FLAME: A Probabilistic Model Combining Aspect Based Opinion Mining and Collaborative Filtering

Yao Wu
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
wuyaow@sfu.ca

Martin Ester
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
ester@cs.sfu.ca

ABSTRACT

Aspect-based opinion mining from online reviews has attracted a lot of attention recently. Given a set of reviews, the main task of aspect-based opinion mining is to extract major aspects of the items and to infer the latent aspect ratings from each review. However, users may have different preferences which might lead to different opinions on the same aspect of an item. Even if fine-grained aspect rating analysis is provided for each review, it is still difficult for a user to judge whether a specific aspect of an item meets his own expectation. In this paper, we study the problem of estimating personalized sentiment polarities on different aspects of the items. We propose a unified probabilistic model called Factorized Latent Aspect ModEl (FLAME), which combines the advantages of collaborative filtering and aspect based opinion mining. FLAME learns users' personalized preferences on different aspects from their past reviews, and predicts users' aspect ratings on new items by collective intelligence. Experiments on two online review datasets show that FLAME outperforms state-of-the-art methods on the tasks of aspect identification and aspect rating prediction.

Categories and Subject Descriptors

H.3.3 [**Information search and retrieval**]: Text Mining

General Terms

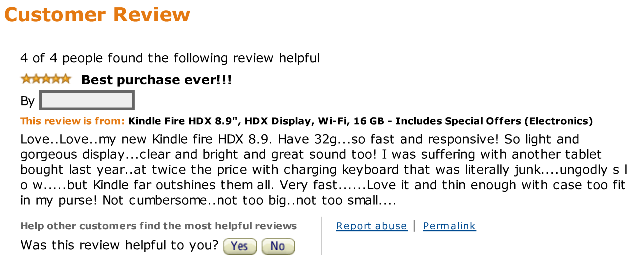
Algorithms, Experimentation

Keywords

Collaborative Filtering; Opinion Mining; Text Mining

1. INTRODUCTION

Nowadays, products and services offered on most online E-commerce websites are accompanied by abundant user-generated reviews, which can help users make better decision. For instance, if a user wants to know more about the battery life of a laptop, comments on the battery life of this specific laptop by other users are more reliable than those given in the official description of the product. However, the volume of reviews grows so rapidly that it gets extremely difficult for users to find useful information in short time. Thus mining useful information out of these huge amount of reviews has become an important way to improve user satisfaction of these online E-commerce websites.


Customer Review

4 of 4 people found the following review helpful

★★★★★ **Best purchase ever!!!**

By [redacted]

This review is from: **Kindle Fire HDX 8.9", HDX Display, Wi-Fi, 16 GB - Includes Special Offers (Electronics)**

Love..Love..my new Kindle fire HDX 8.9. Have 32g...so fast and responsive! So light and gorgeous display...clear and bright and great sound too! I was suffering with another tablet bought last year..at twice the price with charging keyboard that was literally junk....ungodly s l o w.....but Kindle far outshines them all. Very fast......Love it and thin enough with case too fit in my purse! Not cumbersome..not too big..not too small....

Help other customers find the most helpful reviews

Was this review helpful to you?

Figure 1: A Sample Review On Amazon

Aspect-based opinion mining [10] has attracted a lot of attention recently. Given a collection of reviews on a set of items, aspect-based opinion mining methods extract major aspects out of every item based on how often they have been commented by users, and learn users' sentiment polarities toward each aspect based on the opinionated words they used in the reviews. Figure 1 shows a sample review from Amazon[1]. The user assigns a 5-star overall rating, and expresses his opinions on several aspects of the product. From a set of reviews like this, aspect-based opinion mining methods can automatically extract the aspects of the product, such as *performance*, *display*, *value* and *size*, as well as infer latent sentiment scores for each aspect, e.g., 5 stars on its *display*.

Much work has been proposed to help users digest and exploit large number of reviews by aspect-based opinion mining techniques, including information extraction from reviews [11], uncovering the latent aspects of the review sentences [13], inferring the latent aspect ratings and aspect weights of each review document [22, 23], aspect-based review summarization for products [11, 9], etc. These methods either focus on *review-level* analysis (extracting useful information within each review) to help users easily find what they need from a piece of review, or make *product-level* summarization (aggregating the opinions of all the users) to provide an overview of users' feedback on a product. However, an important factor is typically ignored – *preference diver-*

WSDM'15, February 2–6, 2015, Shanghai, China.

http://dx.doi.org/10.1145/2684822.2685291.

---

[1] `http://www.amazon.com`

sity, i.e., users have different preferences that their opinions on the same item may differ from each other. For example, the *food* of the same restaurant might be delicious for some users but terrible for others. When choosing restaurants, a user might want to know whether a restaurant meets his *own* expectation on the aspect of *food*. But when facing a large number of reviews expressing various opinions, it becomes extremely difficult for the user to make the decision: 1) it's impossible for the user to read all the reviews even if the fine-grained *review-level* analysis is provided. 2) the user has no idea of which reviews are more reliable or which reviewers share similar tastes with him. 3) *product-level* summarization is also unreliable since it is generated from reviews by users with different tastes. To help users better utilize the existing reviews, we argue that a new method is required, which can learn a user's personalized preferences on different aspects from his past reviews of other items, and predict his preferences on the aspects of a given item by mining the opinions by other users with similar preferences.

A popular method of learning user preferences is *Collaborative Filtering*, which predicts a user's interests by collaboratively collecting preferences from many other users. Typical collaborative filtering methods take the numeric overall ratings as inputs [7, 19], assuming that users with the same ratings share the same tastes. However, two users who have assigned the same 5-stars to a restaurant might have significantly different reasoning; one might like its *food* while the other likes its *service*. Text reviews provide rich information to make it possible to understand preferences of users at a finer granularity. Some recent work has shown the benefits of utilizing text reviews within the collaborative filtering methods [12, 24]. Unlike these work, we aim at collectively exploring users' preferences on different aspects of the items. A challenge here is that aspect-based sentiment scores are not *explicitly* specified by users, but *implicitly* expressed in the reviews. We propose a new model combining aspect-based opinion mining and collaborative filtering to collectively learn users' preferences on different aspects.

In this work, we introduce the problem of *Personalized Latent Aspect Rating Analysis*. Given a collection of reviews of a set of items by a set of users, the goal is to solve the following two tasks: *a*) learn the latent aspects and their word distribution over a pre-defined vocabulary, and the latent aspect ratings for each review; *b*) for any user u in the data set, predict the latent aspect ratings on the items that he has not yet reviewed. Existing aspect-based opinion mining methods such as [22, 23, 14] are able to solve task a, but are unsuitable for solving task b since they require the text of user u's review for item i as input. Task b is also different from the well-studied rating prediction problem in recommender systems, the goal of which is to predict the overall rating while we want to predict the aspect ratings.

To address the problem of Personalized Latent Aspect Rating Analysis, we propose a unified probabilistic model called Factorized Latent Aspect ModEl (FLAME), which combines the advantages of both collaborative filtering and aspect-based opinion mining so that the two methods can mutually enhance each other. The general idea of FLAME is that we can learn users' preferences based on their past reviews, so that we can collaboratively predict a user's preference of an aspect of an item from the opinions of other users with similar tastes. FLAME improves existing aspect-based opinion mining methods by being able to infer aspect ratings of users on new items[2], and enhances collaborative filtering methods by leveraging reviews to analyze users' preferences on different aspects of items.

We empirically evaluate the proposed FLAME on a hotel review data set from TripAdvisor[3] and a restaurant review data set from Yelp[4]. Experimental results show that FLAME can effectively extract meaningful aspects and predict aspect ratings of a user on new items to him.

The remainder of the paper is organized as follows. Section 2 is devoted to related work. Section 3 introduces the problem definition and useful notations. Section 4 presents the proposed model and describes the inference and parameters estimation techniques. In Section 5 & 6, we report the experimental results on two review data sets and discuss some other applications of the proposed model. Finally, Section 7 concludes the paper with a summary and discusses potential future work.

2. RELATED WORK

Our work is related to two research topics: Collaborative Filtering and Aspect-based Opinion Mining.

2.1 Collaborative Filtering

Collaborative filtering (CF) is a popular method widely used in recommender systems. The assumption behind collaborative filtering is that a given user is more likely to like items that are liked by other users with similar tastes. Various state-of-the-art CF methods are based on latent factor models [7]. Latent factor models assume that a user's rating on a particular item depends on the inner dot product of the latent user factors and the latent item factors.

Some work combining collaborative filtering with *Topic Models* has been proposed to leverage text information in recommender systems. Topic models are introduced by [1] for learning the hidden dimensions of text. The basic assumption of topic models is that documents are represented by mixtures of some latent topics where topics are associated with a multinomial distribution over words of a vocabulary. The earliest work integrating collaborative filtering with topic model is CTM [21], which is proposed for article recommendation. CTM simultaneously trains a topic model on the collection of articles and a latent rating factor model on the ratings of users on articles, while assuming that the latent factors of items depend on the latent topic distributions of their text. A recent work [12] proposes a model called HFT, which aims at improving collaborative filtering using reviews. HFT considers latent rating factors of an item as the properties that the item possesses, and assumes that if a product exhibits a certain property (higher latent rating factor value), this will correspond to a particular topic being discussed frequently (higher probability in topic distribution) [12]. HTF first aggregates all the reviews of an item into a single document, and uses a similar method as CTM to train a topic model and a latent factor model together.

Different from CTM and HTF which learn topic distributions for each item, our approach learns for each review its aspect distribution *as well as* its rating distribution on each aspect.

---

[2] Note that *new items* for a user are the items that he has not rated yet.

[3] `http://www.tripadviosr.com`

[4] `http://www.yelp.com`

2.2 Aspect-based Opinion Mining

The main task of aspect-based opinion mining is extracting the aspects and learning the aspect ratings from a collection of reviews of a given item. Most of the early works of opinion mining are frequency-based approaches [5, 11]. These approaches usually mine the aspects and sentiments by counting the frequencies of words and their co-occurrences with some pre-defined seed words. Recently, several methods based on the variants of topic models [1] have been proposed [20, 25, 6, 14, 15] to learn the aspects and sentiments automatically from the data. These work extends topic models by adding another kind of latent variables to model the latent sentiments of words, i.e., words in reviews are not only dependent on the topics they belong to, but are also related to the sentiments of the reviewers. The most related work is the Latent Aspect Rating Analysis Model (LARAM) [22, 23], which aims at inferring the latent aspect ratings of given reviews. LARAM assumes the overall rating of a review is generated by a weighted sum of the latent aspect ratings, and are generated from the words and the latent topic allocations of the words by a linear regression function. LARAM learns the latent aspect ratings for each review and aspect weights for each reviewer. It should be noted that the aspect weights in LARAM are different from the personalized aspect ratings in our problem. The weights in LARAM represent the importance of the aspects for a reviewer, but personalized tastes represent the ratings/sentiments of users on different aspects. Two reviewers may share similar aspect weights but have totally different ratings on a given aspect.

The main limitation of above aspect-based opinion mining methods is that they do not consider user preferences (across multiple reviews and items) in the learning procedures so that they are unable to predict users' opinions on other items which they have not written reviews on.

The very recently published ETF [24] also considers aspect based opinion mining and collaborative filtering simultaneously. However, ETF employs the aspect-based opinion mining as a preprocessing step, while ours is a unified model with opinion mining as a part of the model. This enables our approach to be used to analyze the aspect distributions of the reviews and latent aspect ratings expressed in the reviews as in [22, 23]. Besides, ETF can not predict users' preferences on the aspects of items.

3. PROBLEM DEFINITION

We assume as input a collection of reviews of some products from a specific category (e.g. restaurant) by a group of reviewers, and each review comes with an overall rating (e.g. 1-5 stars) to express the overall satisfaction of the reviewer.

Review: A review is a piece of text describing opinions of a reviewer towards a specific item. Formally, we use $\mathcal{D} = \{d_1, d_2, ..., d_D\}$ to denote a set of review text documents. For each $d \in \mathcal{D}$, $u_d \in \mathcal{U}$ denotes the user who writes review d and $i_d \in \mathcal{I}$ denotes the reviewed item. We use $\mathcal{D}_u$ to denote the set of reviews that user u writes and use $\mathcal{D}_i$ to denote the set of reviews for item i.

Overall Rating: The overall rating r_d of a review document d is a numerical rating indicating the overall opinion of d, i.e., $r_d \in \mathcal{R}$, where $\mathcal{R} = \{1, 2, ..., R\}$.

Aspect: An aspect is an attribute of the item that has been commented on in a review, e.g., "food", "location" and "service" for a restaurant. In this paper, we only consider the case that all the items are from a same category, i.e., they share the same set of aspects. We use a to denote an aspect, where $a \in \mathcal{A}$ and $\mathcal{A} = \{1, 2, ..., A\}$.

Table 1: Mathematical Notations

Symbol	Size	Description
$\mathcal{U}$	U	Users $\mathcal{U} = \{u \vert u = 1, ..., U\}$
$\mathcal{I}$	I	Items $\mathcal{I} = \{i \vert i = 1, ..., I\}$
$\mathcal{D}$	D	Documents $\mathcal{D} = \{d \vert d = 1, ..., D\}$
$\mathcal{A}$	A	Aspects $\mathcal{A} = \{a \vert a = 1, ..., A\}$
$\mathcal{R}$	R	Numerical ratings $\mathcal{R} = \{r \vert r = 1, ..., R\}$
$\boldsymbol{\phi}_u$	$\mathbb{R}^K$	latent vector of user u
$\boldsymbol{\phi}_i$	$\mathbb{R}^K$	latent vector of item i
$\boldsymbol{\phi}_{i,a}$	$\mathbb{R}^K$	latent vector of aspect a of i
$\boldsymbol{\eta}$	$\mathbb{R}^A$	background aspect distribution
$\boldsymbol{\eta}_u$	$\mathbb{R}^A$	aspect distribution of user u
$\boldsymbol{\eta}_i$	$\mathbb{R}^A$	aspect distribution of item i
$\boldsymbol{\beta}_a$	$\mathbb{R}^V$	word distribution of aspect a
$\boldsymbol{\gamma}_{a,r}$	$\mathbb{R}^V$	word distribution of aspect a and rating r
$\boldsymbol{\theta}_d$	$\mathbb{R}^A$	aspect distribution of document d
$\boldsymbol{\varphi}_{d,a}$	$\mathbb{R}^R$	rating distribution of aspect a of document d
a_t	$\mathbb{R}^1$	aspect of sentence t
s_t	$\mathbb{R}^1$	aspect rating of sentence t

Aspect Rating: The aspect rating $r_{d,a}$ of a review document d is the reviewer u_d's rating towards to the aspect a of the item i_d.i It indicates the opinion of the reviewer regarding to the properties of the corresponding aspect of the item. Note that our method does not need aspect ratings as input, but instead it infers them from the data.

Personalized Latent Aspect Rating Analysis: Given a collection of reviews of a set of items by a set of users, the goal is to solve two tasks: *a*) learn the latent aspects, which represents each aspect as a distribution on a pre-defined vocabulary, and the latent aspect ratings for each review, which indicate the opinions of the reviewer towards the aspects of the item; *b*) predict the latent aspect ratings for user u on new item i that he has not reviewed.

Some important notations used in this paper are listed in Table 1. We use bold math symbols $\boldsymbol{x}_i$ to denote vectors, where the subscript i is used for indexing different vectors. The j-th element of the vector $\boldsymbol{x}_i$ is denoted by $\boldsymbol{x}_i[j]$.

4. PROPOSED MODEL

In this section, we propose the unified probabilistic model *Factorized Latent Aspect ModEl* (FLAME) to address the problem of Personalized Latent Aspect Rating Analysis.

When writing the review, the reviewer first selects a subset of aspects he wants to comment on. We assume that each review document d is associated with an aspect distribution $\boldsymbol{\theta}_d \in \mathbb{R}^A$, which represents the importances of the aspects in the review. The aspect distribution $\boldsymbol{\theta}_d$ depends on three factors: the global aspect distribution $\boldsymbol{\eta}_0$, the aspect distribution of the reviewer $\boldsymbol{\eta}_u$ and the aspect distribution of the item $\boldsymbol{\eta}_i$. $\boldsymbol{\eta}_0$ represents how much each aspect is likely to be mentioned among all the reviews. $\boldsymbol{\eta}_u$ represents reviewer u's preferences on the aspects to comment, e.g., if a user cares more on the *value* of a hotel, he prefers to mention this aspect in his reviews. $\boldsymbol{\eta}_i$ indicates which aspects of item i are more likely to be mentioned. Some aspects are more likely to be mentioned in the reviews of an item. For example, if the food of a restaurant is great, it will receive a lot of praises of the *food* in the reviews. On the other hand, if some aspects of an item are terrible, reviewers would like to criticize on these aspects in their reviews.

Based on this assumption, we define $\boldsymbol{\theta}_d$ using a additive generative methods as follows:

$$\boldsymbol{\theta}_d[a] = \frac{\exp\left(\boldsymbol{\eta}_0[a] + \boldsymbol{\eta}_u[a] + \boldsymbol{\eta}_i[a]\right)}{\sum_{a'=1}^{A} \exp\left(\boldsymbol{\eta}_0[a'] + \boldsymbol{\eta}_u[a'] + \boldsymbol{\eta}_i[a']\right)} \tag{1}$$

where $\{\boldsymbol{\eta}\} = \{\boldsymbol{\eta}_0, \boldsymbol{\eta}_u, \boldsymbol{\eta}_i | u \in \mathcal{U}, i \in \mathcal{I}\}$ are A-dimensional vectors generated from zero-mean Gaussian distributions.

$$\begin{aligned} \boldsymbol{\eta}_0 &\sim \mathcal{N}(\mathbf{0}, \sigma_\eta \boldsymbol{I}) \\ \boldsymbol{\eta}_u &\sim \mathcal{N}(\mathbf{0}, \sigma_\eta \boldsymbol{I}) \\ \boldsymbol{\eta}_i &\sim \mathcal{N}(\mathbf{0}, \sigma_\eta \boldsymbol{I}) \end{aligned} \tag{2}$$

For each aspect, the reviewer has a latent sentiment polarity expressing his opinion on that aspect of the item. We extend Probabilistic Matrix Factorization (PMF) [19] to model the user-specific aspect ratings. PMF assumes that the user u has a vector of latent factors $\boldsymbol{\phi}_u \in \mathbb{R}^K$, which represents his personalized preferences that influence his opinions. Analogously, each item has a latent vector $\boldsymbol{\phi}_i \in \mathbb{R}^K$. The overall rating of u for i is generated by the dot product of the user latent factor and the item latent factor. In our model, to predict user u's opinion on a specific aspect a of item i, we assume there is a latent factor $\boldsymbol{\phi}_{i,a} \in \mathbb{R}^K$ for each aspect a of an item i, and the aspect rating $r_{d,a}$ of review document d is generated from the dot product of the user latent vector $\boldsymbol{\phi}_u$ and the item aspect latent vector $\boldsymbol{\phi}_{i,a}$[5]. The item aspect latent vector $\boldsymbol{\phi}_{i,a}$ describes the latent properties of the corresponding aspect of the item.

$$r_{d,a} \sim \mathcal{N}(\boldsymbol{\phi}_u^\top \boldsymbol{\phi}_{i,a}, \sigma_a^2) \tag{3}$$

To control the model complexity, zero-mean Gaussian priors are placed on the latent factors:

$$\begin{aligned} \boldsymbol{\phi}_u &\sim \mathcal{N}(\mathbf{0}, \sigma_u^2 \boldsymbol{I}) \\ \boldsymbol{\phi}_{i,a} &\sim \mathcal{N}(\mathbf{0}, \sigma_{i,a}^2 \boldsymbol{I}) \end{aligned} \tag{4}$$

We can not directly use the continuous value $r_{d,a}$ to model the word generative process since we need discrete ratings to define the aspect-sentiment vocabulary (see Equation 11). We introduce another latent variable $\boldsymbol{\varphi}_{d,a} \in \mathbb{R}^R$ to represent document d's rating distribution on aspect a, where $\boldsymbol{\varphi}_{d,a}[r]$ is the probability of $p(r_{d,a} = r)$ and $\sum_{r=1}^{R} \boldsymbol{\varphi}_{d,a}[r] = 1$. We define $\boldsymbol{\varphi}_{d,a}$ as follows:

$$\begin{aligned} \boldsymbol{\varphi}_{d,a}[r] &= \frac{\mathcal{N}(r | \boldsymbol{\phi}_u^\top \boldsymbol{\phi}_{i,a}, \sigma_{r,a}^2)}{\sum_{r'=1}^{R} \mathcal{N}(r' | \boldsymbol{\phi}_u^\top \boldsymbol{\phi}_{i,a}, \sigma_{r,a}^2)} \\ &= \frac{\exp\left(-\frac{(r - \boldsymbol{\phi}_u^\top \boldsymbol{\phi}_{i,a})^2}{2\sigma_{r,a}^2}\right)}{\sum_{r'=1}^{R} \exp\left(-\frac{(r' - \boldsymbol{\phi}_u^\top \boldsymbol{\phi}_{i,a})^2}{2\sigma_{r,a}^2}\right)} \end{aligned} \tag{5}$$

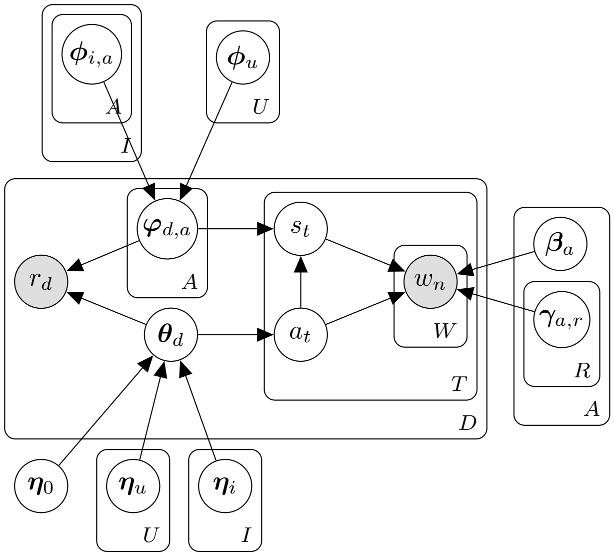


Figure 2: FLAME in graphical model notation.

We assume the overall rating of document d is generated from the weighted sum of the aspect ratings, where the aspect weights consist to the aspect distribution of the document.

$$r_d \sim \mathcal{N}(\sum_a \boldsymbol{\theta}_d[a] \mathbb{E}[r_{d,a}], \sigma_r^2) \tag{6}$$

where $\mathbb{E}[r_{d,a}] = \boldsymbol{\phi}_u^\top \boldsymbol{\phi}_{i,a}$.

For the process of generating words, we follow the assumption in [20, 13, 25] that the words in one sentence of a review refer to the same aspect. Topics learned under this assumption are *local* topics that preserve sentence-level word concurrences [20], while models like LDA [1] produce *global* topics that preserve document-level word concurrences. *Global* topic models are not suitable for aspect identification. For example, because the words *room* and *location* appear together in most reviews, *global* topic models are mostly likely to cluster them in the same topic, but *local* topic models assume they refer to different topics.

For each sentence t in the review d, we draw an aspect a_t from the aspect distribution of the review:

$$a_t \sim Multi(\boldsymbol{\theta}_d) \tag{7}$$

and a sentiment rating $s_t \in \{1, 2, ..., R\}$ on the aspect a_t from the aspect rating distribution:

$$s_t \sim Multi(\boldsymbol{\varphi}_{d,a_t}) \tag{8}$$

Then, in each sentence t, the reviewer selects a set of words $w_n \in t$ to express his opinions on the aspect a_t. We define an aspect-sentiment multinomial word distribution $\boldsymbol{\alpha}_{a,s}$ on the vocabulary, where $\boldsymbol{\alpha}_{a,s}[j]$ represents the probability of generating the j-th word from the vocabulary for aspect a and aspect rating s. w_n can be an aspect word, e.g., *battery*, or a sentiment word, e.g., *good*. So we assume that $\boldsymbol{\alpha}_{a,s}$ depends on two factors: $\boldsymbol{\beta}_a$ and $\boldsymbol{\gamma}_{a,s}$, where $\boldsymbol{\beta}_a$ represents the correlation between the words and the aspect a, and $\boldsymbol{\gamma}_{a,s}$ represents the correlation between the words and the pair of

---

[5] Note that for simplicity we always assume that there is an extra constant column in user/item latent factors to model the bias effect.

Algorithm 1 Generative process of FLAME.

```
Draw $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \sigma_\eta^2 \boldsymbol{I})$
for all $u \in \mathcal{U}$ do
    Draw $\boldsymbol{\phi}_u \sim \mathcal{N}(\mathbf{0}, \sigma_u^2 \boldsymbol{I})$
    Draw $\boldsymbol{\eta}_u \sim \mathcal{N}(\mathbf{0}, \sigma_\eta^2 \boldsymbol{I})$
end for
for all $i \in \mathcal{I}$ do
    Draw $\boldsymbol{\eta}_i \sim \mathcal{N}(\mathbf{0}, \sigma_\eta^2 \boldsymbol{I})$
    for all $a \in \mathcal{A}$ do
        Draw $\boldsymbol{\phi}_{i,a} \sim \mathcal{N}(\mathbf{0}, \sigma_{i,a}^2 \boldsymbol{I})$
    end for
end for
for all $a \in \mathcal{A}$ do
    Draw $\boldsymbol{\beta}_a \sim \mathcal{N}(\mathbf{0}, \sigma_\beta^2 \boldsymbol{I})$
    for all $r \in \mathcal{R}$ do
        Draw $\boldsymbol{\gamma}_{a,r} \sim \mathcal{N}(\mathbf{0}, \sigma_\gamma^2 \boldsymbol{I})$
        Set $\boldsymbol{\alpha}_{a,r}$ using Equation (9)
    end for
end for
for all $d \in \mathcal{D}$ do
    Set $\boldsymbol{\theta}_d$ using Equation (1)
    Set $\boldsymbol{\varphi}_d$ using Equation (5)
    Draw $r_d$ using Equation (6)
    // Generate each sentence $t$ in document $d$
    for all $t \in d$ do
        Draw $a_t \sim \text{Multi}(\boldsymbol{\theta}_d)$
        Draw $s_t \sim \text{Multi}(\boldsymbol{\varphi}_d, \boldsymbol{a}_t)$
        // Generate each word $n$ in sentence $t$
        for all $n \in t$ do
            Draw $w_n \sim \text{Multi}(\boldsymbol{\alpha}_{a_t,s_t})$
        end for
    end for
end for
```

aspect a and aspect rating s.

$$\boldsymbol{\alpha}_{a,s}[j] = \frac{\exp(\boldsymbol{\beta}_a[j] + \boldsymbol{\gamma}_{a,s}[j])}{\sum_{l=1}^{V} \exp(\boldsymbol{\beta}_a[l] + \boldsymbol{\gamma}_{a,s}[l])} \tag{9}$$

where $\boldsymbol{\beta}_a$ and $\boldsymbol{\gamma}_{a,r}$ are V-dimensional vectors generated from zero-mean Gaussian distributions.

$$\begin{aligned} \boldsymbol{\beta}_a &\sim \mathcal{N}(\mathbf{0}, \sigma_\beta \boldsymbol{I}) \\ \boldsymbol{\gamma}_{a,r} &\sim \mathcal{N}(\mathbf{0}, \sigma_\gamma \boldsymbol{I}) \end{aligned} \tag{10}$$

The word w_n is generated as follows:

$$p(w_n | a_t, s_t, \boldsymbol{\alpha}) \sim Multi(\boldsymbol{\alpha}_{a_t,s_t}) \tag{11}$$

Figure 2 shows the graphical representation of FLAME, omitting the priors $\{\sigma\}$. We summarize the generative process in Algorithm 1.

4.1 Variational Inference

In general, we use an EM-style method to learn the parameters in our model. We adopt a mixture of maximum a posteriori (MAP) point estimates and Bayesian inference as in [3]. To be specific, we use a combination of MAP estimation over $\boldsymbol{\Xi} = \{\{\boldsymbol{\eta}\}, \{\boldsymbol{\phi}\}, \boldsymbol{\beta}, \boldsymbol{\gamma}\}$ and Bayesian variational inference over the other latent variables $\boldsymbol{\Delta} = \{\boldsymbol{a}, \boldsymbol{s}\}$ to derive a lower bound of the log likelihood of the data, and maximize the bound with respect to $\boldsymbol{\Xi}$ and variational parameters. It should be noted that $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ are not latent variables in our model. They are fixed given $\boldsymbol{\eta}$ and $\boldsymbol{\phi}$, as shown in Equation (1) and (5).

The variational distributions of the latent variables in $\boldsymbol{\Delta}$ are defined as follows:

$$q(\boldsymbol{a}, \boldsymbol{s} | \boldsymbol{\pi}, \boldsymbol{\lambda}) = \prod_d \prod_{t \in d} q(a_t | \boldsymbol{\pi}_t) q(s_t | \boldsymbol{\lambda}_t) \tag{12}$$

where $\boldsymbol{\pi}_t \in \mathbb{R}^A$ and $\boldsymbol{\lambda}_t \in \mathbb{R}^R$ are free multinomial parameters.

We get the lower bound of the log likelihood of the data as follows:

$$\begin{aligned} \mathcal{L} = & \sum_d \Bigg(\langle \log p(r_d | \boldsymbol{\phi}_u, \boldsymbol{\phi}_{i,a}, \boldsymbol{\theta}_d) \rangle + \sum_{t \in d} \Big(\langle \log p(a_t | \boldsymbol{\theta}_d) \rangle \\ & + \langle \log p(s_t | \boldsymbol{\varphi}_d, a_t) \rangle + \sum_{n \in t} \langle \log p(w_n | a_t, s_t, \boldsymbol{\beta}, \gamma) \rangle \Big) \Bigg) \\ & + \sum_u \Big(\langle \log p(\boldsymbol{\phi}_u | \sigma_u) \rangle + \langle \log p(\boldsymbol{\eta}_u | \sigma_\eta) \rangle \Big) \\ & + \sum_i \Big(\langle \log p(\boldsymbol{\phi}_i | \sigma_i) \rangle + \sum_a \langle \log p(\boldsymbol{\phi}_{i,a} | \sigma_{i,a}) \rangle \\ & + \langle \log p(\boldsymbol{\eta}_i | \sigma_\eta) \rangle \Big) + \langle \log p(\boldsymbol{\eta} | \sigma_\eta) \rangle \\ & + \sum_a \langle \log p(\boldsymbol{\beta}_a | \sigma_\beta) \rangle + \sum_a \sum_r \langle \log p(\boldsymbol{\gamma}_{a,r} | \sigma_\gamma) \rangle \\ & - \sum_d \sum_{t \in d} \Big(\langle \log q(\boldsymbol{a}_t | \boldsymbol{\pi}_t) \rangle + \langle \log q(\boldsymbol{s}_t | \boldsymbol{\lambda}_t) \rangle \Big) \end{aligned} \tag{13}$$

where $\langle p(\boldsymbol{\Omega}) \rangle$ denotes the expectation of the probability of p given the distribution $q(\boldsymbol{\Omega})$.

4.2 Learning the Parameters

In general, the learning procedure can be viewed as coordinate ascent in $\mathcal{L}$, i.e., alternatively optimizing one set of parameters while fixing the others.

Updating $\boldsymbol{\pi}$: We get the solution of $\boldsymbol{\pi}_t$ by setting $\frac{\partial \mathcal{L}_{[\boldsymbol{\pi}_t]}}{\partial \boldsymbol{\pi}_t} = 0$ with the constraint $\sum_a \boldsymbol{\pi}_t[a] = 1$.

$$\boldsymbol{\pi}_t[a] \propto \boldsymbol{\theta}_d[a] \prod_r \left(\boldsymbol{\varphi}_{d,a}[r]^{\boldsymbol{\lambda}_t[r]} \prod_j \boldsymbol{\alpha}_{a,r}[j]^{c_{t,j} \boldsymbol{\lambda}_t[r]} \right) \tag{14}$$

where $c_{t,j}$ is the frequency of the j-th word in sentence t. Since $c_{t,j}$ is sparse, the complexity of updating $\boldsymbol{\pi}_t$ is $O(c_t \cdot R \cdot A)$, where c_t is the number of words in sentence t. The total complexity for updating $\boldsymbol{\pi}$ in one EM iteration is $O(c \cdot R \cdot A)$, where c is the number of words of all the documents.

Updating $\boldsymbol{\lambda}$: The update procedure of $\boldsymbol{\lambda}$ is similar to that for $\boldsymbol{\pi}$.

$$\boldsymbol{\lambda}_t[r] \propto \prod_a \boldsymbol{\varphi}_{d,a}[r]^{\boldsymbol{\pi}_t[a]} \prod_j \boldsymbol{\alpha}_{a,r}[j]^{c_{t,j} \boldsymbol{\pi}_t[a]} \tag{15}$$

The complexity of updating $\boldsymbol{\lambda}$ in one EM iteration is also $O(c \cdot A \cdot R)$.

Updating $\boldsymbol{\phi}_u$: We can get $\mathcal{L}_{[\boldsymbol{\phi}_u]}$ by only retaining those terms in $\mathcal{L}$ that are a function of $\boldsymbol{\phi}_u$:

$$\begin{aligned} \mathcal{L}_{[\boldsymbol{\phi}_u]} = & \sum_{d \in \mathcal{D}_u} \Bigg(- \frac{(r_d - \sum_a \boldsymbol{\theta}_d[a] \boldsymbol{\phi}_u^\top \boldsymbol{\phi}_{i,a})^2}{2\sigma_r^2} \\ & + \sum_t \sum_a \sum_r \boldsymbol{\pi}_t[a] \boldsymbol{\lambda}_t[r] \log \boldsymbol{\varphi}_{d,a}[r] \Bigg) - \frac{\boldsymbol{\phi}_u^\top \boldsymbol{\phi}_u}{2\sigma_u^2} \end{aligned} \tag{16}$$

The derivative of $\mathcal{L}_{[\boldsymbol{\phi}_u]}$ with respect to $\boldsymbol{\phi}_u$ depends on $\boldsymbol{\phi}_u$, so we have to use a gradient ascent based method to update $\boldsymbol{\phi}_u$.

Updating $\boldsymbol{\phi}_{i,a}$:

$$\begin{aligned}\mathcal{L}_{[\boldsymbol{\phi}_{i,a}]} = \sum_{d \in \mathcal{D}_i} \Bigg(&- \frac{(r_d - \sum_a \boldsymbol{\theta}_d[a] \boldsymbol{\phi}_u^\top \boldsymbol{\phi}_{i,a})^2}{2\sigma_r^2} \\ &+ \sum_{t \in d} \sum_a \sum_r \boldsymbol{\pi}_t[a] \boldsymbol{\lambda}_t[r] \log \boldsymbol{\varphi}_{d,a}[r] \Bigg) \\ &- \frac{\boldsymbol{\phi}_{i,a}^\top \boldsymbol{\phi}_{i,a}}{2\sigma_{i,a}^2}\end{aligned} \tag{17}$$

We also use a gradient ascent based method to update $\boldsymbol{\phi}_{i,a}$.

Updating $\boldsymbol{\eta}$:

$$\begin{aligned}\mathcal{L}_{[\boldsymbol{\eta}_0]} =& \sum_{d \in D} \big(- \frac{(r_d - \sum_a \boldsymbol{\theta}_d[a] \mathbb{E}[r_{d,a}])^2}{2\sigma_r^2} \\ &+ \sum_{t \in d} \sum_a \boldsymbol{\pi}_t[a] \log \boldsymbol{\theta}_d[a] \big) - \frac{\boldsymbol{\eta}_0^\top \boldsymbol{\eta}_0}{2\sigma_\eta^2} \\ =& \boldsymbol{\pi}_D^\top \boldsymbol{\eta}_0 - \frac{\boldsymbol{\eta}_0^\top \boldsymbol{\eta}_0}{2\sigma_\eta^2} - \sum_{d \in D} \frac{(r_d - \sum_a \boldsymbol{\theta}_d[a] \mathbb{E}[r_{d,a}])^2}{2\sigma_r^2} \\ &- \sum_{d \in D} N_d \log \Bigg(\sum_{a'} \exp \big(\boldsymbol{\eta}_0[a'] + \boldsymbol{\eta}_u[a'] + \boldsymbol{\eta}_i[a'] \big) \Bigg)\end{aligned} \tag{18}$$

where $\boldsymbol{\pi}_D = \sum_{d \in D} \sum_{t \in d} \boldsymbol{\pi}_t$ and $N_d = \sum_{t \in d} \sum_a \boldsymbol{\pi}_t[a] = \sum_{t \in d} 1$ is the number of sentences in d.

We apply gradient ascent method to optimize $\boldsymbol{\eta}_0$. The derivative with respect to $\boldsymbol{\eta}_0[a]$ is :

$$\begin{aligned}g(\boldsymbol{\eta}_0[a]) =& \boldsymbol{\pi}_D[a] - \sum_{d \in D} N_d \boldsymbol{\theta}_d[a] - \frac{\boldsymbol{\eta}_0[a]}{\sigma_\eta^2} \\ &+ \sum_{d \in D} \frac{(r_d - \sum_a \boldsymbol{\theta}_d[a] \mathbb{E}[r_{d,a}])(\boldsymbol{\theta}_d[a])(1 - \boldsymbol{\theta}_d[a])}{\sigma_r^2}\end{aligned} \tag{19}$$

The update formula of $\boldsymbol{\eta}_u$ and $\boldsymbol{\eta}_i$ is similar. The only difference is to replace $\mathcal{D}$ in Equation (19) with $\mathcal{D}_u$ and $\mathcal{D}_i$ respectively, where $\mathcal{D}_u$ is the set of reviews of user u and $\mathcal{D}_i$ is the set of reviews of item i.

Updating $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$:

$$\begin{aligned}\mathcal{L}_{[\boldsymbol{\beta}_a]} =& \boldsymbol{c}_a^\top \boldsymbol{\beta}_a - \frac{\boldsymbol{\beta}_a^\top \boldsymbol{\beta}_a}{2\sigma_\beta^2} \\ &- \sum_d \sum_{t \in d} \boldsymbol{\pi}_t[a] c_t \sum_r \boldsymbol{\lambda}_t[r] \log \Bigg(\sum_l \exp \left(\boldsymbol{\beta}_a[l] + \boldsymbol{\gamma}_{a,r}[l] \right) \Bigg)\end{aligned} \tag{20}$$

where $\boldsymbol{c}_a[j] = \sum_d \sum_{t \in d} \boldsymbol{\pi}_t[a] c_{t,j}$, and $c_t = \sum_j c_{t,j}$ denotes the number of words in sentence t.

We use Newton's method to optimize $\boldsymbol{\beta}_a$. The derivative with respect to $\boldsymbol{\beta}_a$ is :

$$g(\boldsymbol{\beta}_a) = \boldsymbol{c}_a - \sum_r C_{a,r} \boldsymbol{\alpha}_{a,r} - \frac{\boldsymbol{\beta}_a}{\sigma_\beta^2} \tag{21}$$

where $C_{a,r} = \sum_d \sum_{t \in d} \boldsymbol{\pi}_t[a] c_t \boldsymbol{\lambda}_t[r]$ represents the expected word counts for each (a, r) combination. The Hessian matrix is:

$$H(\boldsymbol{\beta}_a) = \sum_r C_{a,r} \boldsymbol{\alpha}_{a,r} \boldsymbol{\alpha}_{a,r}^\top - diag(\sum_r C_{a,r} \boldsymbol{\alpha}_{a,r} + \frac{1}{\sigma_\beta} \mathbf{1}) \tag{22}$$

The update formula for $\boldsymbol{\beta}_a$ is:

$$\boldsymbol{\beta}_a^{(t+1)} = \boldsymbol{\beta}_a^{(t)} - \boldsymbol{H}^{-1}(\boldsymbol{\beta}_a^{(t)}) g(\boldsymbol{\beta}_a^{(t)}) \tag{23}$$

We use a linear algorithm for the Hessian matrices with special structure [18, 1, 3], which lets the complexity of computing $\boldsymbol{H}^{-1}(\boldsymbol{\beta}_a) g(\boldsymbol{\beta}_a)$ be $O(V)$ instead of $O(V^3)$.

We can also get the derivative and Hessian of $\boldsymbol{\gamma}_{a,r}$ as follows:

$$g(\boldsymbol{\gamma}_{a,r}) = \boldsymbol{c}_{a,r} - C_{a,r} \boldsymbol{\alpha}_{a,r} - \frac{\boldsymbol{\gamma}_{a,r}}{\sigma_\gamma^2} \tag{24}$$

where $\boldsymbol{c}_{a,r}[j] = \sum_d \sum_{t \in d} \boldsymbol{\pi}_t[a] \boldsymbol{\lambda}_t[r] c_{t,j}$.

$$H(\boldsymbol{\gamma}_{a,r}) = C_{a,r} \boldsymbol{\alpha}_{a,r} \boldsymbol{\alpha}_{a,r}^\top - diag(C_{a,r} \boldsymbol{\alpha}_{a,r} + \frac{1}{\sigma_\gamma} \mathbf{1}) \tag{25}$$

The complexity of updating $\boldsymbol{\gamma}_{a,r}$ is also linear in the size of the vocabulary.

Computational Complexity: To conclude, the complexity of one update iteration is $O(c \cdot A \cdot R + T \cdot A \cdot K + D \cdot K + (I + U) \cdot A + A \cdot R \cdot V)$, where c is the total number of words in the corpus, T is the number of sentences in the corpus, and D is the number of documents in the corpus. Usually K, A and R are small constants, so the complexity is linear to the size of the review dataset.

Implementation Notes: An important issue is how to initialize the model. We use the following initialization steps. Taking the TripAdvisor data set as an example, we initialize $\boldsymbol{\beta}_a$ using the names of the aspect, i.e., we set $\boldsymbol{\beta}_{room,room} = 1$ for the aspect *room*, and then learn the aspect distribution of each sentence only based on the initialized $\boldsymbol{\beta}$. Similar techniques are also used in [13]. The aspect ratings of each sentence are initialized using the overall rating of the review. The parameters $\{\sigma\}$ can also be learned using the coordinate ascent-like procedure. We set them manually in our implementation, e.g., we set $\sigma_r^2 = 1$, $\sigma_{r,a}^2 = 0.5$, $\sigma_\eta = 10$, etc. Some optimization techniques, e.g., L-BFGS [17] and backtracking line search [2], are applied to accelerate the gradient ascent updates.

5. EXPERIMENTAL EVALUATION

In this section, we first describe the data sets we used in our experiments and then discuss the experimental results on different tasks.

5.1 Data Sets and Preprocessing

We use two review data sets for our experimental evaluation: the TripAdvisor *hotel* review data[6] and Yelp review data[7]. In the TripAdvisor data, besides the overall rating, users are also asked to provide the aspect ratings on 6 pre-defined aspects: *Location*, *Sleep Quality*, *Room*, *Service*, *Value* and *Cleanliness*, on a scale from 1 star to 5 stars. We use these ground-truth aspect ratings to evaluate our model on the task of aspect rating prediction. For Yelp data set, we

---

[6] `http://www.cs.cmu.edu/~jiweil/html/hotel-review.html`

[7] `http://www.yelp.com/dataset_challenge`

Table 2: Dataset Statistics

	TripAdvisor	Yelp
# Users	9,419	6,944
# Items	1,904	3,315
# Reviews	66,637	115,290
Density	0.37%	0.50%
# Sentences Per Review	12.60 ± 8.64	11.67 ± 7.80
# Words Per Sentence	7.50 ± 3.76	6.47 ± 4.64

extract a subset which only contains the reviews on *restaurants*.

We use the following preprocessing procedure on both of the data sets. We first remove non-English reviews and reviews with less than 3 sentences or 20 words, and then iteratively remove users with less than 5 reviews and items with less than 5 reviews. For the text in reviews, we remove stop words and words that occur in less than 5 reviews, and stem the remaining words using the PorterStemmer[8]. After the preprocessing, we have a *hotel* review data set including 66,637 hotel reviews of 1,904 hotels and a *restaurant* review data set including 115,290 reviews of 3,315 restaurants. The detailed statistics are listed in Table 2.

We randomly split both of the data sets into training and test sets. Specifically, for each user, we randomly select 20% of his reviews as test examples (For users with less than 10 reviews, we randomly select 2 reviews as test examples) and put the rest reviews into the training sets. We train the models on the training data sets and test their performance on the test data sets. We use the model initialization method and parameters selection strategies as discussed in Section 4.

5.2 Quantitative Evaluation

5.2.1 Perplexity on Held-out Reviews

As in standard topic models, we use *perplexity* of the held-out test data sets to compare the generalization performance of FLAME with some other state-of-the-art models.

Evaluation Measure. *Perplexity* is a standard measure for topic models to measure how the model can generate future documents [1]. For review mining, a good aspect-based topic model should be able to predict what the reviewer will write in a new review, which leads to a lower perplexity. A strong correlation of the perplexity and accuracy of aspect-based topic models is shown in [15]. We use a lower bound on perplexity as in [4].

$$\text{Perplexity}(\mathcal{D}_{test}) = \exp\left(-\frac{\sum_d \langle \log p(\boldsymbol{w}_d|\boldsymbol{\Xi})\rangle - \langle p(\boldsymbol{\Delta}_d)\rangle}{\sum_d N_d}\right)$$

We compare FLAME with the basic LDA model [1] and the D-LDA model presented in [15]. D-LDA is a state-of-the-art aspect-based opinion mining model which can be seen as a generalization of several other models [20, 6]. For D-LDA, we also use the assumption that the words in one sentence refer to the same aspect as in FLAME and other models [15, 20, 6]. In the aspect-based topic models, we actually use $A \times R$ latent topics, so we compare with LDA using both A topics (LDA-A) and $A \times R$ topics (LDA-AR).

Table 3: Perplexity on the held-out data sets

	TripAdvisor	Yelp
LDA-A	1012.80	767.24
LDA-AR	918.07	728.00
D-LDA	771.05	621.24
FLAME	**733.12**	**590.46**

For all the models, we use the same parameter settings and stopping criteria. We set $R = 5$ for all the aspect-based topic models. We train the models using the reviews in the training sets and evaluate the perplexity on the test sets. We test various numbers of latent aspects $A = \{6, 12, 24\}$. Since the relative results are similar, we choose $A = 6$ for discussion. Table 3 shows the perplexity on test data sets of FLAME and the comparison partners. We can see that D-LDA and FLAME, which are specifically designed for aspect-based opinion mining, significantly outperform basic LDA methods. FLAME achieves the best results among all the models on both of the data sets. We believe this is because FLAME can predict personalized aspect distribution as well as aspect rating distribution, which other models do not consider.

5.2.2 Aspect Rating Prediction on Held-out Reviews

Since we need the ground-truth aspect ratings to quantitatively compare FLAME with other methods, we evaluate the aspect rating prediction only on the TripAdvisor data set. In order to align the latent aspects to the pre-defined aspects in the TripAdvisor data set, we set A to be 6 and use the initialization techniques discussed in Section 4.

We use the evaluation measures in [22, 23] to evaluate different methods:

- Root Mean Square Error (RMSE) of the predicted aspect ratings compared with the ground-truth aspect ratings.

- Pearson Correlation inside reviews (ρ_A) to measure how well the predicted aspect ratings preserve the relative order of aspects within a review.

  $$\rho_A = \frac{1}{D}\sum_{d=1}^{D} \rho(\boldsymbol{s}_d, \boldsymbol{s}_d^*)$$

  where $\boldsymbol{s}_d^*$ is the ground-truth aspect ratings for document d, and $\boldsymbol{s}_d$ is predicted aspect ratings.

- Pearson Correlation between personalized ranking of items ρ_I. For each user and each aspect, we rank the items by their predicted aspect ratings, and measure how the ranked lists preserve the ground truth.

  $$\rho_I = \frac{1}{U \cdot A}\sum_{u=1}^{U}\sum_{a=1}^{A} \rho(\boldsymbol{s}_{\mathcal{I}_u,a}, \boldsymbol{s}_{\mathcal{I}_u,a}^*)$$

  where $\mathcal{I}_u$ is the set of items in user u's test data, $\boldsymbol{s}_{\mathcal{I}_u,a}$ is the predicted aspect ratings on the set of items and $\boldsymbol{s}_{\mathcal{I}_u,a}$ is the ground-truth ratings.

- Zero-One Ranking loss ($L_{0/1}$) [8], which measures the percentage of mis-ordered pairs of items for each user.

---

[8] `http://tartarus.org/martin/PorterStemmer/`

Table 4: Aspect rating prediction on test set of TripAdvisor data

	PMF	LRR+PMF	FLAME
RMSE	**0.970**	1.000	0.980
ρ_A	N/A	0.110	**0.195**
ρ_I	0.304	0.177	**0.333**
$L_{0/1}$	0.210	0.238	**0.196**

It is computed as follows:

$$\sum_{u} \sum_{i,j \in \mathcal{I}_u} \frac{1}{Z_u} \sum_{a=1}^{A} \mathbf{1}[(\boldsymbol{s}_{u,i,a} - \boldsymbol{s}_{u,j,a}) \cdot (\boldsymbol{s}^*_{u,i,a} - \boldsymbol{s}^*_{u,j,a}) < 0]$$

where Z_u is the number of pairs in user u's test set, $\boldsymbol{s}_{u,i,a}$ is the predicted rating of user u of item i on aspect a, and $\boldsymbol{s}^*_{u,i,a}$ is the ground-truth aspect rating. We do not choose nDCG since each user has few samples in the test data (2.2 test samples per user), the values of nDCG tend to be very close to 1 for all comparison partners.

An intuitive solution of aspect rating prediction is just using the overall rating of the review as prediction. We use PMF [19] to predict the overall ratings of the reviews in the test set and use the predicted overall ratings as predictions for aspect ratings. To our best knowledge, [22, 23] are the only work that predict aspect ratings at review-level. However, they can only predict aspect ratings based on users' reviews. In order to predict the aspect ratings in test set, we first apply the LRR model [22] to extract aspect ratings for each review in the training set, and then use PMF [19] to train and to predict the aspect ratings in test set (we call it LRR+PMF). We use the code of LRR provided by the authors. Same training and testing strategies are applied to all the models for fair comparison. We also test the performance of with different values of the dimensions of latent factors. The relative results are similar, so we only choose $K = 10$ for discussion. Table 4 presents the results for aspect rating prediction on the test set. We also highlight the best performance in each measure in bold.

A general observation is that FLAME outperforms other baseline models on all the measures except RMSE. It has been discussed in [22] that RMSE is less important than other measures since it does not reflect how the relative order of the aspect ratings is preserved. ρ_A measures how a method preserves the relative order of aspect ratings within a review. PMF uses the same predicted ratings for all aspects, so it is not applicable for ρ_A. ρ_I and $L_{0/1}$ are the most important measures for our task where we want to rank items based on the predicted aspect ratings. PMF outputs exactly the same item ranking lists for all aspects, thus it is not suitable for real-world applications. We can see that FLAME gets the best results on the two measures. The gain is especially significant compared to LRR+PMF, where there are about 90% improvement on ρ_I and 40% improvement on $L_{0/1}$.

Note that LRR+PMF does not achieve desirable performance. The reason is that it is a two-step approach that the errors induced in the first step have significant influence on the performance of the second step.

5.3 Qualitative Evaluation

In this subsection, we evaluate FLAME on the task of aspect identification. We perform qualitative analysis of the top words obtained by FLAME to see whether FLAME can produce meaningful aspects.

Figure 3 shows the word-cloud visualization of top words (after stemming) with the highest generating probability in the aspect-rating specific word distributions. We only show 3 aspects due to space limits. The three word-cloud figures in the left column present the topic distribution $\boldsymbol{\beta}$ for the aspects *location*, *service* and *room*, respectively. In general, the top words generated by FLAME represent meaningful and interpretable topics. We observe that the top words match our intuition, e.g., words like "location", "walk", "street" have higher weights in the word distribution of aspect *location*. The middle and right columns show the top words of the 2-star ($\boldsymbol{\gamma}_{a,2}$) and 5-star ($\boldsymbol{\gamma}_{a,5}$) word distributions of for the three aspects,. The aspect-rating specific word distribution can automatically learn the sentiment oriented words, e.g., words like "bad", "old", "creepy" and "homeless" have high weights in the 2-star word distribution of the aspect *location*, while the words like "view", "great", "perfect", "best" have high weights in the 5-star word distribution of *location*.

One contribution of FLAME is that the aspect-rating topics have sentiment polarities, i.e., 5-star topics are more positive than 4-star, and so on. This is different from previous work [20, 14, 16] where the latent ratings in these models are rating labels which do not correspond to sentiment polarities.

6. FURTHER APPLICATIONS

The detailed analysis on personalized latent aspect ratings enables a wide range of applications. Here we discuss three sample applications.

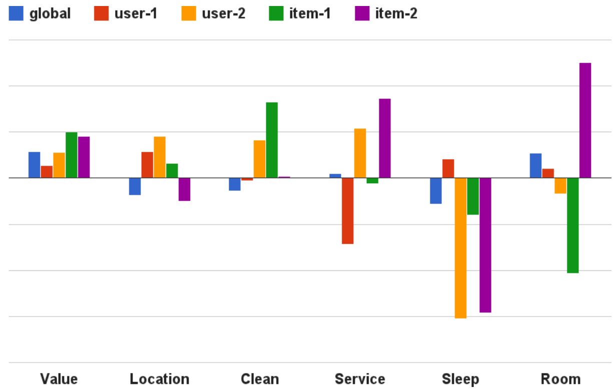
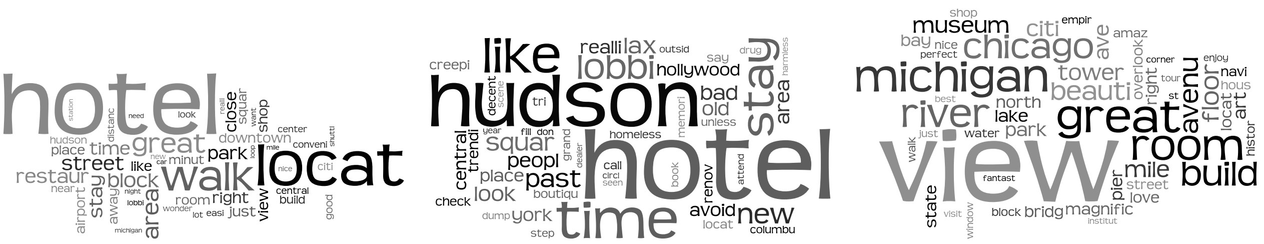
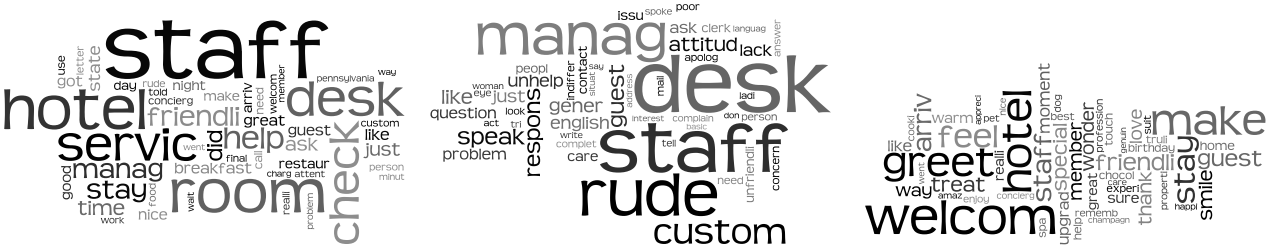


Figure 4: Aspect Weights. *Global* **represents the values of $\boldsymbol{\eta}_0$.** *user-1* **and** *user-2* **are the aspect weights $\boldsymbol{\eta}_u$ of two randomly sampled users, and** *item-1* **and** *item-2* **are the values of $\boldsymbol{\eta}_i$ for two randomly sampled items.**

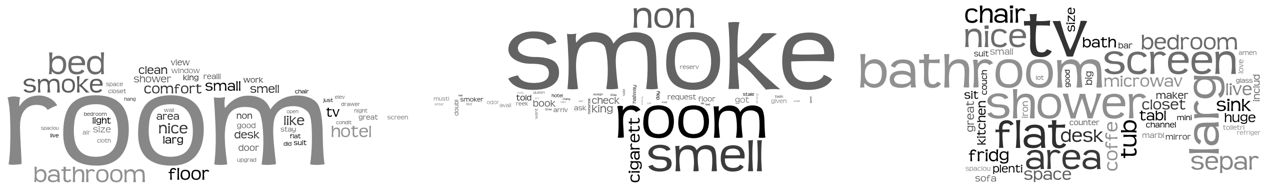
Aspect Distribution: Since FLAME can infer the aspect weights for users and item, we can easily use the values of $\boldsymbol{\eta}_0$, $\boldsymbol{\eta}_u$ and $\boldsymbol{\eta}_i$ for the rating behavior analysis. Figure 4 shows some sample results on TripAdviosr data. From the histogram *Global* in the figure, we can see that *Value* and *Room* are the most discussed aspects, and most people rarely mention the aspect *Sleep*. Note that the values of $\boldsymbol{\eta}_u$



(a) Location (b) Location 2-star (c) Location 5-star



(d) Service (e) Service 2-star (f) Service 5-star



(g) Room (h) Room 2-star (i) Room 5-star

Figure 3: Word-cloud visualization of top words with highest generating probability in β and γ. The left Word size reflects the weight of each word. The three rows are the top words in the topic distributions for the aspects *location*, *service* and *room*, respectively. The left column shows the top words in the aspect-word distributions β of the three aspects. The middle and right columns show the top words in the aspect-rating-word distributions γ. The middle column shows negative ones (2-star) and the right column shows positive ones (5-star).

indicates the biases of users deviating from the global aspect weights. We can see that *user-1* likes to comment on the *Location* and *Sleep*, while *user-2* cares more about the *Service*, *Clean* and *Location*. The two users have opposite weights for the aspects *Service* and *Sleep*. Thus, when they are searching for hotels, the aspects they care about are different. It indicates that letting users choose to rank items based on aspects which they cares about is very useful. The aspect weights of items can be used to help merchants to improve their services. If a specific aspect is discussed a lot and most of the reviews are negative, the merchant should think about how to improve this aspect.

Personalized Review Recommendation: As discussed in Section 1, facing a large number of reviews expressing different opinions, a user might have no idea of which reviews are reliable. FLAME can alleviate this problem by sorting the reviews by the similarities between reviewers with current user. A simple way of computing the similarities between users is to compute the distance between their latent factors. Since personalized review recommendation is hard to evaluate, we would like to leave it as a future work on some data sets with ground-truth of user feedback on reviews.

Recommendation Explanation: Traditional collaborative filtering methods only provide predicted scores for then items, but can not produce reasonable explanations with the recommendations. A recent work [24] has shown the possibility of using the aspect weights to generate some explanations. FLAME can produce more persuasive recommendation explanations by the predicted aspect ratings and some selected reviews written by similar users.

7. CONCLUSION

In this paper, we introduce the problem of Personalized Latent Aspect Rating Analysis to model users' preferences on different aspects. We propose a unified probabilistic model FLAME which combines aspect-based opinion mining and collaborative filtering. FLAME extracts aspect ratings from the review text, and predicts aspect ratings of an item that a user has not yet reviewed based on aspect ratings within other reviews of the item by other users with similar preferences. Our experimental evaluation on a hotel review data set and a restaurant review data set shows that FLAME can effectively solve the research problem. The qualitative evaluation shows that FLAME can automatically extract meaningful aspects and sentiment-oriented aspects. We also investigate the ability of FLAME on the task of generating future review text. Most importantly, our experiments on TripAdvisor data sets show that FLAME significantly outperforms state-of-the-art methods in terms of accuracy of aspect rating prediction.

Some websites like TripAdvisor provide the option of rating some pre-defined aspects. Although these aspect ratings are typically incomplete, they may be helpful to partially guide the learning of latent aspect ratings. It is worth to explore a semi-supervised extension of FLAME. In this paper, we only consider one same type of items, e.g., hotels or restaurants. We plan to consider datasets with multiple types of items which have different aspects, where users may also have different preferences on different types of items.

8. REFERENCES

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, Mar. 2003.

[2] S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[3] J. Eisenstein, A. Ahmed, and E. P. Xing. Sparse additive generative models of text. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1041–1048, 2011.

[4] M. Hoffman, F. R. Bach, and D. M. Blei. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856–864, 2010.

[5] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 168–177, New York, NY, USA, 2004. ACM.

[6] Y. Jo and A. H. Oh. Aspect and sentiment unification model for online review analysis. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 815–824. ACM, 2011.

[7] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[8] J. Lee, S. Bengio, S. Kim, G. Lebanon, and Y. Singer. Local collaborative ranking. In *Proceedings of the 18th international conference on World wide web*, 2014.

[9] F. Li, C. Han, M. Huang, X. Zhu, Y.-J. Xia, S. Zhang, and H. Yu. Structure-aware review mining and summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 653–661. Association for Computational Linguistics, 2010.

[10] B. Liu. Opinion mining and sentiment analysis. In *Web Data Mining*, pages 459–526. Springer, 2011.

[11] B. Liu, M. Hu, and J. Cheng. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, pages 342–351, New York, NY, USA, 2005. ACM.

[12] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Recsys*, 2013.

[13] J. McAuley, J. Leskovec, and D. Jurafsky. Learning attitudes and attributes from multi-aspect reviews. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 1020–1025. IEEE, 2012.

[14] S. Moghaddam and M. Ester. Ilda: interdependent lda model for learning latent aspects and their ratings from online product reviews. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, pages 665–674, New York, NY, USA, 2011. ACM.

[15] S. Moghaddam and M. Ester. On the design of lda models for aspect-based opinion mining. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, CIKM '12, pages 803–812, New York, NY, USA, 2012. ACM.

[16] S. Moghaddam and M. Ester. The flda model for aspect-based opinion mining: addressing the cold start problem. In *Proceedings of the 22nd international conference on World Wide Web*, WWW '13, pages 909–918, 2013.

[17] J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[18] G. Ronning. Maximum likelihood estimation of dirichlet distributions. *Journal of statistical computation and simulation*, 32(4):215–221, 1989.

[19] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2007.

[20] I. Titov and R. McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM, 2008.

[21] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 448–456, New York, NY, USA, 2011. ACM.

[22] H. Wang, Y. Lu, and C. Zhai. Latent aspect rating analysis on review text data: a rating regression approach. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 783–792. ACM, 2010.

[23] H. Wang, Y. Lu, and C. Zhai. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 618–626. ACM, 2011.

[24] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, and S. Ma. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '14, Gold Coast, Australia, 2014. ACM.

[25] W. X. Zhao, J. Jiang, H. Yan, and X. Li. Jointly modeling aspects and opinions with a maxent-lda hybrid. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 56–65, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.