A LATENT ASPECT MINING FRAMEWORK FROM TEXTUAL REVIEWS

Duc-Hong Pham
Faculty of Information Technology, Electric Power University, Hanoi, Vietnam
hongpd@epu.edu.vn

Abstract - The problem of latent aspect mining from textual customer reviews is important in knowledge discovery and natural language processing. Given a collection of review texts, it is required to automatically determine aspect ratings for each textual review and identify important aspects for all textual reviews. Existing works on aspect-based sentiment analysis has used deep learning techniques with architectures designed based on neural networks. However, they take a lot of time to learn models and need computer configuration to be big enough. This paper proposes a framework (which contains three sub-tasks: (1) aspect term extraction, (2) aspect category detection, (3) aspect ratings and important aspects determination) using simpler and more efficient techniques based on constrained-KMeans algorithm and word2vec. In experiment, we use a data set of 174615 reviews of 1768 hotels with five common aspects including cleanliness, location, service, room and value. Experimental results show that aspects represented by averaging word vectors are more effective than represented by a bag of word. In general, our model outperforms some other benchmark algorithms.

***Keywords*:** Aspect based sentiment analysis; Aspect rating; Aspect weight; Aspect term extraction; Aspect category detection.

1. Introduction

Derived from the customer textual review dataset on online commerce websites and in order to help users easily find the necessary information as well as to understand the evaluation opinions on the textual reviews. Some works of Aspect-based Sentiment Analysis (ASA) have been done, such as aspect terms extraction [1, 2, 3], aspect category detection [4, 5, 6], aspect based sentiment classification [7, 8], aspect ranking [9, 10, 11]. Most previous studies have used a bag of words to extract features. They ignore the semantic relations between words, and cause an inaccuracy of opinion predictions. Recently, deep learning models with architectures designed based on neural networks have applied in many studies of ASA, such as [12, 13]. However, they take a lot of time and effort to build an effective deep learning model. In this paper, we propose a simple framework using word2vec and constrained-KMeans algorithm for discovering latent aspects, containing three sub-tasks: (1) Aspect terms extraction; (2) Aspect category detection; (3) Discovering aspect ratings and overall aspect weights. To address the first task, we propose a semi-supervised learning algorithm based on constrained-KMeans algorithm to extract aspect terms. Then, from the results achieved, we propose an iterative algorithm to aspect category detection. For the third task, we propose a neural networks model to generate the overall rating of each textual review with aspect ratings are assumed latent at the hidden layer and overall aspect weights are weights between the hidden layer and the output layer.

We evaluate our proposed framework on the data collected from Tripadvisor.com and use the five aspects including *Value*, *Room*, *Location*, *Cleanliness*, and *Service*. This dataset is also used in previous research [19, 20]. The experimental results have shown the effectiveness of the proposed method in comparison with other methods.

2. Related works

Aspect terms extraction: [14] a neural network containing 7-layer deep convolutional to labeling each word in textual sentences as aspect word or non aspect word. [15] used linguistic rules to identify nominal phrase chunks and regard them as candidate opinion target labels and aspects. Then they propose to extract irrelevant candidates based on domain correlation. Finally, they use these texts with extracted chunks as pseudo labeled data to train a deep gated recurrent unit (GRU) network for aspect term extraction and opinion target extraction. [16] proposed a bidirectional dependency tree network model to find dependency structure features from the given textual sentences. The key idea is to explicitly incorporate both representations gained separately from the bottom-up and top-down propagation on the given dependency syntactic tree. An end-to-end framework is then developed to integrate the embedded representations and BiLSTM plus CRF to learn both tree-structured and sequential features to solve the aspect term extraction problem.

Aspect category detection: [17] a deep learning model to automatically learn useful features for aspect category detection. They first proposed a semi-supervised word embedding algorithm to obtain continuous word representations, then they proposed to generate deeper and hybrid features through neural networks stacked on the word vectors. In our work, we use the obtained aspect term list in the task of aspect term extraction and word vectors are learned from word vector model [23] to detect the aspect category for each sentence. [18] proposed a deep neural network method based on attention mechanism to identify different aspect categories of a given review sentence. They utilizes several attentions with different topic contexts, enabling it to attend to different parts of a review sentence based on different topics.

Discovering aspect ratings and aspect weights: [19] a latent aspect rating analysis model to infer aspect ratings and aspect weights for each review. In [20] proposed a sparse aspect coding model, by considering the user and item side information of review texts, they use two latent variables named as user intrinsic aspect interest and item intrinsic aspect to discover a set of aspects which are previously unknown for each aspect and predict the users rating on each aspect for each review. However these works cannot directly compute overall aspect weights for all textual reviews from their model. Another limitation of these models is to use a bag of words model for representing aspects thus may fail to capture semantic relations between words and cause an inaccuracy of aspect ratings prediction. In our work, we propose a neural networks with assuming that aspect ratings are latent at the hidden layer and overall aspect weights are weights between the hidden layer and the output layer. For the input for our model, we use the computed way of feature representation in [21] that for an aspect in a review, we an average of the word representations for all words present in text of aspect which can capture semantic relations between words. A probabilistic aspect ranking algorithm [22] to identify the importance of aspects by simultaneously considering aspect frequency and the influence of consumer opinions given to each aspect over their overall opinions. This algorithm is provided aspect ratings in input and it infers the aspect weights for each individual review. Then, they compute the overall aspect weights (i.e. important aspects) for all reviews by averaging aspect weights of all reviews. In our work, aspect ratings are'nt provided in input.

3. Our framework

In this section, we first present the problem definition with necessary notations, which will be used in our proposed framework. We then describe the framework including sub-tasks need to be addressed. Finally, we introduce word2vec model and the Constrained-KMeans algorithm, and then we present the proposed method or model for each sub-task.

3.1. *Problem definition*

Given a collection of textual reviews $D = \{\mathrm{d}_1, \mathrm{d}_2, ..., \mathrm{d}_{|D|}\}$ for an interesting entity or topic. For each textual review associated with a numerical overall rating. We assume all aspects using the same dictionary which contains *n* words and is denoted, $V = \{\omega_1, \omega_2, ..., \omega_n\}$, each word $\omega \in V$ is represented by an *m* dimensional vector $v(\omega), (m < n)$.

Let $A = \{A_1, A_2, ..., A_k\}$ be a set of *k* – aspects in *D*, where an aspect A_i is a set of terms that characterize a rating factor in the reviews. We build the dictionary V and identify word vectors in the task of aspect term extraction (in Section 4.2). We identify $\{A_1, A_2, ..., A_k\}$ for aspects in the task of aspect category detection and represent aspect.

We denote the overall aspect weights for all textual review as a *k* - dimensional vector $\alpha = (\alpha_1, \alpha_2, ..., \alpha_k)$, where the *i*-th dimension is a numerical measure, indicating the degree of importance aspect A_i. For each textual review $d \in D$, the overall rating is denoted by O_d, it is given by the user and indicating levels of overall opinion of *d*. Aspect ratings for textual review *d* is a *k* - dimensional vector $r_d = (r_{d1}, \mathrm{r}_{d2}, ..., \mathrm{r}_{dk})$, where the *i*-th dimension is a numerical measure, indicating the degree of satisfaction demonstrated in the review $d \in D$ toward the aspect A_i. We represent aspects for review d as feature matrix $\mathrm{X}_d = (\mathrm{x}_{d1}, \mathrm{x}_{d2}, ..., \mathrm{x}_{dk})$, where each column x_{di} is represented by averaging word vector of aspect A_i on textual review *d*.

Both α for all textual reviews and aspect ratings r_d for each textual review $d \in D$ are unknown, our goal is to discover them from the set given of textual reviews.

3.2. *Our framework description*

In this section, we will describe our framework, it is a new direction in sentiment analysis research. The goal of our framework is to discover aspect ratings for each textual review as well as identify overall aspect weights for all textual reviews. For this purpose, three sub-tasks need to be addressed: (1) aspect term extraction, (2) aspect category detection, (3) discovering aspect rating and (4) overall aspect weight. We illustrate these tasks such as in Figure 1 and describe them as follows:

(1) **Aspect term extraction**: Given a set of textual reviews with pre-identified entities (e.g., hotels, restaurants), each textual review including some sentences. We need to identify the aspect terms present in each sentence and then mix aspect terms into the set of aspects. For example, "*The rooms were great and every time we left and came back it was cleaned*". Term words are extracted such as "*great*", "*rooms*", and "*cleaned*" into the term list of aspect "Room".

(2) **Aspect category detection**: Given a predefined set of aspect categories (e.g. *room*, *service*), for each sentence in a textual review, we attempt to identify the aspect category discussed in its text. For example, *We stayed in a junior suite on ground level and it was spotlessly clean, great maid service and room service on tap 24/7.* We can identify the aspect category discussed in this sentence is *Service.*

(3) **Discovering aspect rating and overall aspect weight**: Based on the results performed in the sub task 1 and in the sub task 2, we attempt to discover aspect ratings for each review. In Figure 2, we show an example of an input-output of our framework. The input is a textual review with only the overall ratings is given, the output are ratings of aspects which we need to discover. In addition, in the process of discovering aspect ratings for each review, we also attempt to identify the overall aspect weights for all reviews. The task identifies the overall aspect weights are very important and it is viewed as identify important aspects from the given set of textual reviews.

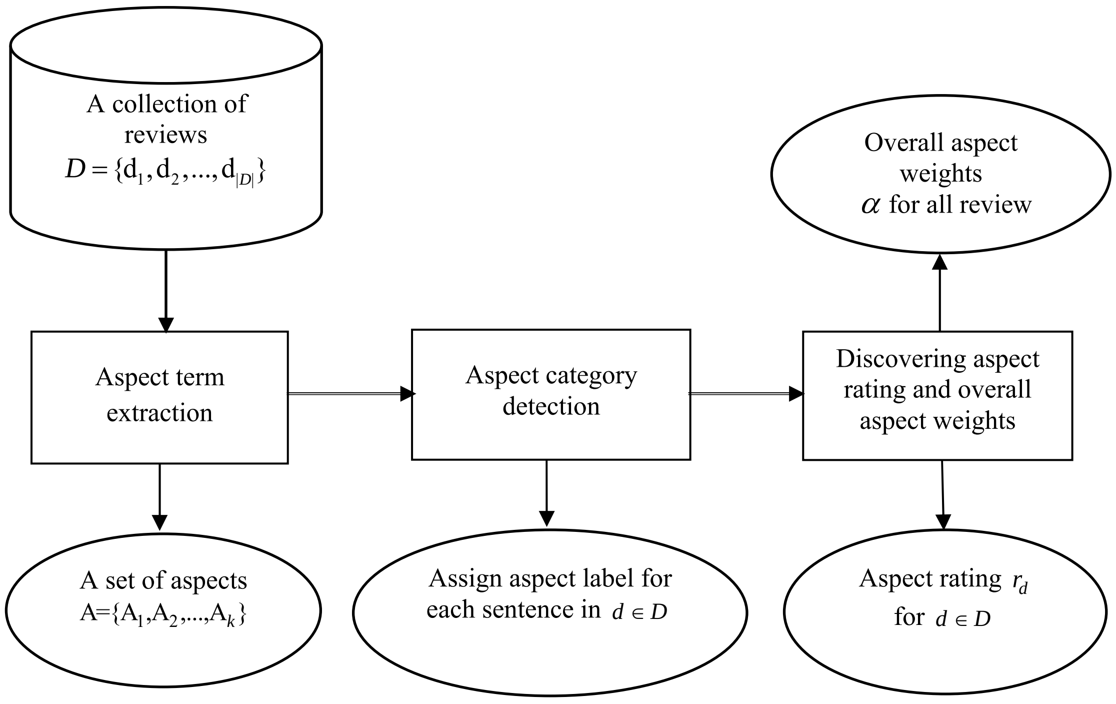
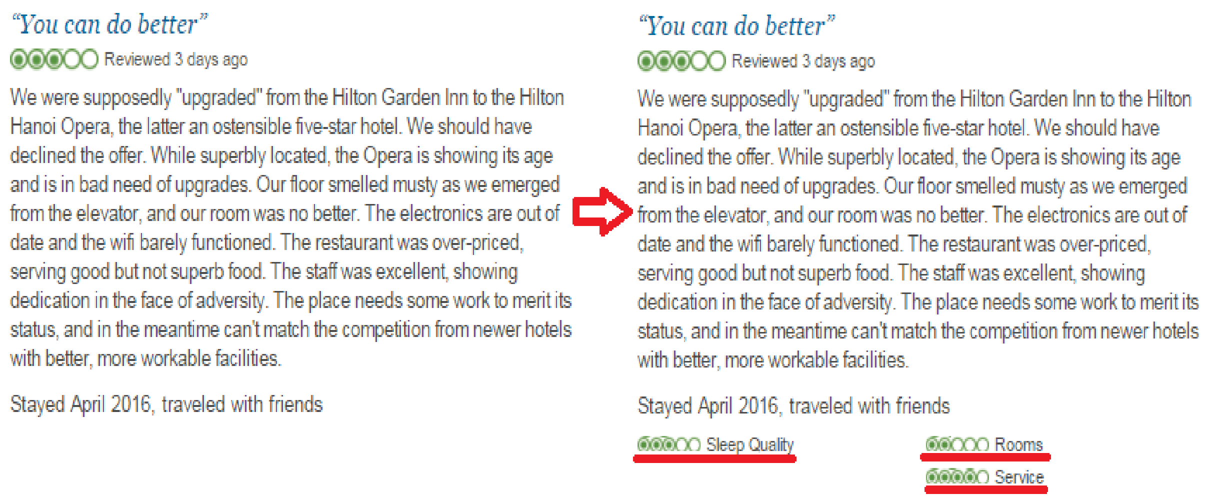

A collection of reviews $D=\{d_1,d_2,...,d_{|D|}\}$
Overall aspect weights α for all review
Aspect term extraction
Aspect category detection
Discovering aspect rating and overall aspect weights
A set of aspects $A=\{A_1,A_2,...,A_k\}$
Assign aspect label for each sentence in $d \in D$
Aspect rating r_d for $d \in D$

Fig. 1. An illustration of the latent aspect mining framework with three sub-tasks are illustrated in rectangles and the result corresponding with them are in ovals


"You can do better"
Reviewed 3 days ago

We were supposedly "upgraded" from the Hilton Garden Inn to the Hilton Hanoi Opera, the latter an ostensible five-star hotel. We should have declined the offer. While superbly located, the Opera is showing its age and is in bad need of upgrades. Our floor smelled musty as we emerged from the elevator, and our room was no better. The electronics are out of date and the wifi barely functioned. The restaurant was over-priced, serving good but not superb food. The staff was excellent, showing dedication in the face of adversity. The place needs some work to merit its status, and in the meantime can't match the competition from newer hotels with better, more workable facilities.

Stayed April 2016, traveled with friends

"You can do better"
Reviewed 3 days ago

We were supposedly "upgraded" from the Hilton Garden Inn to the Hilton Hanoi Opera, the latter an ostensible five-star hotel. We should have declined the offer. While superbly located, the Opera is showing its age and is in bad need of upgrades. Our floor smelled musty as we emerged from the elevator, and our room was no better. The electronics are out of date and the wifi barely functioned. The restaurant was over-priced, serving good but not superb food. The staff was excellent, showing dedication in the face of adversity. The place needs some work to merit its status, and in the meantime can't match the competition from newer hotels with better, more workable facilities.

Stayed April 2016, traveled with friends

Sleep Quality
Rooms
Service

Fig. 2. An example of an input-output with the input on the left, and the output on the right. In which ratings of aspects: "*Sleep Quality*", "*Rooms*" and "*Service*" are discovered and assign them as numerical stars in the output

4. The proposed method

4.1. *Aspect term extraction and aspect category detection*

In this section, we first introduce the word vector model and the semi-supervised learning constrained-KMeans algorithm, then we propose a semi-supervised learning algorithm based on constrained-KMeans algorithm to extract aspect term for each aspect. The final, based on word vectors learned from the word vector model and the obtained list of terms of aspects, we propose a aspect category detection algorithm for each textual review.

4.1.1. Word vector model

Word vector model takes textual sentences in documents/reviews as input, and produces the representation for each word, many word vector models have been proposed such as in [24, 25, 26]. In this paper, we apply a simple and effective method [23], which contains two architectures named as the continuous bag of words (CBOW) model and the skip-gram model to learn word vectors.

4.1.2. Constrained-KMeans algorithm

The constrained-KMeans algorithm [27] is very effective in cluster data. Let $V = \{\omega_1, \omega_2, ..., \omega_n\}$ be a set of points and given a set of seed points $S = \{\mathrm{S}_1, \mathrm{S}_2, ..., \mathrm{S}_k\}$ of *k*-clusters, the clustering problem requests split V into k-clusters. Unlike the traditional KMeans algorithm, this algorithm uses the seed clustering to initialize the KMeans algorithm. Thus, rather than initializing KMeans from *k* random means. We compute the h-th mean to initialize for the h-th cluster as $\mu_h \leftarrow \frac{1}{|S_h|} \sum_{\omega \in S_h} v(\omega), h = 1, 2, ..., k$. The seed clustering is only used for initialization, and the seeds are not used in the following steps of the algorithm. In the sub-sequent steps, the cluster memberships of the data points in the seed set are not recomputed in the assign-cluster steps of the algorithm - the cluster labels of the seed data are kept unchanged, and only the labels of the non-seed data are re-estimated.

4.1.3. Aspect term extraction and aspect category detection

In order to extract aspect term, we first split each review in all given reviews into sentences, then we use all sentences as input for word vector model (i.e. CBOW model or Skip-Gram model) and we use this model to learn word vector representations for words. Denoted $V = \{\omega_1, \omega_2, ..., \omega_n\}$ be a dictionary include all words presented as word vectors learned from the word vector model. Next, based on the dictionary V, we attempt to identify terms for each aspect as follows: Assuming that each aspect is provided a few seed terms, we propose using a semi-supervised constrained-KMeans algorithm to cluster words in V into *k*-clustering words and then corresponding to each clustering word is a terms set of a specified aspect. Specifically, given the seed terms $S = \{\mathrm{S}_1, \mathrm{S}_2, ..., \mathrm{S}_k\}$ of k - aspects, and the dictionary V, we propose the algorithm to extract aspect term as Algorithm 1 in which we additional use the choosing threshold p.

```
Algorithm 1: Aspect term extraction
Input: $V=\{\omega_1,\omega_2,...,\omega_m\}$, $S=\{S_1,S_2,...,S_k\}$, choosing threshold $p$
Step 0: Initialize $\mu_h^{(0)} \leftarrow \frac{1}{|S_h|}\sum_{\omega\in S_h} v(\omega)$, for $h=1,2,...,k; t\leftarrow 0$
Step 1: Repeat until convergence
    1.1. assign aspect: For each $\omega\in V$, if $\omega\in S_h$ assign $\omega$ to the aspect $h$ (i.e. push it into set $A_h^{(t+1)}$).
         For $\omega\notin S_h$, if $sim(v(\omega),\mu_h^{(t)}) < p$ and $h^* = \arg\max_h sim(v(\omega),\mu_h^{(t)})$ then assign $\omega$
         to the aspect $h^*$ (i.e. push it into set $A_h^{(t+1)}$)
    1.2. estimate means: $\mu_h^{(t+1)} \leftarrow \frac{1}{|A_h^{(t+1)}|}\sum_{\omega\in A_h^{(t+1)}} v(\omega)$
    1.3. $t\leftarrow(t+1)$
Output: $A=\{A_1,A_2,...,A_k\}$
```

For aspect category detection, we attempt to map each sentence in a review into a specified aspect. From the dictionary V with word vectors learned from the word vector model and the list of terms of aspects $A=\{A_1,A_2,...,A_k\}$ obtained in the algorithm 1, we use them to detect aspect category for each sentence in reviews. Specifically, given a collection of reviews $\{d_1,d_2,...,d_{|D|}\}$, for a review d, we split review d into a set of sentences $S=\{s_1,s_2,...,s_{|S|}\}$, after for a sentence $s_i\in S$, we average all the word vectors of words in its text, we denote this average vector as μ_{s_i}. Then, we compute the cosine similarity between the average vector μ_{s_i} and the average vector of each aspect, we assign the sentence s_i to the aspect that the maximum cosine similarity with this sentence. The algorithm detects aspect category is proposed as Algorithm 2.

```
Algorithm 2: Aspect category detection
Input: A collection of textual reviews $D=\{d_1,d_2,...,d_{|D|}\}$, $A=\{A_1,A_2,...,A_k\}$, choosing threshold $p$
Step 0: $\mu_{A_h} \leftarrow \frac{1}{|A_h|}\sum_{\omega\in A_h} v(\omega)$, h=1, 2,..., k
Step 1: for each $d\in D$ do
    1.1. Split review $d$ into a set of sentences $S=\{s_1,s_2,...,s_{|S|}\}$
    1.2. $\mu_{s_i} \leftarrow \frac{1}{|s_i|}\sum_{\omega\in s_i} v(\omega)$, for i=1,2,...,|S| assign aspect : if $(sim(\mu_h,\mu_{s_i}) < p)$ and
         $h^* = \arg\max_h sim(\mu_h,\mu_{s_i})$ then assign $s_i$ to the aspect $h^*$
Output: All sentences in each review $d\in D$ are assigned to the aspect
```

4.2. *Discovering aspect ratings and overall aspect weights*

In this section, we apply a neural network model to discover aspect ratings for each review as well as overall aspect weights for all reviews.

We assume both aspect ratings and overall aspect weights to be latent in a neural network model and call this model as Latent rating mining neural network model (LRMNN). In Figure 3 Shows the architecture of the LRMNN model.

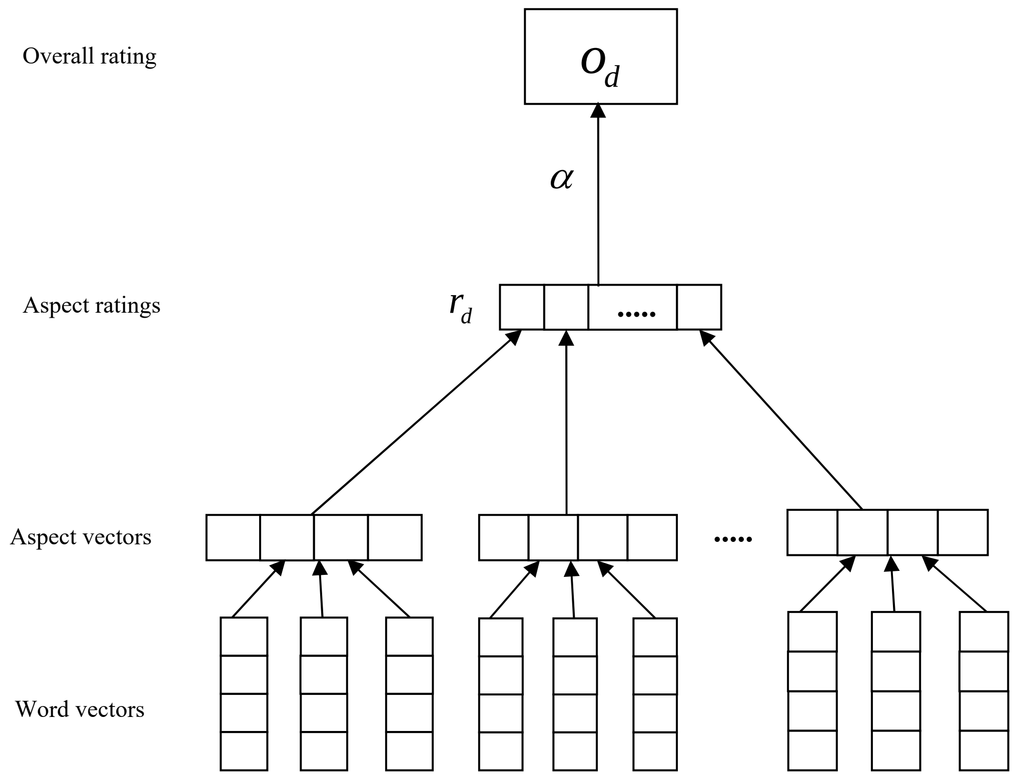


Fig. 3. An illustration of the LRMNN model discovers aspect ratings and overall aspect weights for all reviews, in which are units at the hidden layer and are the weights between the hidden layer and the output layer.

For a review d , we compute aspect feature vector x_{di} of aspect A_i by averaging word vectors for words in its text as follows:

$$x_{di} = \frac{\sum_{p=1}^{Q} v_p}{\left|\sum_{p=1}^{Q} v_p\right|} \tag{1}$$

where Q is the number of words, v_p is the vector representation of ω_p in the text assigned to aspect A_i , and $|y|$ means the $L1-norm$ of $|y|$

Let $\mathrm{w}_i = (\mathrm{w}_{i1}, \mathrm{w}_{i2}, ..., \mathrm{w}_{in})$ be a weight vector of aspect A_i . Then, the aspect rating r_{di} is generated based on a linear combination of aspect vector feature and its weight vector as $r_{di} \sim \sum_{l=1}^{n} x_{dil}.\mathrm{w}_{il}$ [19]. Specifically, we assume that aspect rating is generated at the hidden layer of the neural network and compute it by:

$$r_{di} = \mathrm{sigm}(\sum_{l=1}^{n} \mathrm{x}_{dil}\mathrm{w}_{il} + \mathrm{w}_{i0}) \tag{2}$$

Where $\mathrm{sigm}(y) = 1/(1+e^{-y})$, w_{i0} is a bias term.

The overall aspect weights for all review are the weights between the hidden layer and the output layer. The overall rating is generated at the output layer of the neural network and computed based on the weighted sum of a and r_d as follows:

$$\hat{O}_d = \sum_{i=1}^{k} r_{di}\alpha_i \tag{3}$$

subject to $\sum_{i=1}^{k}\alpha_i = 1$, $0 \le \alpha_i \le 1$ for $i = 1, 2, \ldots, k$

Suppose $\sum_{i=1}^{k}\alpha_i = 1$ and $0 \le \alpha_i \le 1$ instead of the overall aspect weight α_i by the auxiliary overall aspect weight $\overset{\wedge}{\alpha_i}$ as follows:

$$a_i = \frac{\exp(\overset{\wedge}{\alpha_i})}{\sum_{l=1}^{k}\exp(\overset{\wedge}{\alpha_l})} \tag{4}$$

The equation (4) becomes a equation as follows:

$$\overset{\wedge}{O_d} = \sum_{i=1}^{k} r_{di}\frac{\exp(\overset{\wedge}{\alpha_i})}{\sum_{l=1}^{k}\exp(\overset{\wedge}{\alpha_l})} \tag{5}$$

Let O_d be the desired target values of the overall rating of review d, the cross entropy cost function for the review d is as follows:

$$C_d = -O_d \log \overset{\wedge}{O_d} - (1 - O_d)\log(1 - \overset{\wedge}{O_d}) \tag{6}$$

The cross entropy error function (CEE) for the data set $\mathrm{D}=\{(\mathrm{r}_d, O_d)\}_{d=1}^{|D|}$ is,

$$E(\mathrm{w}, \overset{\wedge}{\alpha}) = \sum_{d\in\mathrm{D}} C_d = -\sum_{d\in\mathrm{D}}(O_d \log \overset{\wedge}{O_d} + (1 - O_d)\log(1 - \overset{\wedge}{O_d})) \tag{7}$$

To avoid over fitting and no loss of generality, we add regularizers to the $E(\mathrm{w}, r, \overset{\wedge}{\alpha})$. Regularization terms are ubiquitous. They typically appear as an additional term in an optimization problem

$$E(\theta) = -\sum_{d\in\mathrm{D}}(O_d \log \overset{\wedge}{O_d} + (1 - O_d)\log(1 - \overset{\wedge}{O_d}))) + \frac{1}{2}\lambda\|\theta\|^2 \tag{8}$$

Where $\mathrm{W} = (\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{k})$, $\mathrm{w}_0 = (\mathrm{w}_{01}, \mathrm{w}_{02}, \ldots, \mathrm{w}_{0k})$, and $\theta = [\mathrm{W}, \overset{\wedge}{\alpha}]$ is a set of the model parameters, λ is the regularization parameter and $\|\theta\|^2 = \sum_i \theta_i^2$ is a norm regularization term. In order to compute the parameters θ, we apply back-propagation algorithm with stochastic gradient descent to minimize this cost function. Each element of the weights in the parameters θ is updated at time *t + 1* as :

$$\theta(t+1) = \theta(t) - \eta\frac{\partial E(\theta)}{\partial\theta} \tag{9}$$

where η is the learning rate.

The steps of the algorithm discover aspect ratings and overall aspect weights as in Algorithm 3.

Algorithm 3: Discovering aspect ratings and aspect weights for each review

```
Input: A collection of reviews $\{d_1, d_2, ..., d_{|D|}\}$, each review $d$ is assigned overall rating $O_d$, the learning
rate $\eta$, the error threshold $\varepsilon$, the iterative threshold $I$ and the regularization parameter $\lambda_1$, $\lambda_2$
Step 0: $t=0$;
        Initialize $W$, $w_0$, $\hat{\alpha}$
        for each review $d \in D$,
        calculate each $x_{di} \in X_d$ According to Eq. (1);
Step 1: for $iter$=0 to $I$ do
         for each pair $(X_d, O_d) \in D$ do
            1.1. Calculate $\alpha_i$ According to Eq. (4);
            1.2. Calculate $r_{di}$ at time $t$ at hidden layer According to Eq. (2);
            1.3. Calculate $\hat{O}_d$ at time $t$ at output layer According to Eq. (3);
            1.4. Update parameters in $\theta$ at time $t+1$ using Eq. (8);
Step 2: For offline learning, the Step 1 may be repeated until the iteration error $\frac{1}{|D|}\sum_{d \in D}\left|O_d - \hat{O}_d(t)\right|$ is
less than the error threshold or the number of iterations have been completed.
Output: $W$, $w_0$, $\hat{\alpha}, R$
```

After obtaining W, w_0, and $\hat{\alpha}$ for review d, we compute each aspect rating r_{di} according to Eq. (2). For overall aspect weights for all reviews, we compute each $\alpha_i \in \alpha$ according to Eq. (4).

5. Experiments

In this section, we first describe a real data set to empirically validate the our proposed framework. We then show some typical results achieved and use metrics to compare our proposed method with other methods.

5.1. *Data set*

We use a real data set contain 174,615 reviews of 1,768 hotels, where reviews are provided associated with overall ratings as well as with ground truth aspect ratings on 5 aspects: *Value*, *Rooms*, *Location*, *Cleanliness*, and *Service*. All the ratings in this data set are in the range from 1 star to 5 stars.

We conduct pre-processing on this data set as follows: remove stop words and the words convert into lower cases. Stemming to remove the differences between inflected forms of a word, in order to reduce each word to its root form. After processing the data, we split reviews into sentences and be determined 2,126,919 sentences. We also built a dictionary with 29,349 words (denote it as V), which the frequency of each word in all reviews is no less than 10. Table 1 show statistical data in our experiments.

Table 1. Evaluation Data Statistics

Number of reviews	174615
Number of hotels	1768
Number of aspects	5
Number of sentences	2126919
Number of words in dictionary	29349

5.2. *Word vectors*

To obtained word vectors for words in the given data, we use the continuous bag-of-words architecture of Word2Vec with the window size of context is 7, the word frequency threshold is 10 and the size of word vector is 200 to learn word vector for each word.

5.3. *Experimental result*

Based on the information about aspects provided in the data set, we aim at predicting aspect ratings corresponding to ground-truth aspect ratings. To support this task we select a few seed words (i.e. as in Table 2) for each aspect and use it to initialize input for our aspect term extraction algorithm.

Table 2. Seed words given as initial input

Aspect	Seed words
Values	value, price, money, usd
Rooms	rooms, room, space, bed
Location	location, centre, taxi, bus
Cleanliness	clean, dirty, filthy, damp
Service	service, manager, food, breakfast

We perform the aspect term extraction algorithm with input are word vectors in V and seed words of aspects to identify the full list of keywords for each aspect. In Table 3, we show some the detected key terms of 5 aspects.

Table 3. The detected key terms of some aspects

Value	**Rooms**	**Location**	**Cleanliness**	**Service**
gratuity	pilled	zeil	permeated	medallions
steal	pillow	zoologischer	sewage	tex-mex
tariff	suit	harbourfront	stale	steak
wattage	sandpaper	two-minute	odour	sakura
26/day	boxsprings	down-town	stank	japaneese
negotiated	stain	15-20mins	odors	agave
charges	cushion	urgell	cigarette	oceania
charged	screws	one-minute	doelen	guinea
overcharged	serta	clot	disgusting	itallian
30/day	sagged	tiergarten	non-smoking	hibachi

We perform the Algorithm 2 to detect aspect category for each review and we then mix words of the sentences of the same aspect, the word vectors of these words are used as input for the Algorithm 3. The Algorithm 3 is performed with initialize the learning rate $\eta = 0.015$, the error threshold $\varepsilon = 10^{-4}$, the iterative threshold I=1000, the regularization $\lambda = 10^{-3}$. In Table 4, we show the aspect rating determining for five hotels with the same mean (average) overall rating as 3.5 which we randomly select from our results achieved, note that the ground-truth aspect ratings in parenthesis. We can see that the result of aspect rating prediction very close to the value rating in the ground-truth aspects.

Table 4. The aspect rating discover for five hotels

Hotel Name	**Values**	**Rooms**	**Location**	**Cleanliness**	**Service**
Giada Hotel	3.7(3.8)	3.1(3.4)	4.0(4.5)	4.5(3.9)	3.8(3.5)
Sivory Punta Cana Boutique Hotel	3.3(3.3)	4.3(3.8)	3.6(3.6)	4.2(4.0)	3.7(4.1)
The Hotel California A Piece of Pineapple Hospitality	3.3(3.7)	3.2(3.6)	3.9(3.9)	3.8(4.0)	3.8(3.8)
Hotel Brunelleschi	3.4(3.3)	3.3(3.2)	4.1(4.7)	3.5(3.5)	3.9(3.5)
Hotel Machiavelli Palace	3.6(3.7)	3.4(3.4)	3.9(4.3)	4.0(3.9)	3.8(3.6)

In Table 5, we show the results of overall aspect weights. We can see the importance of each aspect, for example, the results here indicate that the aspects “Service”, “Rooms” are the most important aspects. This information is valuable to the hotel manager because it can help them to know which aspects are important to customers.

Table 5. Overall aspect weight of each aspect

Aspects	Overall weights
Values	0.124
Rooms	0.305
Location	0.167
Cleanliness	0.039
Service	0.365

5.4. *Evaluation*

Because the given data have not the ground-truth aspect term as well as have not the ground-truth aspect category, therefore, we can'nt directly evaluate the results of the Algorithm 1 and the Algorithm 2. For the Algorithm 3, we evaluate the results of aspect ratings prediction in two cases: (1) aspect features are represented by bag of words, we use the dictionary contains 3,941 words; (2) aspect features are represented based on word vectors, we use the word vectors of words in text part of each aspect as inputs in a aspect features-level and compute the aspect feature representation as Eq. (1).

Other models are compared our model LRMNN as follows:

Global Prediction [28] : Using the global information of the overall ratings, each word is assigned a label as a rating value. After they classify each word into different rating values, the rating for each aspect is calculated by aggregating the rating of the words. However, this method does not infer aspect weights for reviews.

LRR [19]: A method using probabilistic rating regression model to infers aspect weights and aspect ratings for each review.

Prank [29]: We apply this algorithm to identify aspect ratings, but it is a fully supervised algorithm, it uses the aspect ratings in the ground-truth for training phase.

LSAWs [30]: A least square based model for Identifying the overall aspect weights.

5.4.1. *Evaluation on aspect rating*

Let D_{test} be a set of test data, we use the four metrics for evaluating aspect rating prediction, (1) root mean square error on aspect rating prediction (we denote it as Δ_{aspect}, lower mean better performance), (2) aspect correlation inside reviews [19] (P_{aspect}, higher mean better), (3) aspect correlation across reviews prediction [19] (P_{review}, higher mean better), (4) nDCG of aspect ranking in reviews ($n\mathrm{DCG}_{aspect}$, higher mean better), using this metric to evaluate the model's ranking accuracy of aspects inside reviews with the ground truth aspect ratings are used as the graded relevance in the measure to measure the prediction quality of our proposed LRMNN model in comparison with other methods.

1. Root mean square error on aspect rating prediction is defined as:

$$\Delta_{aspect} = \sqrt{\frac{1}{|D_{test}|}\sum_{d=1}^{|D_{test}|}\sum_{i=1}^{k}(r_{di}^{*} - r_{di})^2 / k}$$

where r_{di}^{*} is the ground-truth rating for aspect A_i, the predicted aspect rating r_{di}.

2. Aspect correlation inside reviews is defined as:

$$P_{aspect} = \frac{1}{|D_{test}|}\sum_{d=1}^{|D_{test}|} P_{r_d^*, r_d}$$

where $P_{r_d^*, r_d}$ is the Pearson correlation between two vectors r_d^* and r_d. P_{aspect} aims to measure how well the predicted aspect ratings can preserve the relative order of aspects within a review given by their ground-truth ratings.

3. Aspect correlation across all reviews is defined as:
$P_{review} = \frac{1}{k}\sum_{i=1}^{k} P_{r_i^*, r_i}$ where r_i^* and r_i are the predicted and ground-truth rating vectors for aspect A_i across all the reviews. $P_{r_i^*, r_i}$ is the Pearson correlation between two vectors r_i^* and r_i.
4. Normalized discounted cumulative gain is defined as:
$$n\text{DCG}_{aspect} = \frac{1}{|D_{test}|}\sum_{d=1}^{|D_{test}|}\frac{DCG_d}{IDCG_d}$$
where $DCG_d = \sum_{i=1}^{k}\frac{2^{r_{di}}-1}{\log(i+1)}$ is that highly relevant documents appearing lower in a search result list should be penalized as the graded relevance value is reduced logarithmically proportional to the position of the result. The discounted CG accumulated at a particular rank position d, r_{di} is the predicted aspect rating for aspect A_i, $IDCG_d$ is computed the same as the DCG_d but it uses aspect ratings in the ground-truth.

We evaluate on two experimental cases, including aspect features based on bag of words and aspect features based on word vectors. In each the experimental case, the models used the same data set, we perform 5 times for training and testing, and report the mean value of metrics. In each time, we select randomly 75% of given reviews to train, the remaining 25% of given reviews to test. In Table 6, we show the mean value of four metrics for each method.

Table 6. Comparison with other models

Aspect feature	Method	Δ_{aspect}	P_{aspect}	P_{review}	$n\text{DCG}_{aspect}$
Bag of words $(\|V\|=3941)$	Global Prediction	0.825	0.316	0.569	0.705
	LRR	0.718	0.363	0.638	0.736
	Our LRMNN	0.723	0.451	0.632	0.756
	PRank	0.439	0.624	0.743	0.898
Word vectors $(\|V\|=3941)$	LRR	0.743	0.403	0.654	0.844
	Our LRMNN	0.712	0.468	0.644	0.928
	PRank	0.412	0.631	0.777	0.925
Word vectors $(\|V\|=29349)$	LRR	0.738	0.406	0.659	0.823
	Our LRMNN	**0.705**	**0.488**	**0.711**	**0.913**
	PRank	0.409	0.635	0.779	0.924

We can see that when aspect features present based on bag of words, although our model does not performs better than LRR model on Δ_{aspect} and P_{review} but it performs better than Global Prediction on P_{aspect}, P_{review} and $n\text{DCG}_{aspect}$ and it also performs better than LRR on P_{aspect} and $n\text{DCG}_{aspect}$. The model PRank performs best in all metrics. However, this model is fully supervised (i.e. both aspect ratings and overall rating) while others are not. When aspect features represent based on word vectors with dimensional number is 200 and the dictionary size of words is $|V|=3941$ or $|V|=29349$, we see that all models perform better when they use aspect features present based on bag of words. For each model, our model LRMNN performs better than LRR on Δ_{aspect}, P_{aspect} and $n\text{DCG}_{aspect}$, the model Global Prediction only depend on bag of words that does not depend on the dimensional word vector, so we do not evaluate it in this experimental case. For each the dictionary size of words, we use $|V|=29349$ for metrics better than $|V|=3941$.

5.4.2. *Evaluation on overall weighting aspects*

Since the given data have not the ground-truth aspect weights of aspects, we only can evaluate them indirectly through overall rating prediction. The combination of the predicted results of aspect rating and the inference results of aspect weight help us to predict overall rating.

We denote $\alpha_{(GlobalPrediction+LSAWs)}$ is the vector of the overall aspect weights which is computed by the method LSAWs; α_{PRR} is the vector of the overall aspect weights and is computed by LRR algorithm; α_{LRMNN} is the vector of the overall aspect weights which is computed by the method LRMNN.

We evaluate the quality of these overall aspect weights through three cases of overall rating prediction: (1) $PRank+\alpha_{(GlobalPrediction+LSAWs)}$, (2) $PRank+\alpha_{PRR}$, (3) $PRank+ \alpha_{LRMNN}$ We choose the two popular metrics, Mean Absolute Error (MAE) and Root mean square error to measure the differences of overall rating.

1. MAE is defined as:

$$MSE=\frac{1}{|D_{test}|}\sum_{d=1}^{|D_{test}|}\left|O_{d^*}-O_d\right|$$

2. RMSE is defined as:

$$RMSE=\sqrt{\frac{1}{|D_{test}|}\sum_{d=1}^{|D_{test}|}(O_{d^*}-O_d)^2}$$

Where O_{d^*} denotes the ground-truth overall rating for review *d*, O_d denotes the prediction overall rating for review *d*. The smaller MSE or $RMSE$ value means a better performance.

Table 7. The differences of overall ratings predicted with ground-true overall ratings

Algorithm	MSE	$RMSE$
$PRank+\alpha_{(GlobalPrediction+LSAWs)}$	0.292	0.384
$PRank+\alpha_{PRR}$	0.260	0.363
$PRank+ \alpha_{LRMNN}$	0.278	0.352

In Table 7 shows results of these evaluations. We see that the overall aspect weights are identified based on the combination of Global Prediction and LSAWs gives the worst result. For the metric MSE, the LRR gives the smallest differences of overall ratings predicted. For the metric $RMSE$, the our LRMNN gives the smallest mean. So with this metric it indicates the overall aspect weights (i.e. important aspects) are identified from our model LRMNN is the best.

6. Conclusion

In this paper, we have proposed a latent aspect mining framework for aspect based sentiment analysis from textual review data. The framework includes four main sub-tasks, i.e aspect term extraction, aspect category detection, aspect rating detection for each review, and indentify overall aspect weights for all reviews. Through experimental results, we see that aspect vectors represented by averaging word vectors are more effective than represented by a bag of word model. In most of the metrics used in experiment, our model LRMNN outperforms Global Prediction and LRR model.

References

[1] T. Wong, W. Lam, "Hot item mining and summarization from multiple auction web sites," in Proceedings of the 5th IEEE International Conference on Data Mining, IEEE Computer Society, 2005, pp. 797–800.

[2] W. Jin and H. H. Ho, “A novel lexicalized HMM-based learning framework for web opinion mining,” in 26th International Conference on Machine Learning, 2009, pp. 465–472.

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

[4] G. Ganu, N. Elhadad, A. Marian, "Beyond the stars: Improving rating predictions using review text content," in Proceedings of 12th International Workshop on the Web and Databases, WebDB, 2009, pp. 1–6.

[5] S. Kiritchenko, X. Zhu, C. Cherry, S. Mohammad, "Detecting aspects and sentiment in customer reviews," in: SemEval@COLING, The Association for Computer Linguistics, 2014, pp. 437–442.

[6] J. J. McAuley, J. Leskovec, D. Jurafsky, “Learning attitudes and attributes from multi-aspect reviews,” in: 12th IEEE International Conference on Data Mining, 2012, pp. 1020–1025

[7] X. Ding, B. Liu, P. S. Yu, “A holistic lexicon-based approach to opinion mining”, in: WSDM, ACM, 2008, pp. 231–240.

[8] W. Xu, Y. Tan, “Semi-supervised target-oriented sentiment classification”, Neurocomputing 337 (2019) 120 – 128

[9] B. Snyder, R. Barzilay, “Multiple aspect ranking using the good grief algorithm,” in: Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, April 22-27, 2007, Rochester, New York, USA, 2007, pp. 300–307.

[10] W. Wang, H. Wang, Y. Song, "Ranking product aspects through sentiment analysis of online reviews," Journal of Experimental & Theoretical Artificial Intelligence 29 (2) (2017) 227–246.

[11] Y. Liu, J.-W. Bi, Z.-P. Fan, "Ranking products through online reviews: A method based on sentiment analysis technique and intuitionistic fuzzy set theory," Information Fusion 36 (2017) 149 – 16.

[12] X. Li, L. Bing, P. Li, W. Lam, Z. Yang, "Aspect term extraction with history attention and selective transformation," in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, IJCAI'18, AAAI Press, 2018, pp. 4194–4200.

[13] J. Zhang, G. Xu, X. Wang, X. Sun, T. Huang, "Syntax-aware representation for aspect term extraction," Advances in Knowledge Discovery and Data Mining, Springer International Publishing, Cham, 2019, pp. 123–134.

[14] S. Poria, E. Cambria, A. F. Gelbukh, "Aspect extraction for opinion mining with a deep convolutional neural network," Knowl.-Based Syst. 108 (2016) 42–49.

[15] Chuhan Wu and Fangzhao Wu and Sixing Wu and Zhigang Yuan and Yongfeng Huang, "A hybrid unsupervised method for aspect term and opinion target extraction," Knowl.-Based Syst. 148 (2018) 66–73.

[16] Huaishao Luo, Tianrui Li, Bing Liu, Bin Wang and Herwig Unger, "Improving Aspect Term Extraction With Bidirectional Dependency Tree Representation," TASLP, 2019, 27(7):1201-1212.

[17] Xinjie Zhou, Xiaojun Wan* and Jianguo Xiao, "Representation Learning for Aspect Category Detection in Online Reviews", in Proceedings of AAAI, 2015, pp.417-423.

[18] S Movahedi, E Ghadery, H Faili, A Shakery, "Aspect Category Detection via Topic-Attention Network," CoRR abs/1901.01183 (2019)

[19] H.Wang, Y.Lu and Ch.Zhai, "Latent Aspect Rating Analysis on Review Text Data: A Rating Regression Approach," In Proceedings of SIGKDD, 2010, pp.168-176.

[20] Y.Xu, T.Lin and W.Lam, "Latent Aspect Mining via Exploring Sparsity and Intrinsic Information," In Proceedings of CIKM, 2014, pp.879-888.

[21] [21] Andrew L. Maas and Andrew Y. Ng, "A Probabilistic Model for Semantic Word Vectors," In Proceedings of NIPS, 2011,

[22] Z.Zha, J.Yu, J.Tang, M.Wang and T.Chua, "Product Aspect Ranking and Its Applications," IEEE Transactions on Knowledge and Data Engineering, vol. 26, no.5, pp.1211-1224, 2014.

[23] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean. "Distributed Representations of Words and Phrases and their Compositionality," In Proceedings of NIPS, 2013, pp. 1-9.

[24] Y Bengio, R Ducharme, P Vincent, and C Jauvin, "A neural probabilistic language model," Journal of Machine Learning Research, 3(6):1137–1155, 2003.

[25] W. Yih, K. Toutanova, J. Platt and C. Meek, "Learning Discriminative Projections for Text Similarity Measures," Proceedings of the Fifteenth Conference on Computational Natural Language Learning, 2011, pp. 247-256.

[26] J Pennington, R Socher and CD Manning, "Glove: Global Vectors for Word Representation," EMNLP, 2014, 1532-1543

[27] Sugato Basu, "Semi-supervised Clustering by Seeding," In ICML, 2002.

[28] Y. Lu, C. Zhai, and N. Sundaresan, "Rated aspect summarization of short comments," In Proceedings of WWW, 2009, pp. 131–140.

[29] K.Crammer and Y.Singer, "Pranking with ranking," In Proceedings of NIPS, 2001, pp.641-647.

[30] Duc-Hong Pham and Anh-Cuong Le and Thi Kim Chung Le, "A Least Square based Model for Rating Aspects and Identifying Important Aspects on Review Text Data," NICS, 2016, pp. 16-18.