD=70%</th>
<th>TD=80%</th>
<th>TD=90%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RMSE</td>
<td>MAE</td>
<td>RMSE</td>
<td>MAE</td>
</tr>
<tr>
<td>PMF</td>
<td>1.2504</td>
<td>0.9939</td>
<td>1.2475</td>
<td>0.9917</td>
</tr>
<tr>
<td>MMMF</td>
<td>1.2332</td>
<td>0.9726</td>
<td>1.2292</td>
<td>0.9695</td>
</tr>
<tr>
<td>CTR</td>
<td>1.2199</td>
<td>0.9705</td>
<td>1.2102</td>
<td>0.9635</td>
</tr>
<tr>
<td>UCR</td>
<td>1.2031</td>
<td>0.9565</td>
<td>1.2009</td>
<td>0.9547</td>
</tr>
<tr>
<td>J-UCR</td>
<td>1.1757</td>
<td>0.9289</td>
<td>1.1712</td>
<td>0.9274</td>
</tr>
</tbody>
</table>

1. On the three datasets, our proposed model UCR achieves lower prediction errors than all baseline models. It indicates that the UGC text can indeed effectively learn a user’s preference.
2. Our proposed model \textit{J-UCR} consistently outperforms all baseline models on the three datasets. For a detailed example, in the Yelp dataset, on average, \textit{J-UCR} achieves 11.79\% improvement in MAE than \textit{PMF} and 4.34\% improvement in MAE than \textit{CTR}. On the three datasets, the improvements achieved by \textit{J-UCR} over all baselines are significant according to paired \textit{t}-test ($p < 0.0001$). Additionally, \textit{J-UCR} behaves better than \textit{UCR} in all cases. Such an improvement verifies the following three points: 1) UGC text can also help learn an item’s aspects. 2) Modeling users’ interested topics and items’ aspects jointly is effective in the rating prediction task. 3) The EM-style algorithm of parameter estimation is effective.

3. The RMSE and MAE values become smaller along with the increase of the training set density. That is expected since more training data can help learn user and item latent features more accurately.

5.5. Impact of $K$

The parameter $K$ denotes the number of topics and latent features. If $K$ is too small, the topics will be too coarse-grained and some useful features will be missed. In contrast, if $K$ is too large, some valuable features will be broken down into too many small features. There are some existing works using nonparametric topic modeling technique to infer the number of topics (Teh et al., 2006; Buntine and Mishra, 2014). However, they are also sensitive to the hyperparameter choice. We tried different values of $K$ and investigated its impact on our proposed model \textit{J-UCR}. The experiments are conducted on the MovieLens dataset and Yelp dataset under the case that the training set density (TD for short) is 80\% and 90\%. All other parameters are set to the values in the default parameter setting.

Figures 3 and 4 show that on the two datasets, along with the increase of $K$, RMSE and MAE values first decrease and reach minima (where $K$ is equal to 8 on the MovieLens dataset, and $K$ is equal to 10 on the Yelp dataset), and then raise when $K$ continues to increase. It demonstrates that on different datasets, \textit{J-UCR} can get satisfactory performance with a relatively small topic/latent feature number, which enhances the
applicability of our model. On the one hand, our model can be adapted in different recommendation scenarios. On the other hand, a small $K$ can accelerate the generation of recommendation results. Also, it can be inferred that in the above two recommender systems, the number of latent factors that can impact a user’s decision is limited.

5.6. Impact of $\lambda_u$, $\lambda_v$

In the log likelihood (see Eq. 7, in Section 4.2), $\frac{1}{2} \sum_{m=1}^{M} u_i^T u_i$ and $\frac{1}{2} \sum_{n=1}^{N} v_j^T v_j$ act as two regularization terms, which are used to avoid the overfitting problem during the parameter estimation procedure. The parameters $\lambda_u$, $\lambda_v$ control the penalty to latent feature vectors $\mathbf{u}_i$ and $\mathbf{v}_j$. If $\lambda_u$, $\lambda_v$ are large, $\mathbf{u}_i$, $\mathbf{v}_j$ will be penalized largely to be close to the zero vector, which is the mean of the Gaussian distribution $\mathcal{N}(0, \lambda_u^{-1} I_K)$ and $\mathcal{N}(0, \lambda_v^{-1} I_K)$. Then the topic proportions $\theta^u$ and $\theta^v$ will take a major role in rating prediction. That is to say, we trust the latent topics learned from the UGC text more than the latent features collaboratively learned from the rating records. If $\lambda_u$, $\lambda_v$ are small, $\mathbf{u}_i$, $\mathbf{v}_j$ will tend to diverge from the zero-mean vector. In this section, we investigate the impact of $\lambda_u$, $\lambda_v$ on the performance of our proposed model $J$-UCR. $\lambda_u$ and $\lambda_v$ are assigned the same value in all cases. The experiments are conducted on the Movielens dataset and Yelp dataset under the case that the training set density (TD) is 60%, 70%, 80% and 90%. All other parameters are set to the values in the default parameter setting. In Figures 5 and 6, the bars in each group from left to right represent the value of $\lambda_u$, $\lambda_v$ as 5, 1, 0.1, 0.01 and 0.001 respectively.

From Figures 5 and 6, it can be seen that on the two datasets, $J$-UCR achieves lowest prediction errors when $\lambda_u$ and $\lambda_v$ are equal to 1. It indicates that a too small $\lambda_u$ or $\lambda_v$ (for example, $\lambda_u = \lambda_v = 0.001$ in Figures 5 and 6) cannot exert enough penalty to avoid the overfitting problem. In such a case, $\mathbf{u}_i$ and $\mathbf{v}_j$ tend to be overwhelmed $\theta^u$ and $\theta^v$ in predicting missing ratings (see Eq. 4 in Section 4.2). That is, the latent topics learned from the UGC text does not make an enough contribution to the rating prediction. However, if $\lambda_u$ and $\lambda_v$ are too large (for example, $\lambda_u = \lambda_v = 5$ in Figures 5 and 6), $\mathbf{u}_i$ and $\mathbf{v}_j$ will be penalized too largely. The user and item latent features, which are collaboratively learned, tend to be ignored in the rating prediction.
Efficiency Analysis

This section studies the efficiency of the proposed parameter estimation algorithm (see Section 4.2). As an EM-style algorithm, our parameter estimation algorithm is implemented in an iterative manner. Thus, the efficiency is determined by the convergence and time cost per iteration. First, we report the experimental results on the convergence of the J-UCR model in the following Figures 7 and 8. The experiments are conducted on the Movielens dataset under the case that the training set density (TD) is 80% (see Figure 7) and 90% (see Figure 8). All parameters are set to the values in the default parameter setting.

From Figures 7 and 8, we can observe that for both training set densities, our proposed model J-UCR converges with less than 500 iterations. It indicates that with our default parameter setting, J-UCR has the capability of overcoming the overfitting problem in parameter estimation, and converging to a local optimum. Another observation is that the prediction errors, both RMSE and MAE, decrease sharply after the initialization (i.e., iteration = 0) of the parameter estimation algorithm. It illustrates that our algorithm can quickly find the right direction to the optimum in the solution space.
Next, we present the time cost per iteration of our proposed models (UCR and J-UCR) compared to the baseline model CTR. The experiments are conducted on the Movielens dataset, and on a normal Dell computer with Intel Core i5-2500 CPU (3.30 GHz, Dual Core) and 8 Giga Byte (GB) memory. All parameters are set to the values in the default parameter setting. We report the results under the case that the training set density is 90% and 60% in Table 8.

### Table 8: Time cost per iteration

<table>
<thead>
<tr>
<th>Model</th>
<th>CTR</th>
<th>UCR</th>
<th>J-UCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training set density = 90%</td>
<td>1.475s</td>
<td>3.469s</td>
<td>5.213s</td>
</tr>
<tr>
<td>Training set density = 60%</td>
<td>1.124s</td>
<td>3.186s</td>
<td>4.481s</td>
</tr>
</tbody>
</table>

From Table 8, we can observe that the time cost per iteration of CTR is less than that of UCR or J-UCR. Such a result is to be expected due to the following two reasons. First, the UGC text has a larger volume than the item descriptions, which are the input of the CTR model. Second, CTR or UCR only takes the text of either items or users as the input. In contrast, J-UCR takes the UGC text of users and items as the input, and conducts the task of jointly learning user and item topics/aspects.
5.8. Sensitivity Analysis to the Learning Rate

The learning rate $\rho$ (see $\rho$ in Eq. 9, Section 4.2) controls the learning step in each iteration of the gradient descent algorithm. A too small $\rho$ is likely to slow the searching for the optimum. A too large $\rho$ likely leads the algorithm to skipping the optimum, and further leads to the overfitting problem. This section investigates the sensitivity of our proposed model $J$-$UCR$ to the learning rate. The experiments are conducted on the Movielens dataset under the case that the training set density (TD) is 80% and 90%. The parameter estimation algorithm is implemented in 500 iterations. All other parameters are set to the values in the default parameter setting. The experimental results are shown in Figures 9 and 10.

As shown in Figures 9 and 10, under both training set densities, the lowest prediction errors (RMSE and MAE) are achieved in the range of $0.00020 \sim 0.00024$. Also, the variation of the prediction errors is smooth and small in the whole range of $\rho$ (i.e., $0.00012 \sim 0.00026$). Note that, even the worst performance of $J$-$UCR$ that is achieved at $\rho = 0.00012$ is better than the performance of the baselines. For example, in the case of 90% training set density (see Figure 10 (a)), the highest RMSE value is 0.7935 (achieved at $\rho = 0.00012$), which is still lower than the RMSE value achieved by $CTR$.  

![Figure 9: Sensitivity to the learning rate (training set density = 80%)](image)

![Figure 10: Sensitivity to the learning rate (training set density = 90%)](image)
and UCR (see Table 5 in Section 5.4). It indicates that J-UCR is relatively insensitive to the learning rate, and can achieve superior results in a wide range of learning rate.

5.9. Performance Evaluation on Different Rating Scales

As a challenging problem, the “cold-start” problem widely exists in recommender systems (Bobadilla et al., 2013). It refers that users only have very limited ratings, for example, less than 5 ratings. So it is hard to make satisfactory recommendation for them. Without enough training data, many previous works, which are only based on collaborative filtering, usually give prediction results with low accuracy.

We evaluate the performance of our proposed model J-UCR in a series of user groups with different rating scales on the Yelp dataset. We categorize users into different groups based on the number of their observed ratings in the training dataset (80% and 90%). Figure 11 shows the log-log distribution of the number of observed ratings. Figures 12(b) and 13(b) show the distribution of the number of observed ratings by groups. It can be clearly seen that a large part of users have very limited ratings, while only a small part of users have a large number of ratings. For example, in the case of 90% training set density (see Figure 13(b)), 75.67% users have no more than 5 ratings. The experiments are conducted on the Yelp dataset under the case that training set density (TD) is 80% and 90%. All parameters are set to the values in the default parameter setting.

Figures 12(a) and 13(a) demonstrate that J-UCR consistently outperforms CTR and PMF in all user groups. Especially, J-UCR behaves much better than the two baselines in the “cold-start” scenario, such as the “1-2” group and “3-4” group. Since in the “cold-start” scenario, a user has very limited reviews, such an improvement tells us that even a single or two reviews may contain very valuable information. For example, from the single review “I really like this computer, because it’s quite suitable to use during traveling”, we can infer that this user likes traveling or will go traveling recently. So, this review provides us a precise opportunity to recommend outdoor equipments or mobile devices to him.

![Figure 11: Distribution of the number of observed ratings](image)

Figure 11: Distribution of the number of observed ratings

Figures 12(a) and 13(a) demonstrate that J-UCR consistently outperforms CTR and PMF in all user groups. Especially, J-UCR behaves much better than the two baselines in the “cold-start” scenario, such as the “1-2” group and “3-4” group. Since in the “cold-start” scenario, a user has very limited reviews, such an improvement tells us that even a single or two reviews may contain very valuable information. For example, from the single review “I really like this computer, because it’s quite suitable to use during traveling”, we can infer that this user likes traveling or will go traveling recently. So, this review provides us a precise opportunity to recommend outdoor equipments or mobile devices to him.
5.10. Topic Discovery

In addition to achieving higher prediction accuracy, our proposed models can also generate interpretable aspects/topics for items and users. Usually, under an aspect or a topic, only the top terms with large probabilities in the topic-term distribution (i.e., $\beta_u$ and $\beta_v$ in J-UCR) are reliable. Other terms with small probabilities may not be truly related to the topic, since their probabilities may be acquired through the smoothing effect of the Dirichlet hyperparameters. In this section, we show the item aspects that are composed of top terms, which are learned by our proposed model J-UCR from the item-term co-occurrence matrix (see Table 3, in Section 4.2). The experiments are conducted on the Last.fm dataset and Yelp dataset, and the parameter setting is the same with the default setting specified in Section 5.2. We show 7 aspects out of the whole aspects in Tables 9 and 10. The aspect names are labeled manually, and exhibited in the first row of the two tables.

It can be seen that the aspects in Tables 9 and 10 are highly interpretable. For example, in Table 9, in the first column, heavy metal, metalcore, thrash metal black metal and power metal clearly form the aspect Metal Music. Note that in the item-
topic distribution $\theta^v$ (see $\theta^v$ in Section 4.2), $\tilde{\theta}^v_j$ illustrates the probability distribution of aspects in item $j$. So those aspects with large probabilities in $\tilde{\theta}^v_j$ can be uncovered, and further we can recognize item $j$’s main aspects since all aspects have been interpreted by words or phrases. We can also acquire the topics that are learned from the user-term co-occurrence matrix (see Table 2, in Section 4.1). However, since a user $i$ may be interested in more than one topic, the terms in the collection of user $i$’s UGC text (i.e., $\tilde{e}_i$, see Section 4.1) are usually given to various items with different aspects. Thus, the aspects in Tables 9 and 10 are more interpretable.

<table>
<thead>
<tr>
<th>Metal</th>
<th>Rock</th>
<th>Country</th>
<th>Band</th>
<th>Blues</th>
<th>Hip-hop</th>
<th>Punk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Music</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy metal</td>
<td>indie rock</td>
<td>indie</td>
<td>band</td>
<td>classic</td>
<td>rap</td>
<td>punk</td>
</tr>
<tr>
<td>metalcore</td>
<td>rock</td>
<td>country</td>
<td>guitar</td>
<td>blue</td>
<td>hip-hop</td>
<td>punk rock</td>
</tr>
<tr>
<td>trash metal</td>
<td>pop rock</td>
<td>mellow</td>
<td>bass</td>
<td>70s</td>
<td>rapper</td>
<td>pop punk</td>
</tr>
<tr>
<td>black metal</td>
<td>hard rock</td>
<td>beautiful</td>
<td>drum</td>
<td>guitar</td>
<td>lip</td>
<td>emo core</td>
</tr>
<tr>
<td>power metal</td>
<td>grunge</td>
<td>love</td>
<td>bassist</td>
<td>blues rock</td>
<td>American</td>
<td>hardcore</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ice Cream</th>
<th>Chinese Food</th>
<th>Drink</th>
<th>Parking</th>
<th>Vet</th>
<th>Japanese Food</th>
<th>Mexican Food</th>
</tr>
</thead>
<tbody>
<tr>
<td>cream</td>
<td>chicken</td>
<td>beer</td>
<td>park</td>
<td>dog</td>
<td>sushi</td>
<td>taco</td>
</tr>
<tr>
<td>flavor</td>
<td>rice</td>
<td>wine</td>
<td>area</td>
<td>staff</td>
<td>roll</td>
<td>Mexican</td>
</tr>
<tr>
<td>ice</td>
<td>Chinese</td>
<td>bar</td>
<td>place</td>
<td>pet</td>
<td>fish</td>
<td>salsa</td>
</tr>
<tr>
<td>chocolate</td>
<td>soup</td>
<td>drink</td>
<td>parking</td>
<td>doctor</td>
<td>fresh</td>
<td>burrito</td>
</tr>
<tr>
<td>sweet</td>
<td>noodle</td>
<td>patio</td>
<td>time</td>
<td>office</td>
<td>menu</td>
<td>tortilla</td>
</tr>
</tbody>
</table>

6. Conclusion and Future Work

This paper builds two UGC-based statistical models, which can leverage different types of UGC to make a connection between a user’s interested topics and an item’s aspects. In both models, besides the traditional rating records, the UGC text is incorporated into collaborative filtering algorithms (i.e. PMF) via topic modeling. Comprehensive experiments are conducted on three real-world datasets, and the experimental results demonstrate our models’ effectiveness. In detail, this paper first provides a unified way to use different types of UGC in a recommender system. Second, this paper verifies that social tags and user reviews are valuable resources to infer a user’s preference and an item’s aspects. Note that, although we test our models’ performance on the datasets of tags and reviews, other kinds of UGC can also be taken as our models’ inputs in the same way as stated in Sections 4.1 and 4.2. Third, the experimental results verify that the generative way of a rating and UGC text in our models is effective. Also, the parameter estimation algorithm is verified to be effective.
These achievements are also valuable to other related research fields. For example, in the future, we plan to investigate the function of tweets in inferring user preferences in the social networking site. Meanwhile, we also plan to apply our proposed models into other recommendation problems, such as group recommendation and friend recommendation.

Acknowledgements

This paper is supported by National Natural Science Foundation of China (No.61272129), National High-Tech Research Program of China (No.2013AA01A213), New-Century Excellent Talents Program Ministry of Education of China (No.NCET-12-0491), Zhejiang Provincial Natural Science Foundation (LR13F020002), Science and Technology Program of Zhejiang Province (No.2012C01037-1) and China Scholarship Council.

References


Pandora, 2015. Music genome project. URL http://www.pandora.com/about/mgp


