Multi-Aspect Rating Inference with Aspect-Based Segmentation

Jingbo Zhu, *Member*, *IEEE*, Chunliang Zhang, and Matthew Y. Ma, *Senior Member*, *IEEE*

Abstract—This paper explores the problem of content-based rating inference from online opinion-based texts, which often expresses differing opinions on multiple aspects. To sufficiently capture information from various aspects, we propose an aspect-based segmentation algorithm to first segment a user review into multiple single-aspect textual parts, and an aspect-augmentation approach to generate the aspect-specific feature vector of each aspect for aspect-based rating inference. To tackle the problem of inconsistent rating annotation, we present a tolerance-based criterion to optimize training sample selection for parameter updating during the model training process. Finally, we present a collaborative rating inference model which explores meaningful correlations between ratings across a set of aspects of user opinions for multi-aspect rating inference. We compared our proposed methods with several other approaches, and experiments on real Chinese restaurant reviews demonstrated that our approaches achieve significant improvements over others.

Index Terms—Sentiment analysis, content-based rating inference, aspect-based segmentation, collaborative rating inference

1 Introduction

With the growing availability of opinion-rich resources such as online product reviews, sentiment analysis and opinion mining have become an emerging research direction whose goal is to automatically determine people's assessments on social issues or products from the rich sources of online opinion-based texts ([5], [14], [15], [17], [22], [31] [32] [33]). Until recently, many practical applications (e.g., in the film industry) cared more about public opinions on a product in a multipoint rating scale (e.g., one to five stars) instead of positive and negative polarities.

Nowadays, people increasingly express their opinions in the form of textual reviews without explicit numeric ratings. The goal of *content-based rating inference* is to recover a user's true recommendation in the form of numeric ratings from textual reviews that reflects the user's assessment (satisfaction level) on some social issue or product. The most common way to tackle this problem is either as an n-ary classification task ([21]) or as an ordinal regression problem ([29]). We focus on utilizing ordinal regression algorithms or, more precisely, the PRanking (Perceptron-based Ranking) algorithm of Crammer and Singer [4].

In practice, there are several key issues in performing a content-based rating inference task. The first issue is how to segment a real review into multiple single-aspect textual parts as basic units, referred to as *aspect-based segmentation*. Customers generally express differing opinions on multiple aspects simultaneously in the same review and even in the same sentence. For example, a restaurant review sentence "*the food is great, but it is very expensive*" contains two aspect mentions, including a positive *food*-aspect mention "*the food is great*" and a negative *charge*-aspect mention "*it is very expensive*." From the real Chinese restaurant reviews used in our evaluation, we found that more than 20 percent of review sentences can exhibit more than one aspect, referred to as *multi-aspect sentences*. Therefore, treating a multi-aspect user review as a single-aspect mention for aspect-based rating inference cannot lead to satisfactory results.

Another key issue in a content-based rating inference task is how to accommodate the inconsistencies among various rating annotations. In the restaurant or product review domain, rating annotations are often not consistent, namely imperfect ratings ([16]). This is because user reviews are generally provided by different users, and there is a lack of calibration in a multipoint rating system used for rating annotation. For example, sometimes it is "*hard*" to distinguish the difference between the 4-point and 5-point ratings provided by two distinct users, which can possibly express the same strongly positive opinion, thus leading to ambiguities and uncertainties, referred to as *inconsistent rating annotation*.

A third issue in tackling the content-based rating inference problem is how to explore meaningful correlations between ratings across various aspects of user opinions for multi-aspect rating inference. In real reviews, a user's opinion on some aspect (e.g., positive on *environment*) would implicitly influence his/her opinions on others (e.g., positive on *service*).

To address these issues, first we present an *aspect-based segmentation* algorithm to segment a user review into multiple single-aspect textual parts as basic processing units, and present an aspect-augmentation approach to generate the aspect-specific feature vector of each aspect for aspect-based rating inference. Second, we present an improved ordinal

- *J. Zhu and C. Zhang are with the Key Laboratory of Medical Image Computing (Ministry of Education) and the Natural Language Processing Laboratory, Institute of Computer Software, College of Information Science and Engineering, Northeastern University, Shenyang 110004, Liaoning, China. E-mail: {zhujingbo, zhangcl}@mail.neu.edu.cn.*
- *M.Y. Ma is with Scientific Works, 6 Tiffany Court, Princeton Jct, NJ 08550. E-mail: mattma@ieee.org.*

Manuscript received 5 Jan. 2012; revised 3 Apr. 2012; accepted 25 Apr. 2012; published online 5 June 2012.
Recommended for acceptance by C. Pelachaud.
For information on obtaining reprints of this article, please send e-mail to: taffc@computer.org, and reference IEEECS Log Number TAFFC-2012-01-0004.
Digital Object Identifier no. 10.1109/T-AFFC.2012.18.

A real Chinese Restaurant Review Sample	
The 1st Sentence	环境不错，菜品一般，味道不怎么样，很贵。(*The environment is nice, the quality of food is so so, and the taste is not good, the food is very expensive.*)
The 2nd Sentence	*服务我很欣赏，服务细节比较到位。(I like their service very much, the service is excellent.)*
The 3rd Sentence	饮料免费，还不错。*(Drink is free, that is good.)*

Fig. 1. A real Chinese restaurant review sample.

regression algorithm to overcome the inconsistencies among various rating annotations by optimizing training sample selection for model training, referred to as *tolerance-based criterion*. Third, we present an *aspect-oriented collaborative filtering* technique which explores correlations between ratings across a set of aspects in a conventional user-oriented collaborative filtering paradigm widely used in the context of personalized recommendation. Finally, we present a *collaborative rating inference* (CRI) framework which combines the best of both the collaborative filtering and the PRanking-based paradigms. Experimental results on a large real Chinese restaurant review dataset show that our approaches can achieve significant improvements over standard approaches. Our approaches are easy to implement and also applicable to other languages, such as English, or other domains, such as product or movie reviews.

2 Rating Inference Model

A multi-aspect content-based rating inference system takes as input a set of textual reviews and multiple predefined aspects, and identifies the rank (rating) of each aspect from each review, also referred to as *aspect-based rating inference* in this paper. A real Chinese restaurant review sample is shown in Fig. 1 to describe the aspect-based rating inference task. This sample contains five aspects, including *environment*, *discount*, *food*, *charge*, and *service* aspects, each with a rank (rating) in a 5-point rating scale. The ratings of various aspects for this review sample are shown in Table 1.

The simplest way used in tackling the multi-aspect rating inference is to treat each aspect as an independent and separate ranking (or classification) problem. This paper considers the problem of content-based rating inference as a task of ranking by using a widely used ordinal regression model called the PRanking algorithm [4]. PRanking has been successfully applied to rating inference tasks, e.g., in the restaurant domain [29]. To ease our discussion, we first introduce how to utilize the PRanking algorithm as the basic ordinal regression model for single-aspect content-based rating inference.

Without loss of generality, in a single-aspect content-based rating inference task we are given a sequence $(x^1, y^1), \ldots, (x^t, y^t), \ldots$ of instance (review)-rank pairs. Each instance x^t is a feature vector and its corresponding rank (rating) y^t is an element from a set of possible ranks, i.e., $Y = \{1, 2, \ldots k\}$. When a 5-point rating system was used, i.e., $k = 5$. Note that $y^t = 5$ indicates a strongly positive polarity, and $y^t = 1$ indicates a strongly negative polarity.

TABLE 1
Rank of Various Aspects for the Review Sample Shown in Fig. 1

Aspect	Rank
Environment	4
Discount	4
Food	1
Charge	1
Service	5

PRanking is implemented by means of an online iterative learning process whose goal is to find a direction defined by a parameter vector $\mathbf{w}$ and a set of thresholds $\mathbf{b}$ and use them to predict the appropriate rating for the given review. In content-based rating inference tasks, the goal of a standard PRanking-based ordinal regression model is to learn a ranking rule H_{PR} that maps from instances to ranks, i.e., $H_{PR} : X \rightarrow Y$. The family of ranking rules employs a vector $\mathbf{w}$ and a vector of thresholds $\mathbf{b} = \{b_1, \ldots, b_k\}$ which is a set of k thresholds $b_1 \leq \cdots \leq b_{k-1} \leq b_k = \infty$. In the prediction process, an unlabeled instance x that satisfies $b_{r-1} < \mathbf{w} \bullet \mathrm{x}_r < b_r$ is assigned a rank r ($\in Y$). Formally, given a ranking rule $H_{PR}(.)$ defined by $\mathbf{w}$ and $\mathbf{b}$, the predicted rank of an unlabeled review (instance) x is

$$H_{PR}(x) = \min_{r \in Y} \{r : w \bullet x - b_r < 0\}. \quad (1)$$

At each learning round, the PRanking algorithm updates the current ranking rule $H_{PR}(.)$ by modifying $\mathbf{w}$ and $\mathbf{b}$ if a prediction mistake was made. In other words, an update decision is based on a criterion whether the predicted rank and the true rank are not same. The goal of content-based rating inference algorithms is to learn an appropriate ranking rule $H(.)$ that minimizes the distance between predicted ranks and true ranks, namely *ranking-loss*. Specifically, the PRanking-based rating inference algorithm aims to learn an appropriate $H_{PR}(.)$ which minimizes the ranking-loss defined to be the number of thresholds between the true rank and the predicted rank. A good PRanking algorithm can make the predicted rank as close as possible to the true rank for each unlabeled instance (review).

3 Aspect-Based Segmentation

In a typical content-based rating inference system ([21], [11], [29]), predicting the ratings of different aspects is implemented based on the same feature vector of the unlabeled review as input. Formally, given a unlabeled review[1] x and a ranking rule $H(.)$ (e.g., $H_{PR}(.)$ defined by (1)), the rank y_a of some aspect a can be predicted by

$$y_a = H(x, a). \quad (2)$$

However, a real review often simultaneously exhibits more than one aspect, as shown in Fig. 1. In such a case, it is problematic to predict the ranks of different aspects by taking as input the same feature vector of this review sample according to (2). To tackle this problem, a natural solution is to segment the input review into multiple

1. For notational convenience, hereafter when we refer to a textual review x, we will also use x to represent its feature vector in the case of no confusion.

A multi-aspect review sentence:
环境不错，菜品一般，味道不怎么样，很贵。
(*The environment is nice, the quality of food is so so, the taste is not good, and the food is very expensive.*)
Aspect-based segmentation:
环境不错 (*The environment is nice*)/ENVIRONMENT-segment || 菜品一般，味道不怎么样 (*the quality of food is so so, the taste is not good*)/FOOD-segment || 很贵 (*the food is very expensive*)/CHARGE-segment

Fig. 2. The segmentation of the first review sentence shown in Fig. 1.

single-aspect textual parts as basic units for aspect-based rating inference, referred to as *aspect-based segmentation*. We can reformulate (2) as

$$y_a = H(x_a, a), \tag{3}$$

where x_a represents the *aspect-specific* feature vector of aspect a for rating inference, which can be built based on the results of aspect-based segmentation on the review x.

Formally, given an unlabeled review x and L predefined aspects $\{a_1, \ldots, a_L\}$ of user opinions, the goal of aspect-based segmentation is to divide x into $L+1$ single-aspect textual parts, that is,

$$AS(x) = \{as_{nul}, as_1, \ldots, as_L\}, \tag{4}$$

where $AS(.)$ indicates an aspect-based segmentation function, as_{null} is a null-aspect textual part,[2] and as_i is the single-aspect textual part of an aspect a_i. In the next section, we will address how to utilize $AS(x)$ to generate the aspect-specific feature vector x_a of each aspect a used in (3).

The most straightforward way in tackling the aspect-based segmentation is to identify the aspect of each review sentence, and divide the whole review into multiple single-aspect textual parts by grouping some review sentences associated with the same aspect, referred to as the *full-stop-based segmentation* (FSS) method. For aspect identification, one of the state-of-the-art supervised classifiers such as Maximum Entropy (MaxEnt) model ([2]) can be designed to identify the aspect of each review sentence. However, as mentioned before, a real review often contains many multi-aspect sentences. For example, the first sentence of the review sample from Fig. 1, also shown in Fig. 2, is a multi-aspect sentence that expresses user opinions on three different aspects. It thus raises a crucial question of how to split a multi-aspect review sentence into multiple single-aspect units (segments) for aspect-based segmentation, referred to as *multi-aspect sentence segmentation* (MAS).

Fig. 2 depicts an example of MAS on the first review sentence in Fig. 1. Based on the results of MAS on each multi-aspect review sentences, the review x can be easily segmented into multiple single-aspect textual parts by grouping some single-aspect segments associated with the same aspect.

For convenience of discussion, we adopt a term of *subsentence*[3] to represent a segment candidate, separated by a comma within a review sentence. For example, "菜品一般 (*the quality of food is so so*)" and "味道不怎么样(*the taste is not good*)" are two different subsentences of the multi-aspect review sentence in Fig. 2. It is noteworthy that a subsentence is often not a complete single-aspect mention.

2. Some sentences in a real review do not express any aspect, referred to as *null-aspect sentences*.

3. The separation mark between two adjacent subsentences in a review sentence is defined as a comma or a semicolon.

Input: A review sentence $C=c_1c_2...c_n$
A (MaxEnt-based) supervised classifier F
Output: The resulting segmentation $U=u_1u_2...u_k$
Initialization:
$U=\{C\}$, $SP=\{sp_1, ..., sp_n\}$, where sp_i denotes a segmentation position candidate at the end boundary of the sub-sentence c_i.

LOOP: Iterative Segmentation
Step1: Start from the leftmost to find a $sp \in SP$ of the given sentence C, that satisfies two conditions:
- Produce two new segments v and w with respect to sp;
- v and w express two different aspects (determined by F)

Sopping Criterion:
- This algorithm can stop if such a sp in SP is NOT found at Step1.

Step2: Modify U and SP by
- U←{v, w} and $sp \notin SP$

Step3: Verify U by
- Check whether any two contiguous segments u_i and u_{i+1} in U exhibit the same aspect. If not found, go to **Step1,** otherwise
  a) Group u_i and u_{i+1} as u^*
  b) $u_i \notin U$, $u_{i+1} \notin U$, $U \leftarrow \{u^*\}$
  c) $sp^* \in SP$, where sp^* denotes the corresponding segmentation position between u_i and u_{i+1}

ENDLOOP

Fig. 3. Search-based MAS algorithm.

Formally, let $C = c_1c_2 \ldots\ c_n$ be a review sentence consisting of n subsentences, and $U = u_1u_2 \ldots u_k$ be its segmentation consisting of k segments. A segment can be a subsentence or a combination of multiple consecutive subsentences. Each segment in the segmentation U expresses a particular aspect, while any two contiguous segments exhibit different aspects. The implementation of our MAS algorithm is summarized in Fig. 3.

In principle, our algorithm is implemented by adopting the basic idea of the *grid search* algorithm ([8]), which includes three core steps. The first step aims to identify an appropriate segmentation position that can split some current segment into two smaller parts, each expressing a different aspect. In the second step, the current segmentation U is updated by adding these two small parts as new segments produced in the first step. Since it is possible that a new segment has the same aspect as that of its adjacent segment, in such a case we should group both segments of the same aspect into a new bigger segment, as shown in the third step. Our MAS algorithm continues until there is no appropriate segmentation position found, namely, *stopping criterion*.

Let us revisit the first review sentence C in Fig. 1, which consists of four subsentences, denoted by $C = c_1c_2c_3c_4$. In our MAS algorithm, the segmentation process on C can be depicted as in Fig. 4. For example, on the third round, there

Round	Segmentation	Aspects
0	$\|\|c_1c_2c_3c_4\|\|$ /* U={C}*/	<*food*>
1	$\|\|c_1\|\|c_2c_3c_4\|\|$ /*$sp = c_1\|\|c_2$, v=c_1, w=$c_2c_3c_4$*/	<*environment*> <*food*>
2	$\|\|c_1\|\|c_2\|\|c_3c_4\|\|$ /*$sp = c_2\|\|c_3$, v=c_2, w=c_3c_4*/	<*environment*> <*food*> <*charge*>
3	$\|\|c_1\|\|c_2\|\|c_3\|\|c_4\|\|$ /* $sp = c_3\|\|c_4$, v=c_3, w=c_4 sp*= $c_2\|\|c_3$, u_i=c_2, u_{i+1}=c_3, u*=c_2c_3*/	<*environment*> <*food*> <*charge*>
4	***STOP***: *satisfying the stopping criterion.*	
Resulting segmentation U*= $c_1\|\|c_2c_3\|\|c_4$ Associated aspects: <*environment*> <*food*> <*charge*>		

Fig. 4. The segmentation process of our algorithm on the first review sentence in Fig. 1.

are two separate actions. First, a new segmentation position between c_3 and c_4 is found, i.e., $U = ||c_1||c_2||c_3||c_4||$. Second, since the aspects of c_2 and c_3 are the same (i.e., *food*), the algorithm combine both units as a single-aspect segment of *food* aspect, that is, $U = ||c_1||c_2c_3||c_4||$.

4 Aspect-Specific Feature Vector Generation

In this section, we describe our approach to building the aspect-specific feature vector of each aspect based on the result of MAS on the given review. Without loss of generality, we are given a review x, having L predefined aspects $\{a_1, \ldots, a_L\}$, and the results of its aspect-based segmentation $AS(x) = \{as_{nul}, as_1, \ldots, as_L\}$. Let us first present three simple and "*obvious*" approaches below and illustrate their limitations:

- The *ALL* method ignores the results of aspect-based segmentation $AS(x)$ and simply considers the original feature vector of x (typically either a real vector or a binary vector) as the aspect-specific feature vector of each aspect a_i, as done in typical content-based rating inference systems [21], [11], [29].
- The *SinAsp* method considers the single-aspect textual part as_i to build the aspect-specific feature vector of each aspect a_i. Consequently, this method ignores the single-aspect textual parts of other aspects and as_{nul}.
- The basic idea of the *WeiAsp* method is to reweight information from the single-aspect part as_i and it does not ignore the whole information from x. For example, if there are five predefined aspects, we can weight information from the single-aspect part as_i by 5.0 to generate the aspect-specific feature vector of aspect a_i. The optimal weight can be chosen by cross validation.

In the ALL method, the aspect-specific feature vectors of different aspects are the same as the original feature vector of the input review. Since a real review often expresses different opinions (ranks) on various aspects, the SinAsp method aims to build the aspect-specific feature vector of each aspect based on the corresponding single-aspect textual part in $AS(x)$; it therefore obviously poses a risk in missing some useful information from other aspects. The WeiAsp method seems to possibly alleviate this problem by using a reweighting technique. However, the results of aspect-based segmentation $AS(x)$ are still not sufficiently utilized by the WeiAsp method.

TABLE 2
An Example of a Training Set and the Results of Aspect-Based Segmentation on Each Sample

no	Review sample x_i	Ranks	$AS(x_i)$
1	Food is good, service is good.	FOO: 4 SER: 4	{Food is good} {service is good}
2	Food is good, service is bad.	FOO: 4 SER: 1	{Food is good} {service is bad}
3	Food is bad, service is good.	FOO: 1 SER: 4	{Food is bad} {service is good}
4	Food is bad, service is bad.	FOO: 1 SER: 1	{Food is bad} {service is bad}

FOO and SER denote two different aspects, food and service.

To address these challenges, this paper presents a new method, inspired by the feature-augmentation approach [6], to build the aspect-specific feature vector of each aspect based on $AS(x)$, referred to as the *aspect-augmentation* approach. The feature-augmentation approach was originally proposed to tackle the domain adaptation problem. In the feature augmentation approach, the augmented feature space consists of $K + 1$ copies of the original feature space for K domains [6]. In principle, the multi-aspect rating inference we focus in this paper is different from the domain adaptation problem.

To sufficiently capture the information from various aspects (i.e., $AS(x)$), our aspect-augmentation approach builds the aspect-specific feature vector of each aspect with respect to an augmented input space. Formally, an augmented input space $\Phi(x)$ for L aspects can be described as

$$\Phi(x) = <V_0, V_1, \ldots, V_L>, \quad (5)$$

where V_i denotes the augmented feature space for the single-aspect part as_i in $AS(x)$. The augmented feature space $\Phi(x)$ is divided into $L + 1$ separate subspaces, each representing one of the single-aspect parts in $AS(x)$. For instance, V_i and V_j are used to represent two different single-aspect parts as_i and as_j, respectively. Note that the first subspace V_0 can be used to represent as_{nul} or the whole review x in our methods.

Before we proceed with our methods for the aspect-specific feature vector generation, let us consider why it might be expected to work. Suppose we are given four simple review samples and the results of aspect-based segmentation on each sample, as shown in Table 2. We can interpret each review sample as a feature vector with the presence of words "*food*," "*service*," "*good*," and "*bad*" using the ALL method and $\Phi(x)$ defined by (5), as shown in Table 3.

For the ALL method, an aspect-based rating inference algorithm possibly yields satisfactory performance on review samples x_1 and x_4, but has no ability to distinguish between both samples x_2 and x_3. With respect to $\Phi(x)$ in (5), the feature vectors of samples x_2 and x_3 can provide discriminative information for achieving satisfactory performance by an aspect-based rating inference algorithm.

To sufficiently capture information from the results of aspect-based segmentation $AS(x)$ on a review x, this paper

TABLE 3
The Feature Vector of Each Review Sample with Respect to the ALL Method and $\Phi(x)$

no	ALL method	$\Phi(x_i)$	Ranks
1	*food*:1 *service*:1 *good*:2	[*food*:1 *good*:1] [*service*:1 *good*:1]	FOO: 4 SER: 4
2	*food*:1 *service*:1 *good*:1 *bad*:1	[*food*:1 *good*:1] [*service*:1 *bad*:1]	FOO: 4 SER: 1
3	*food*:1 *service*:1 *good*:1 *bad*:1	[*food*:1 *bad*:1] [*service*:1 *good*:1]	FOO: 1 SER: 4
4	*food*:1 *service*:1 *bad*:2	[*food*:1 *bad*:1] [*service*:1 *bad*:1]	FOO: 1 SER: 1

[] indicates the augmented feature subspace for a single-aspect part in $AS(x)$.

TABLE 4
The Augmented Feature Vectors of the *food* Aspect Produced by Various Methods

x_1	Food is good, service is good.

$AS(x_1)$	as_{nul}={}
	as_1={food is good} //the *food* aspect
	as_2={service is good} //the *service* aspect

Φ_0	<**0**, [*food*:1 *good*:1], [*service*:1 *good*:1]>
Φ_1	<[*food*:1 *service*:1 *good*:2], [*food*:1 *good*:1], **0**>
ALL	<[*food*:1 *service*:1 *good*:2], **0**, **0**>
SinAsp	<**0**, [*food*:1 *good*:1], **0**>
WeiAsp	<[*food*:2 *service*:1 *good*:3], **0**, **0**>

The information from as_1 is weighted twice as much in the WeiAsp

presents a new method to generate the augmented feature vector $\Phi_0(x, a_i)$ of each aspect a_i as

$$\Phi_0(x, a_i) = <[as_{nul}], [as_1], \ldots, [as_L]>, \quad (6)$$

where, for notational convenience, $[as_i]$ represents the augmented feature vector of as_i in $AS(x)$ with respect to the corresponding augmented feature subspace V_i.

To tackle the problem that the SinAsp method would result in missing some useful information from the whole review x, this paper presents the second new method to generate the augmented feature vector $\Phi_1(x, a_i)$ of each aspect a_i as

$$\Phi_1(x, a_i) = <[x], 0, \ldots, 0, [as_i], 0, \ldots, 0>, \quad (7)$$

where **0** indicates the zero vector.

Roughly speaking, the method $\Phi_1(x, a_i)$ adopts a similar idea to the WeiAsp method, in which the information from as_i is weighted by 2.0 because as_i is a part of x. Besides, we can equally reformulate the ALL, SinAsp, and WeiAsp methods with respect to $\Phi(.)$ in (5) as follows:

- The ALL method can be expressed as

$$\Phi_{ALL}(x, a_i) = <[x], 0, \ldots, 0>. \quad (8)$$

- The SinAsp method can be expressed as

$$\Phi_{SinAsp}(x, a_i) = < 0, \ldots, 0, [as_i], 0 \ldots, 0 > . \quad (9)$$

- The WeiAsp method can be expressed as

$$\Phi_{WeiAsp}(x, a_i) = <[x^*], 0, \ldots, 0>. \quad (10)$$

Note that $[x^*]$ in $\Phi_{WeiAsp}(.)$ denotes an augmented feature vector in which the information from as_i is weighted for aspect a_i. Let us revisit the first review sample x_1 shown in Table 2. The augmented feature vectors of the *food* aspect produced by various methods are depicted in Table 4.

5 Inconsistent Rating Annotation

In a typical rating prediction task, e.g., in the context of recommender systems and user preference prediction [1], [19], each reviewer rated a certain (large) number of objects (e.g., movies in the EachMovie dataset). The inconsistent rating annotation problem is generally ignored for the same reviewer. However, the inconsistent rating annotation problem existing in real restaurant reviews is severe enough such that there is no guarantee that the PRanking training algorithm can find the right direction for parameter update at each learning round because the inconsistent rating annotation problem would result in an undesired parameter update decision made during the training process.

Let us take a look at the example shown in Table 5. Two real Chinese restaurant reviews on the *food* aspect are provided by two distinct users. In practice, different users rated the same object (i.e., restaurant) independently.

We first ran a small pilot investigation, in which 10 native Chinese graduates were asked to independently rate both reviews with respect to a 5-point rating scheme, without looking at the ratings posted by the users. Eight of them rated both reviews as a 5-point rank, and a 4-point rank was chosen by the rest. Interestingly, no one assigned distinct ranks to these two reviews shown in Table 5. According to the pilot investigation results, both exemplary reviews should be rated to the same rank (5-point or 4-point). However, user1 and user2 rated them as two distinct ranks, indicating an inconsistent rating annotation problem.

Suppose that both exemplary reviews are used for PRanking training at the tth learning round; if the current ranking rule $H_{PR}()$ predicts the same rank to both example reviews, this inconsistent rating annotation problem would result in making an undesired update decision on the current ranking rule $H_{PR}(.)$ due to distinct true ranks for both reviews. To remedy this problem, this paper presents a *tolerance-based criterion* to determine whether an update decision should be made at each learning round.

As mentioned before, a standard PRanking learning algorithm makes an update decision when the predicted rank and the true rank of the current instance (review) x are not the same, defined as

TABLE 5
Two Real Restaurant Reviews on the *food* Aspect

User	Rating	Review
1	4	因为真的是很经济实惠, 超喜欢烤鸡翅 (*It is really very cheap and good value. I favor roasted chicken wings very much.*)
2	5	今天我在这里吃晚饭，感觉非常好。 (*Today I have a dinner here. My feeling is great.*)

$$\hat{y}^t \neq y^t. \tag{11}$$

Our tolerance-based criterion adopts a "*soft*" way to make the update decision during the learning process in which (11) can be modified as

$$\left|\hat{y}^t - y^t\right| > \alpha, \tag{12}$$

where the tolerance factor α is an integer between $[1, k-1]$. Note that k is the highest rank.

In a 5-point rating system, 4-point is closer to 5-point than to 2-point. As observed in the pilot investigation mentioned above, it indicates no clear gap between ratings 4-point and 5-point in practice. In other words, 4-point and 5-point are all possibly used by two different users to express the same strongly positive opinion. The goal of introducing a tolerance factor α in (12) is to reduce the effects on incorrectly updating the ranking rule that is caused by such a small gap. For example, $\alpha = 1$ indicates that the current rank rule $H_{PR}(.)$ can be kept when the current training instance (review) x with 5-point true rank is predicted as 4-point rank according to (12).

6 Collaborative Rating Inference

6.1 Collaborative Filtering

A real review generally contains a user's opinions on more than one aspect simultaneously, and a user's opinion on some aspects often influences his/her opinions on other aspects. For example, a "*luxurious*" restaurant often provides "*excellent*" service and "*expensive*" foods, i.e., positive on *environment* => positive on *service* and negative on *charge*. It is significant to explore meaningful correlations between ratings (opinions) across a set of aspects by adopting the *collaborative filtering* paradigm [10], [27].

Collaborative filtering is a process of filtering for information or patterns based on collaboration among multiple agents (e.g., customers), and in recent years has been successfully applied to many e-commerce applications such as Amazon and Netflix. A typical collaborative filtering system aims to answer a question [27]: Based on a rating matrix and the known ratings (opinions) of an active (current) user, can we predict how this user will rate the other items (product)? The rating matrix represents the rating of each user on various items.

However, in practice, there are two challenges for applying conventional collaborative filtering techniques to multi-aspect rating inference. First, conventional collaborative filtering generally assumes a single score (rating) per object (e.g., product or movie). In the restaurant review datasets used in our evaluation, each review rates a restaurant on multiple aspects such as *food* and *service*. Second, in our evaluation dataset, only an average 1.03 reviews were provided by each user. Since there are no sufficient rating data for each user, it is almost impossible to explore meaningful correlations between ratings across multiple users.

To address these challenges, unlike conventional *user-oriented collaborative filtering* (UOCF) paradigms [10], [27], this paper adapts an *aspect-oriented collaborative filtering* (AOCF) paradigm that explores correlations between ratings across a set of aspects by replacing the *user-specific* similarity metric with an *aspect-specific* similarity metric. In AOCF, there is sufficient rating data for each aspect and a single rating per object (review) for each aspect.

To implement the AOCF paradigm, we adapt a *Pearson correlation coefficient* (PCC)-based approach [27] in which PCC is used as the correlation measure between different aspects. PCC corresponds to an inner product between normalized rating vectors, and is defined as

$$\begin{aligned} PCC(a,b) &= \frac{\mathrm{cov}(a,b)}{\sigma_a \sigma_b} \\ &= \frac{\sum_{1\le i\le m}(a_i - \bar{a})(b_i - \bar{b})}{\sqrt{\sum_{1\le i\le m}(a_i - \bar{a})^2}\sqrt{\sum_{1\le i\le m}(b_i - \bar{b})^2}}, \end{aligned} \tag{13}$$

where $a = \{a_1, \ldots, a_m\}$ and $b = \{b_1, \ldots, b_m\}$ represent m ratings of two aspects a and b in the training set, and $\bar{a}$ and $\bar{b}$ denote the average of a's and b's ratings, respectively.

Let Ψ be a set of L predefined aspects of user opinions. To predict an aspect a's rating prediction $H_{CF}(x,a)$ on an input unlabeled review x, we can take a weighted average of all the rating predictions of other aspects on this review, defined as

$$H_{CF}(x,a) = \left[\bar{a} + \frac{\sum_{b\in\Psi\wedge b\neq a}(b_x - \bar{b})PCC(a,b)}{\sum_{b\in\Psi\wedge b\neq a} PCC(a,b)}\right], \tag{14}$$

where $[.]$ denotes a mathematical operation of rounding to integer, e.g., $[4.3] = 4$ or $[4.6] = 5$. b_x denotes the rating of the aspect b on the unlabeled review x and can be predicted by the PRanking algorithm.

6.2 Collaborative Rating Inference

Now we have two different paradigms including PRanking and AOCF for multi-aspect rating inference. As mentioned before, the PRanking algorithm predicts the rating of each aspect from the content of the unlabeled review, as shown in (1), and treats each aspect as an independent and separate ranking problem to implement multi-aspect rating inference. As shown in (14), the AOCF paradigm finds aspects most similar to some active (predicted) aspect, and then forms a weighted vote over these neighbors to predict unobserved ratings on the unlabeled reviews. We pursue that PRanking and AOCF are two complementary views for multi-aspect rating inference and can be unified in a common learning architecture.

To combine the best of both paradigms, this paper utilizes a *linear combination* method to predict the rating of each aspect a on an unlabeled review x, referred to as *collaborative rating inference*, defined as

$$H_{CRI}(x,a) = [\beta_a \times H_{PR}(x,a) + (1-\beta_a) \times H_{CF}(x,a)], \tag{15}$$

where $[.]$ denotes a mathematical operation of rounding to integer. $H_{CRI}(x,a)$ denotes the CRI rating prediction of some aspect a on the unlabeled review x. For each aspect a, $\beta_a \in [0,1]$ is a real-value coefficient used to balance the contributions of both techniques.

One key implementation issue is to how to automatically determine the optimal value of β during the online learning process. To tackle this problem, this paper presents a joint learning algorithm in which two parameter vectors $\mathbf{w}$ and $\mathbf{b}$

Input: (x^1,y^1), …, (x^m,y^m) ,
PRanking rule $H_{PR}(.)$ (from Equation 1)
Aspect-based collaborative filtering rule $H_{CF}(.)$ (from Equation 14)
Collaborative and content-based rating inference rule $H_{CRI}(.)$ (from Equation 15)

Initialize: Set $w[i]^1 = 0, b[i]_1^1,...,b[i]_{k-1}^1 = 0, b[i]_k^1 = \infty, \beta[i]^1 = 1$ // Linear combination coefficient β

Loop: FOR t = 1,2,...,m

1. Get a new instance x^t.
2. Predict $\hat{y}^t_{PR} = H_{PR}(x^t; w^t, b^t)$ (Equation 1); $\hat{y}^t_{CF} = H_{CF}(x^t; H_{PR}(.))$ (Equation 14);
   $\hat{y}^t = H_{CRI}(x^t; H_{PR}(.), H_{CF}(.), \beta^t)$ (Equation 14).
3. Get the true rank y^t of x^t.
4. FOR aspect i=1, ..., l:
   IF $\hat{y}[i]^t \neq y[i]^t$ THEN update model (otherwise set $w^{t+1} = w^t, \forall r : b_r^{t+1} = b_r^t, \beta^{t+1} = \beta^t$):
   a) FOR r = 1, …, *k-1* : IF $y[i]^t \leq r$ THEN $y[i]_r^t = -1$ ELSE $y[i]_r^t = 1$.
   b) FOR r = 1, …, *k-1* : IF $(\hat{y}_{PR}[i]^t - r) y[i]_r^t \leq 0$ THEN $\tau[i]_r^t = y[i]_r^t$ ELSE $\tau[i]_r^t = 0$.
   c) Update $w[i]^{t+1} \leftarrow w[i]^t + (\sum_r \tau[i]_r^t) x^t$.
   d) FOR r = 1, …, *k-1* : update $b[i]_r^{t+1} \leftarrow b[i]_r^t - \tau[i]_r^t$.
   e) SWITCH $\delta = (\hat{y}[i]^t - y[i]^t)(\hat{y}_{PR}[i]^t - \hat{y}_{CF}[i]^t)$
      CASE δ>0 : $\beta[i]^{t+1} = \beta[i]^t - \varepsilon$
      CASE δ<0 : $\beta[i]^{t+1} = \beta[i]^t + \varepsilon$
      CASE δ=0 : $\beta[i]^{t+1} = \beta[i]^t$
   ENDSWITCH
   ENDIF
ENDFOR

Output: $H_{CRI}(x; w^{T+1}, b^{T+1}, \beta^{T+1})$.

Fig. 5. A joint learning framework based on the PRanking training algorithm [4].

of $H_{PR}(.)$ and the coefficient β can be jointly tuned to learn a optimal ranking rule $H_{CRI}(.)$. This joint learning algorithm is implemented based on the online learning framework of the PRanking algorithm, as shown in Fig. 5.

The standard criterion of making updates on vectors **w** and **b** is defined as checking whether the rank predicted by $H_{CRI}(.)$ and the gold rank are the same (see (11)). When the tolerance-based criterion is considered, we can use the (12). If a prediction mistake is made, the model parameters (**w**, **b**, and β) are updated according to steps 4(a-e) as shown in Fig. 5, where steps 4(a-d) are the same updates on **w** and **b** used by the standard PRanking training algorithm [4]. Our joint learning framework jointly tunes the coefficient β with **w** and **b** at each learning round, as shown in the step 4(e). The goal of parameter updates during the training process is to make the next predicted rank as close as possible to the true rank. To achieve this goal, step 4(e) shows, respectively, two different cases of updating the parameter β to be a larger or smaller value. Note that ϵ is used to control the update step on the coefficient β, and was set to 0.01 in our evaluation.

7 Evaluation

7.1 Settings

We constructed some experiments to evaluate the effectiveness of various aspect-based rating inference techniques, including *PRanking* (baseline), *aspect-oriented collaborative filtering*, *collaborative rating inference* with different $\Phi_i(.)$, two aspect-based segmentation methods (FSS and MAS), and the tolerance-based criterion on a publicly available Chinese restaurant review dataset (ChiSet).[4] The ChiSet was collected from the *DianPing.com*, and contains 13,350 Chinese reviews for 100 restaurants. Each review is accompanied by a set of three aspects covering *environment*, *food*, and *service*, and is rated with respect to a 5-point scale. In the preprocessing step we utilized the NEUCSP[5] tool to implement Chinese word segmentation and POS tagging. After the removal of stop words, each review is represented as a vector of lexical features.

We first randomly selected 1,000 review sentences to investigate the distribution of different types of sentences, including multi-aspect, single-aspect, and null-aspect sentences. Table 6 depicts that 23.5 percent of review sentences are multi-aspect sentences, indicating that MAS is an important issue for aspect-based rating inference.

In the following rating inference experiments, we randomly selected 10 percent of data as a development set, and performed a 10-fold cross validation on the remaining 90 percent data. The development set was used to determine optimal numbers[6] of training iterations for each method. All experimental results reported are the average of 10 trials. To build the maximum entropy

4. http://www.nlplab.com/neu_restaurant_reviews.rar.
5. NEUCSP is a Chinese word segmentation and POS tagging tool at http://www.nlplab.com/chinese/source.htm.
6. Experimental results show that four training iterations can often result in satisfactory performances for each model.

TABLE 6
Distribution Analysis of Three Types of Review Sentences

Types	Number of Sentences
Multi-aspect sentence	235 (23.5%)
Single-aspect sentence	509 (50.9%)
Null-aspect sentence	256 (25.6%)
Total	1000 (100%)

(MaxEnt) [2] classifier for aspect identification used by our MAS algorithm, 1,000 single-aspect review sentences were randomly chosen for human annotation on these four aspects, including the null aspect. The average accuracy of the resulting MaxEnt-based classifier used for aspect identification is 82 percent.

In our experiments, two different evaluation metrics ([4], [1]) were adopted to quantify the accuracy of predicted ratings for each aspect $\hat{R} = (\hat{r}_1, \ldots, \hat{r}_n)$ as compared to the true ratings $R = (r_1, \ldots, r_n)$. Ranking loss and zero-one error are defined as below to assess the effectiveness of each method to be evaluated.

- *Ranking Loss*: The ranking loss is the average deviation of the predicted rating from the actual rating, defined as

$$loss(\hat{R}, R) = \left\|\hat{R} - R\right\|_1 / n.$$

- *Zero-One Error*: The zero-one error metric assigns an error of 1 to every incorrect prediction, defined as

$$ZOE(\hat{R}, R) = |\{i : \hat{r}_i \neq r_i\}|/n.$$

Lower values of ranking loss and zero-one error indicate better performance.

7.2 Results

Table 7 depicts the effectiveness of various content-based rating inference methods. No tolerance-based criterion was used for model training. We evaluate the effectiveness of different methods $\Phi_i(.)$ to generate the aspect-specific feature vector of each aspect for rating inference. Note that the ALL method is the baseline method of aspect-specific feature vector generation and is the same as that used by conventional content-based rating inference systems ([21], [11], [29]). Tables 9, 10, 11, 12, 13, and 14 depict results of paired t-tests (p-value > 0.05) between various rating inference methods reported in Tables 7 and 8.

Regarding the ALL method for generating the aspect-specific feature vector of each aspect, Table 7 shows that the CRI method can achieve significant improvements over the

TABLE 7
Ranking Loss and Zero-One Error of Various Rating Inference Methods without Using Tolerance-Based Criterion for Model Training, Including the PRanking, Aspect-Oriented Collaborative Filtering, and Collaborative Rating Inference

Method	Ranking Loss				Zero-One Error			
PRanking	Env.	Food	Service	Average	Env.	Food	Service	Average
ALL	0.831	0.870	0.886	0.862	0.633	0.650	0.642	0.642
SinAsp-FSS	1.159	1.028	0.996	1.061	0.759	0.702	0.649	0.703
WeiAsp-FSS	0.848	0.909	0.882	0.880	0.637	0.667	0.642	0.649
$\Phi_0(.)$-FSS	0.802	0.869	0.859	0.843	0.621	0.652	0.635	0.636
$\Phi_1(.)$-FSS	0.852	0.868	0.861	0.861	0.643	0.649	0.632	0.641
SinAsp-MAS	1.100	1.009	1.051	1.054	0.739	0.697	0.670	0.702
WeiAsp-MAS	0.841	0.912	0.869	0.874	0.638	0.666	0.630	0.645
$\Phi_0(.)$-MAS	**0.795**	**0.845**	**0.841**	**0.827**	**0.617**	**0.636**	**0.628**	**0.627**
$\Phi_1(.)$-MAS	0.822	0.855	0.854	0.844	0.634	0.641	**0.628**	0.634
AOCF	Env.	Food	Service	Average	Env.	Food	Service	Average
ALL	0.845	0.875	0.818	0.846	0.641	0.657	0.631	0.643
SinAsp-FSS	0.905	1.019	0.931	0.952	0.661	0.689	0.670	0.673
WeiAsp-FSS	0.852	0.872	0.829	0.851	0.649	0.656	0.633	0.646
$\Phi_0(.)$-FSS	0.859	0.870	0.820	0.850	0.652	**0.650**	**0.625**	0.642
$\Phi_1(.)$-FSS	0.836	0.873	0.817	0.842	0.644	0.654	0.628	0.642
SinAsp-MAS	0.901	1.002	0.906	0.937	0.661	0.679	0.668	0.669
WeiAsp-MAS	0.845	0.885	0.832	0.854	0.639	0.657	0.635	0.644
$\Phi_0(.)$-MAS	0.864	**0.865**	0.813	0.847	0.655	0.652	0.630	0.646
$\Phi_1(.)$-MAS	**0.831**	0.870	**0.810**	**0.837**	**0.636**	0.653	0.626	**0.638**
CRI	Env.	Food	Service	Average	Env.	Food	Service	Average
ALL	0.710	0.792	0.779	0.760	0.588	0.632	0.608	0.609
SinAsp-FSS	0.830	0.853	0.842	0.841	0.648	0.653	0.636	0.646
WeiAsp-FSS	0.723	0.809	**0.772**	0.768	0.599	0.636	0.607	0.614
$\Phi_0(.)$-FSS	0.719	0.809	0.792	0.773	0.597	0.635	0.621	0.618
$\Phi_1(.)$-FSS	0.712	0.788	0.774	0.758	0.594	0.627	0.615	0.612
SinAsp-MAS	0.817	0.886	0.840	0.847	0.641	0.654	0.642	0.645
WeiAsp-MAS	0.722	0.812	0.785	0.773	0.594	0.636	0.607	0.613
$\Phi_0(.)$-MAS	0.718	**0.780***	0.804	0.767	0.593	**0.615***	0.621	0.610
$\Phi_1(.)$-MAS	**0.700***	0.786	**0.772***	**0.753***	**0.579***	0.627	**0.604***	**0.603***

*The best performance for each rating inference method is in **boldface**. Among all methods, the best performance is in **boldface** and marked with a symbol "*" in each column. "Env." indicates the* environment *aspect.*

TABLE 8
Ranking Loss and Zero-One Error of Various Rating Inference Methods with Tolerance-Based Criterion $(\alpha = 1)$ for Model Training, Including the PRanking, Aspect-Based Collaborative Filtering and Collaborative Rating Inference

Method	Ranking Loss				Zero-One Error			
PRanking	Env.	Food	Service	Average	Env.	Food	Service	Average
ALL	0.769	0.822	0.830	0.807	0.615	0.628	0.624	0.622
SinAsp-FSS	0.981	1.027	0.912	0.973	0.692	0.707	0.660	0.686
WeiAsp-FSS	0.800	0.876	0.824	0.834	0.623	0.662	0.628	0.638
$\Phi_0(.)$-FSS	0.745	0.796	0.800	0.780	0.603	0.633	0.626	0.621
$\Phi_1(.)$-FSS	0.758	0.798	0.817	0.791	0.610	0.622	0.620	0.617
SinAsp-MAS	1.013	1.013	0.994	1.007	0.698	0.702	0.677	0.692
WeiAsp-MAS	0.803	0.852	0.821	0.825	0.627	0.653	0.623	0.634
$\Phi_0(.)$-MAS	**0.738**	**0.780**	**0.779**	**0.766**	**0.599**	**0.620**	0.615	**0.611**
$\Phi_1(.)$-MAS	0.761	0.807	0.805	0.791	0.608	0.632	**0.613**	0.618
AOCF	Env.	Food	Service	Average	Env.	Food	Service	Average
ALL	0.827	0.871	0.779	0.826	0.637	0.657	**0.613**	0.636
SinAsp-FSS	0.879	0.925	0.887	0.897	0.659	0.683	0.647	0.663
WeiAsp-FSS	0.841	0.831	0.804	0.825	0.643	0.641	0.626	0.637
$\Phi_0(.)$-FSS	0.815	0.807	0.775	0.799	0.634	0.632	0.623	0.630
$\Phi_1(.)$-FSS	0.819	0.839	0.773	0.810	0.632	0.646	0.619	0.632
SinAsp-MAS	0.866	0.944	0.904	0.905	0.653	0.686	0.662	0.667
WeiAsp-MAS	0.829	0.850	0.788	0.822	0.638	0.647	0.623	0.636
$\Phi_0(.)$-MAS	**0.812**	**0.800**	**0.767**	**0.793**	0.634	**0.629**	0.619	**0.628**
$\Phi_1(.)$-MAS	0.813	0.847	**0.767**	0.809	**0.630**	0.646	0.618	0.631
CRI	Env.	Food	Service	Average	Env.	Food	Service	Average
ALL	0.732	0.771	0.750	0.751	0.603	0.621	0.609	0.611
SinAsp-FSS	0.808	0.833	0.809	0.817	0.637	0.655	0.648	0.647
WeiAsp-FSS	0.743	0.787	0.766	0.765	0.610	0.631	0.616	0.619
$\Phi_0(.)$-FSS	0.717	0.763	0.756	0.745	0.595	0.620	0.617	0.611
$\Phi_1(.)$-FSS	0.721	0.759	**0.740***	0.740	0.596	**0.613***	**0.606***	**0.605***
SinAsp-MAS	0.811	0.847	0.834	0.831	0.640	0.657	0.647	0.648
WeiAsp-MAS	0.734	0.794	0.782	0.770	0.604	0.636	0.624	0.622
$\Phi_0(.)$-MAS	**0.709***	**0.754***	0.745	**0.736***	**0.588***	0.615	0.611	**0.605***
$\Phi_1(.)$-MAS	0.714	0.761	**0.740***	0.738	0.590	0.620	0.611	0.607

*The best performance for each rating inference method is in **boldface**. Among all methods, the best performance is in **boldface** and marked with a symbol "*" in each column. "Env." indicates the environment aspect.*

PRanking in terms of ranking loss and zero-one error for each aspect. As mentioned before, the AOCF is implemented based on the output of the PRanking, i.e., b_x in (14). In such a case, the effectiveness of the PRanking on b_x can influence the performance of the AOCF on $H_{CF}(x, a)$, as shown in (14). Compared to the PRanking, the AOCF achieves better performance in ranking loss, but obtains almost the same performance in zero-one error.

Among all aspect-specific feature vector generation methods $\Phi_i(.)$, Table 7 shows that the SinAsp results in negative effects on the performance of each rating inference method. In practice, a possible reason is that most of the real reviews used in our evaluation often miss the aspect-specific information for some aspect. In such a case, the SinAsp method would produce a null feature vector for aspect-based rating inference on some aspect without any corresponding aspect-specific information in a real review. It also indicates that only using the aspect-specific information of each aspect for aspect-based rating inference is not sufficient. Daume [6] reported that the simple WeiAsp is one of the

TABLE 9
Paired t-Tests between Various Rating Inference Methods with the ALL Setting Reported in Tables 7 and 8 in Terms of *Average Ranking Loss*, Involving the PRanking, Aspect-Based Collaborative Filtering, and Collaborative Rating Inference

	PRanking	AOCF	CRI	PRanking (α=1)	AOCF (α=1)	CRI (α=1)
PRanking	N/A	<<	<<	<<	<<	<<
AOCF	>>	N/A	<<	<<	<<	<<
CRI	>>	>>	N/A	>>	>>	<<
PRanking (α=1)	>>	>>	<<	N/A	>>	<<
AOCF (α=1)	>>	>>	<<	<<	N/A	<<
CRI (α=1)	>>	>>	>>	>>	>>	N/A

Given p-value > 0.05, the notions of A (row) ">>" B(column), "<<" and "~" indicate A is better, worse and no significant difference than/from B on performance comparison, respectively.

$\alpha = 1$ *indicates the use of the tolerance-based criterion for model training.*

TABLE 10
Paired t-Tests between Various Rating Inference Methods with the ALL Setting Reported in Tables 7 and 8 in Terms of *Average Zero-One Error*, Involving the PRanking, Aspect-Based Collaborative Filtering, and Collaborative Rating Inference

	PRanking	AOCF	CRI	PRanking (α=1)	AOCF (α=1)	CRI (α=1)
PRanking	N/A	~	<<	<<	<<	<<
AOCF	~	N/A	<<	<<	<<	<<
CRI	>>	>>	N/A	>>	>>	~
PRanking (α=1)	>>	>>	<<	N/A	>>	<<
AOCF (α=1)	>>	>>	<<	<<	N/A	<<
CRI (α=1)	>>	>>	~	>>	>>	N/A

Given p-value > 0.05, the notions of A (row) ">>" B(column), "<<" and "~" indicate A is better, worse and no significant difference than/from B on performance comparison, respectively.

$\alpha = 1$ *indicates the use of the tolerance-based criterion for model training.*

TABLE 11
Paired t-Tests between Various Rating Inference Methods with the Best Setting Reported in Tables 7 and 8 in Terms of *Average Ranking Loss*, Involving the PRanking, Aspect-Based Collaborative Filtering, and Collaborative Rating Inference

	PRanking	AOCF	CRI	PRanking (α=1)	AOCF (α=1)	CRI (α=1)
PRanking	N/A	>>	<<	<<	<<	<<
AOCF	<<	N/A	<<	<<	<<	<<
CRI	>>	>>	N/A	>>	>>	<<
PRanking (α=1)	>>	>>	<<	N/A	>>	<<
AOCF (α=1)	>>	>>	<<	<<	N/A	<<
CRI (α=1)	>>	>>	>>	>>	>>	N/A

Given p-value > 0.05, the notions of A (row) ">>" B(column), "<<" and "~" indicate A is better, worse and no significant difference than/from B on performance comparison, respectively.

$\alpha = 1$ *indicates the use of the tolerance-based criterion for model training. Note that the performance of each rating inference method with the best setting is reported in* ***boldface*** *in Tables 7 and 8.*

TABLE 12
Paired t-Tests between Various Rating Inference Methods with the Best Setting Reported in Tables 7 and 8 in Terms of *Average Zero-One Error*, Involving the PRanking, Aspect-Based Collaborative Filtering, and Collaborative Rating Inference

	PRanking	AOCF	CRI	PRanking (α=1)	AOCF (α=1)	CRI (α=1)
PRanking	N/A	>>	<<	<<	~	<<
AOCF	<<	N/A	<<	<<	<<	<<
CRI	>>	>>	N/A	>>	>>	~
PRanking (α=1)	>>	>>	<<	N/A	>>	<<
AOCF (α=1)	~	>>	<<	<<	N/A	<<
CRI (α=1)	>>	>>	~	>>	>>	N/A

Given p-value > 0.05, the notions of A (row) ">>" B(column), "<<" and "~" indicate A is better, worse and no significant difference than/from B on performance comparison, respectively.

$\alpha = 1$ *indicates the use of the tolerance-based criterion for model training. Note that the performance of each rating inference method with the best setting is reported in* ***boldface*** *in Tables 7 and 8.*

state-of-the-art methods for domain adaptation. However, Table 7 shows that in most cases the WeiAsp method yields no significant improvement over the ALL method for each rating inference method in terms of both performance metrics. One possible cause is that WeiAsp can be affected by an incorrect segmentation caused by MAS algorithm on some multi-aspect review sentences because WeiAsp method reweighs information from the segmentation.

For each rating inference methods, in most cases MAS outperforms FSS for $\Phi_0(.)$ and $\Phi_1(.)$, and all the best performances can be achieved by using our aspect-augmentation method and the MAS algorithm (i.e., $\Phi_0(.)$-MAS and $\Phi_1(.)$-MAS), as shown in Table 7. However, in practice, the MAS algorithm can possibly result in incorrect segmentation on some multi-aspect review sentences. These incorrect segmentation results might cause negative effects on the WeiAsp method when reweighting the information from these incorrect segments, e.g., the WeiAsp-MAS for the CRI method shown in Table 7. Among all methods, our CRI method with $\Phi 1(.)$-MAS can achieve the best performance in terms of average ranking loss and zero-one error.

As discussed in Section 5, in practice there is often a problem of inconsistent rating annotations among distinct users that would cause negative effects on the performance of content-based rating inference. Table 8 depicts the effectiveness of various rating inference methods by incorporating a tolerance-based criterion for model training to address this issue. Seen from Tables 7 and 8, the tolerance-based criterion yields significant improvement for each rating inference method in most cases. Table 8 also shows that incorporating the tolerance-based criterion for model training can significantly improve the standard PRanking algorithm for different segmentation methods and different $\Phi_i(.)$ used.

Among all rating inference methods shown in Table 8, the CRI with $\Phi_0(.)$-MAS works the best in terms of average

TABLE 13
WindowDiff Values of Various Methods for Aspect-Based Segmentation on 1,000 Review Sentences

Methods	WindowDiff
FSS	0.20
COMMA	0.68
This work (MAS)	**0.15**

TABLE 14
Ranking Loss of Various Rating Inference Methods with the ALL Method and the Best (Average Ranking-Loss) Settings, Including the PRanking, Aspect-Based Collaborative Filtering, Collaborative Rating Inference, and Good Grief Algorithm ([29])

Method	Env.	Food	Service	Average
GG	0.820	0.840	0.831	0.830
PRanking	0.831	0.870	0.886	0.862
PRanking*	0.795	0.845	0.841	0.827
AOCF	0.845	0.875	0.818	0.846
AOCF*	0.831	0.870	0.810	0.837
CRI	0.710	0.792	0.779	0.760
CRI*	**0.700**	**0.786**	**0.772**	**0.753**

The bold number indicates the best ranking loss performance for each aspect. "Env." indicates the environment *aspect.*

ranking loss and zero-one error metrics. Among all competing methods shown in Tables 7 and 8, the CRI with $\Phi_0(.)$-MAS and tolerance-based criterion achieves the best average ranking loss performance (0.736), and the CRI with $\Phi_1(.)$-MAS achieves the best zero-one error performance (0.603).

7.3 Aspect-Based Segmentation

We randomly selected 1,000 review sentences for evaluating the effectiveness of three aspect-based sentence segmentation methods, including FSS, MAS, and COMMA. The COMMA method is to segment a review sentence by comma or semicolon. To form the gold standard, two human judges were asked to segment each review sentence in terms of aspect change. If there were disagreements between the two judges, a third human judge acted as an adjudicator.

In defining the performance of segmentation, precision and recall have the shortcoming that every inaccurately estimated segment boundary is penalized equally whether it is near or far from a true segment boundary. We adopt the *WindowDiff* metric [24], which is defined as

$$WindowDiff(ref, hyp) = \frac{1}{N-k}\sum_{i=1}^{N-k}(|b(ref_i, ref_{i+k}) - b(hyp_i, hyp_{i+k})| > 0),$$

where ref and hyp represent the reference segmentation and a hypothesized segment. $b(i,j)$ denotes the number of segmented boundaries between positions i and j in the sentence, and N denotes the number of subsentences in that sentence. The value k is the average number of subsentences per segment in the gold standard, which is set to 3.

By definition, the smaller the WindowDiff value is, the better the segmentation performance is. As shown in Table 13, our segmentation model (MAS) achieves the best performance, whereas the comma-based method is the worst. In our experimental data, only 23.5 percent of review sentences in the restaurant reviews are multi-aspect cases. Therefore, the COMMA method makes wrong segmentation decisions on sentences containing one or more commas (or semicolons), and the FSS method has no ability to correctly segment each multi-aspect review sentence.

TABLE 15
Zero-One Error of Various Rating Inference Methods with the ALL Method and the Best (Average Zero-One Error) Settings

Method	Env.	Food	Service	Average
GG	0.631	0.644	0.633	0.636
PRanking	0.633	0.650	0.642	0.642
PRanking*	0.617	0.636	0.628	0.627
AOCF	0.641	0.657	0.631	0.643
AOCF*	0.636	0.653	0.626	0.638
CRI	0.588	0.632	0.608	0.609
CRI*	**0.579**	**0.627**	**0.604**	**0.603**

TABLE 16
The Best Settings of Each Rating Inference Method Reported in Table 7

	PRanking*	AOCF*	CRI*
Ranking loss	$\Phi_0(.)$-MAS	$\Phi_1(.)$-MAS	$\Phi_1(.)$-MAS
Zero-one error	$\Phi_0(.)$-MAS	$\Phi_1(.)$-MAS	$\Phi_1(.)$-MAS

7.4 Further Comparison

We also compare various methods to the Good Grief algorithm[7] (GG) proposed by Snyder and Barzilay [29] in terms of ranking loss and zero-one error. Tables 14 and 15 compare each aspect-based rating inference method with the ALL method, GG [29], and each rating inference method with the best settings. "*" represents the best setting of each method reported in Table 7.

In this experiment, GG is based on the PRanking algorithm, and models the agreement among all aspects for multi-aspect rating inference. As shown in Tables 14 and 15, GG can achieve better performance than the baseline (the standard PRanking algorithm), AOCF and AOCF* methods, but obtains worse performance than the PRanking* and our CRI methods in terms of average ranking loss and zero-one error metrics.

In our ChiSet evaluation dataset, most reviews tend to express differing degrees of satisfaction on different aspects. In such a case, the CRI model can effectively capture meaningful correlations between different aspects for multi-aspect rating inference that the GG model cannot handle sufficiently. It is noteworthy that the PRanking can outperform the GG by using aspect-based segmentation technique and our aspect-augmentation method, i.e., the PRanking with $\Phi_0(.)$-MAS. Further, as shown in Table 16, we can see that the best setting of each rating inference method is based on our augmented feature vector generation methods (Φ_0 and Φ_1) and MAS-based segmentation algorithm.

8 Related Work

One type of related work is sentiment classification, whose goal is to predict the polarities of opinion texts with respect

7. The code for Good Grief algorithm used in this paper is available at http://people.csail.mit.edu/bsnyder/naacl07.

to positive or negative by using supervised binary classification techniques [23], [30], [28]. Recent work on aspect-based opinion analysis generally focused on product reviews [25], [20], [9], [18], [26], [14], [15], [7]. One representative work of such techniques is feature-based sentiment summarization [14], [15], which aims to produce a summary expressing the aggregated sentiment for each feature of a product and supporting textual evidence. Ding et al. [7] further proposed a holistic lexicon-based approach to improve the method of Hu and Liu [15] by addressing two issues: 1) opinion words that are content dependent, and 2) aggregating multiple opinion words in the same review sentence. Titov and McDonald [31] used the aspect ratings manually provided by users for the purpose of sentiment analysis.

In related areas of opinion extraction from user reviews, some previous efforts have focused on the extraction of opinion topics [14], [15], [25] that is limited to extracting the mentions of product names and their features. Kim and Hovy [17] presented a technique for extracting opinion topics based on semantic frames, and provided a limited evaluation. However, these previous efforts did not address the issue of segmentation of multi-aspect sentences for aspect mention extraction or opinion topic extraction.

Some previous research work expanded traditional binary sentiment classification to rating inference with respect to a multipoint scale [21], [11]. These previous studies focused on single-aspect content-based rating inference tasks. To address the multi-aspect rating prediction (inference) issue, some researchers [10], [27], [13], [1], [19] studied the rating prediction issue under a collaborative filtering framework that considers the similarities among different users. However, in practice, since a user generally provides only a few (less than three) restaurant reviews, it is infeasible to capture the meaningful correlations among different users for collaborative filtering in multi-aspect rating inference focused in this paper. To address this challenge, this paper utilizes an aspect-oriented collaborative filtering method for multi-aspect rating inference by exploring the correlations between ratings across a set of aspects of user opinions.

Snyder and Barzilay [29] presented a Good Grief algorithm to model the dependencies between different aspects. However, in their model only the agreement among all aspects can be captured. Since most of the real reviews used in our evaluation tend to express differing degrees of satisfaction on different aspects, as reported in the experiments, the Good Grief algorithm cannot work well in practice on such a dataset, particularly on a dataset with a severe problem of inconsistent rating annotation. Gupta et al. [12] experimentally evaluated various classification and regression models for multiscale multi-aspect rating inference from textual customer reviews. These previous efforts considered the same feature vector of the unlabeled review to predict ratings of different aspects, while our approaches utilize an aspect-based segmentation algorithm to build an augmented feature vector of each aspect for aspect-based rating inference.

Our previous work [33] has considered the issue of multi-aspect sentence segmentation in Chinese opinion polling tasks with respect to positive and negative polarities instead of a multiscale rating scale. The segmentation algorithm used is implemented based on aspect-related terms learned from unlabeled data using a bootstrapping learning algorithm. The single-aspect textual segments are used as basic units for opinion polling, as done by the SinAsp method. In this work, to effectively capture the information from the results of aspect-based segmentation, we present an aspect-augmentation approach to building the aspect-specific augmented feature vector of each aspect for aspect-based rating inference.

Pang and Lee [21] pointed out two crucial problems for content-based rating inference, including the inconsistent rating annotation problem and mismatching between ratings and the text. However, none of these previous studies have addressed the inconsistent rating annotation problem in content-based rating inference tasks. In practice, supervised classification models and standard ordinal regression algorithms would result in unsatisfactory performance on data with a severely inconsistent rating annotation problem. In this paper, we present a tolerance-based criterion for model training to address the issue of inconsistent rating annotation to improve ordinal regression models for content-based rating inference.

9 Conclusion and Discussion

This paper addresses four issues of multi-aspect content-based rating inference involving aspect-based segmentation, aspect-specific feature vector generation, inconsistent rating inference, and aspect-oriented collaborative filtering. A collaborative rating inference framework is built with the combination of the best of two complementary views for content-based rating inference, i.e., the aspect-oriented collaborative filtering and the PRanking algorithm. Our approaches are easy to implement, and can be applied to other languages such as English or other domains such as product or movie reviews.

In this work, actually the interdependencies between various aspects are exploited in the segmentation and collaborative filtering stages, wherein the output of the segmentation stage is fed into input to the collaborative filtering stage in sequence. While this creates the problem of error propagation, it is worth jointly learning the information used for both stages to improve this pipeline [3]. Besides, in our future work we will also further focus on addressing this issue and how to incorporate the knowledge of rating pairwise ordering to improve ordinal regression models for multi-aspect rating inference.

Acknowledgments

This work was supported in part by the National Science Foundation of China (61073140), Specialized Research Fund for the Doctoral Program of Higher Education (20100042110031), and the Fundamental Research Funds for the Central Universities.

References

[1] J. Basilico and T. Hofmann, "Unifying Collaborative and Content-Based Filtering," *Proc. 21st Int'l Conf. Machine Learning*, pp. 65-72, 2004.

[2] A.L. Berger, A.D.P. Stephen, and J.D.P. Vincent, "A Maximum Entropy Approach to Natural Language Processing," *Computational Linguistics*, vol. 22, no. 1, pp. 39-71, 1996.

[3] R.C. Bunescu, "Learning with Probabilistic Features for Improved Pipeline Models," *Proc. Conf. Empirical Methods in Natural Language Processing*, pp. 670-679, 2008.

[4] K. Crammer and Y. Singer, "Pranking with Ranking," *Proc. Advances in Neural Information Processing Systems,* pp. 641-647, 2001.

[5] K. Dave, S. Lawrence, and D.M. Pennock, "Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews," *Proc. 12th Int'l Conf. World Wide Web,* pp. 519-528, 2003.

[6] H. Daume III, "Frustratingly Easy Domain Adaptation," *Proc. Ann. Meeting Assoc. Computational Linguistics,* pp. 256-263, 2007.

[7] X. Ding, B. Liu, and P.S. Yu, "A Holistic Lexicon-Based Approach to Opinion Mining," *Proc. Int'l Conf. Web Search and Web Data Mining,* 2008.

[8] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification,* second ed. John Wiley & Sons, Inc., 2001.

[9] M. Gamon, A. Aue, S. Corston-Oliver, and E. Ringger, "Pulse: Mining Customer Opinions from Free Text," *Proc. Sixth Int'l Symp. Intelligent Data Analysis,* pp. 121-132, 2005.

[10] D. Goldberg, D. Nichols, B. Oki, and D. Terry, "Using Collaborative Filtering to Weave an Information Tapestry," *Comm. ACM,* vol. 35, pp. 61-70, 1992.

[11] A.B. Goldberg and X. Zhu, "Seeing Stars When There Aren't Many Stars: Graph-Based Semi-Supervised Learning for Sentiment Categorization," *Proc. HLT/NAACL First Workshop Graph Based Methods for Natural Language Processing,* pp. 45-52, 2006.

[12] N. Gupta, G. Di Fabbrizio, and P. Haffner, "Capturing the Starts: Predicting Ratings for Service and Product Reviews," *Proc. NAACL HLT Workshop Semantic Search,* pp. 36-43, 2010.

[13] E.F. Harrington, "Online Ranking/Collaborative Filtering Using the Perceptron Algorithm," *Proc. 20th Int'l Conf. Machine Learning,* 2003.

[14] M. Hu and B. Liu, "Mining Opinion Features in Customer Reviews," *Proc. 19th Nat'l Conf. Artificial Intelligence,* 2004.

[15] M. Hu and B. Liu, "Mining and Summarizing Customer Reviews," *Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 168-177, 2004.

[16] S.Y. Jung, J.H. Hong, and T.S. Kim, "A Statistical Model for User Preference," *IEEE Trans. Knowledge and Data Eng,* vol. 17, no. 6, pp. 834-843, June 2005.

[17] S-M. Kim and E. Hovy, "Extracting Opinions, Opinion Holders, and Topics Expressed in Online News Media Text," *Proc. ACL/COLING Workshop Sentiment and Subjectivity in Text,* 2006.

[18] N. Kobayashi, K. Inui, and Y. Matsumoto, "Extracting Aspect-Evaluation and Aspect-of Relations in Opinion Mining," *Proc. Joint Conf. Empirical Methods in Natural Language Processing and Computational Natural Language Learning,* pp. 1065-1074, 2007.

[19] C.W.-K. Leung, S.C.-F. Chan, and F.-L. Chung, "Integrating Collaborative Filtering and Sentiment Analysis: A Rating Inference Approach," *Proc. European Conf. Artificial Intelligence Workshop Recommender Systems,* pp. 62-66, 2006.

[20] B. Liu, M. Hu, and J. Cheng, "Opinion Observer: Analyzing and Comparing Opinions on the Web," *Proc. 14th Int'l Conf. World Wide Web,* pp. 342-351, 2005.

[21] B. Pang and L. Lee, "Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales," *Proc. 43rd Ann. Meeting Assoc. for Computational Linguistics,* pp. 115-124, 2005.

[22] B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis," *Foundations and Trends in Information Retrieval,* vol. 2, nos. 1/2, pp. 1-135, 2008.

[23] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs Up? Sentiment Classification Using Machine Learning Techniques," *Proc. Conf. Empirical Methods in Natural Language Processing,* 2002.

[24] L. Pevzner and M.A. Hearst, "A Critique and Improvement of an Evaluation Metric for Text Segmentation," *Computational Linguistics,* vol. 28, no. 1, pp. 19-35, 2002.

[25] A.M. Popescu and O. Etzioni, "Extracting Product Features and Opinions from Reviews," *Proc. Conf. Empirical Methods in Natural Language Processing,* 2005.

[26] G. Qiu, B. Liu, J. Bu, and C. Chen, "Expanding Domain Sentiment Lexicon through Double Propagation," *Proc. 21st Int'l Jont Conf. Artificial Intelligence,* pp. 1199-120, 2009.

[27] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An Open Architecture for Collaborative Filtering of Netnews," *Proc. ACM Conf. Computer Supported Cooperative Work,* pp. 175-186, 1994.

[28] E. Riloff, S. Patwardhan, and J. Wiebe, "Feature Subsumption for Opinion Analysis," *Proc. Conf. Empirical Methods in Natural Language Processing,* pp. 440-448, 2006.

[29] B. Snyder and R. Barzilay, "Multiple Aspect Ranking Using the Good Grief Algorithm," *Proc. Human Language Technology Conf. North Am. Chapter of the Assoc. Computational Linguistics,* pp. 300-307, 2007.

[30] M. Thomas, B. pang, and L. Lee, "Get Out the Vote: Determining Support or Opposition from Congressional Floor-Debate Transcripts," *Proc. Conf. Empirical Methods in Natural Language Processing,* pp. 327-335, 2006.

[31] I. Titov and R. McDonald, "A Joint Model of Text and Aspect Ratings for Sentiment Summarization," *Proc. Ann. Meeting Assoc. Computational Linguistics,* pp. 308-316, 2008.

[32] T. Zagibalov and J. Carroll, "Unsupervised Classification of Sentiment and Objectivity in Chinese Text," *Proc. Third Int'l Joint Conf. Natural Language Processing,* pp. 304-311, 2008.

[33] J. Zhu, H. Wang, M. Zhu, and B.K. Tsou, "Aspect-Based Opinion Polling from Customer Reviews," *IEEE Trans. Affective Computing,* vol. 2, no. 1, pp. 37-49, Jan.-Mar. 2011.

Jingbo Zhu received the PhD degree in computer science from Northeastern University, Shenyang, China, in 1999. He has been with the Institute of Computer Software and Theory, Northeastern University, since 1999. Currently, he is a full professor in the Department of Computer Science and is in charge of research activities within the Natural Language Processing Laboratory. He was a visiting scholar at ISI, University of Southern California, Los Angeles, from 2006 to 2007. He has published more than 150 papers and holds four US patents. His current research interests include syntactic parsing, machine translation, sentiment analysis, and machine learning for natural language processing. He is a member of the IEEE.

Chunliang Zhang received the BS degree in mining construction engineering from Northeastern University, Shenyang, China, in 1994 and the MA degree in English language and literature from Jilin University, Changchun, in 2003. He has been working toward the PhD degree in natural language processing since 2010. Presently, he is an associate professor in the College of Foreign Language Studies at Northeastern University. His current research interests include textual data annotation, sentiment analysis, and machine translation.

Matthew Y. Ma received the BE degree in electrical engineering from Tsinghua University, Beijing, China, the MS degree in electrical engineering from the State University of New York at Buffalo, and the PhD degree in electrical and computer engineering from Northeastern University, Boston, Massachusetts. He is currently with Scientific Works, Princeton Junction, New Jersey. Prior to that, he had 11 years tenure as a senior scientist at Panasonic R&D Company of America focusing on mobile and imaging research and two other firms focusing on patent research and strategy. He is the inventor of 14 granted US patents and is the author of approximately 30 conference and journal publications. He has published two books in the field of pattern recognition: *Personalization and Recommender Systems* (World Scientific, 2008) and *Mobile Multimedia Processing: Fundamentals, Methods and Applications* (Springer, 2010). He has also authored *Fundamentals of Patenting and Licensing for Scientists and Engineers* (World Scientific, 2009). He is an associate editor of the *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)* and a US patent agent admitted to the US Patent Trademark & Patent Office. He has been an affiliated professor at Northeastern University, Shenyang, China, since 2001, and an invited lecturer at the University of California, Berkeley, and Tsinghua University for a "Patent Strategy and Innovation" course in 2011 and 2012, respectively. His primary research interests include patent analytics research, image analysis, pattern recognition, and natural language processing. He is a senior member of the IEEE.