Information Sciences 451–452 (2018) 295–309

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Leveraging sentiment analysis at the aspects level to predict ratings of reviews

Jiangtao Qiu[a,b,c], Chuanhui Liu[d], Yinghong Li[e], Zhangxi Lin[f,g,*]

[a] School of Information, Southwestern University of Finance and Economics, Chengdu, China
[b] Research Center on Big Data, Southwestern University of Finance and Economics, Chengdu, China
[c] Key Laboratory of Financial Intelligence and Financial Engineering of Sichuan, Chengdu, China
[d] School of Economics, Southwestern University of Finance and Economics, Chengdu, China
[e] School of Humanities, Southwestern University of Finance and Economics, Chengdu, China
[f] School of Economics, Xihua University, Chengdu, China
[g] The Rawls College of Business Administration, Texas Tech University, Lubbock, TX, USA

ARTICLE INFO

Article history:
Received 19 November 2016
Revised 28 November 2017
Accepted 2 April 2018
Available online 7 April 2018

Keywords:
Sentiment analysis
Class imbalance
Ratings of reviews
Business Intelligence

ABSTRACT

Online reviews are an important asset for users who are deciding to buy a product, see a movie, or go to a restaurant and for managers who are making business decisions. The reviews from e-commerce websites are usually attached to ratings, which facilitates learning from the reviews by users. However, many reviews that spread across forums or social media are written in plain text, which is not rated, and these reviews are called non-rated reviews in this paper. From the perspective of sentiment analysis at the aspects level, this study develops a predictive framework for calculating ratings for non-rated reviews. The idea behind the framework began with an observation: the sentiment of an aspect is determined by its context; the rating of the review depends on the sentiment of the aspects and the number of positive and negative aspects in the review. Viewing term pairs that co-occur with aspects as their context, we conceived of a variant of a Conditional Random Field model, called SentiCRF, for generating term pairs and calculating their sentiment scores from a training set. Then, we developed a cumulative logit model that uses aspects and their sentiments in a review to predict the ratings of the review. In addition, we met the challenge of class imbalance when calculating the sentiment scores of term pairs. We also conceived of a heuristic re-sampling algorithm to tackle class imbalance. Experiments were conducted on the Yelp dataset, and their results demonstrate that the predictive framework is feasible and effective at predicting the ratings of reviews.

Published by Elsevier Inc.

1. Introduction

Online reviews are an important asset for users who are deciding to buy a product, see a movie, or go to a restaurant and for managers who are making business decisions. When we talk about the reviews in the context of e-commerce, they usually refer to the text that is posted under the products, services, or businesses shown on the e-commerce website. Always, they are attached by star ratings that vary from 1-star to 5-star, which could facilitate learning about the reviews by the visitors. Fig. 1 exhibits an example of such a review on the Amazon website, which involves the Samsung Galaxy S6. A 3-star rating is assigned to the item.

* Corresponding author.
E-mail addresses: qiujt_t@swufe.edu.cn (J. Qiu), 114020202002@2014.swufe.edu.cn (C. Liu), abstract@swufe.edu.cn (Y. Li), zhangxi.lin@ttu.edu (Z. Lin).

https://doi.org/10.1016/j.ins.2018.04.009
0020-0255/Published by Elsevier Inc.

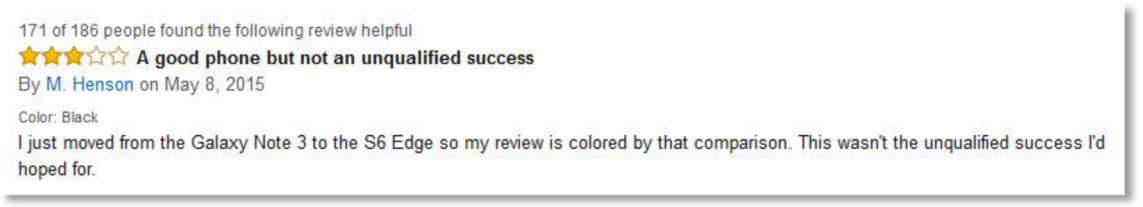

Fig. 1. A review on Amazon.

Fig. 2. A tweet on Twitter.

Cheesecake Factory is a solid choice, and you pretty much know what you're getting when you go. Mom and I visited on a Monday evening- no wait, seated right away. Service wasn't particularly friendly, and I did have a wait for a refill, but it was adequate. As always, there is a huge menu with lots of choices and large portions. My meal was a bit underseasoned, almost bland, but did the trick. As always, the cheesecakes are the stars, and the Red Velvet Cheesecake was delicious.

Fig. 3. A review with a 3-star rating. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

However, other types of reviews are also widely spread across forums or social media. They do not show significant differences compared with those on the e-commerce website, except for the lack of ratings. This paper calls them non-rated reviews. For example, Fig. 2 illustrates a tweet on Twitter that addresses the iPhone 7.

Many business intelligence (BI) applications consider text messages that are scattered across the Internet to be important data sources. If these applications can provide ratings for the non-rated reviews, it will help a substantial amount in making business decisions. For example, in a system, we seek to obtain reviews from the Internet that are associated with a certain business and then calculate the star ratings for them. Furthermore, we aggregate the rated reviews to determine a dynamic performance that enables managers to conveniently gain insight into the business. Predicting the ratings of the non-rated reviews could be a valuable tool for building business intelligence applications.

The ratings prediction is a subfield of sentiment classification [3,4,9,24,26]. This subfield has attracted much interest over the past decade since it emerged in 2005 [8]. For instance, TASS is an experimental evaluation workshop for sentiment analysis focused on the Spanish language since 2012. Participants are expected to submit experiments for the 6-label evaluation (strong positive, positive, neutral, negative, strong negative and a no sentiment tag) on a public corpus. The six labels are equivalent to the rating of stars in the e-commerce.

However, most previous research regards predicting ratings of reviews as a task of the document-level multilabel classification [6–8]. Motivated by Observation 1, we believe the performance of prediction of ratings of reviews can be significantly improved by leveraging sentiment analysis at the aspects-level. Examining reviews on the e-commerce website, we obtain Observation 1.

Observation 1: Every review involves at least one aspect; almost all of the aspects in a 5-star rating review are positive sentiment; the aspects of a 1-star review tend to be negative sentiments; and a 3-star review comprises a nearly equal number of positive and negative aspects.[1]

Fig. 3 shows a 3-star rating review that was derived from the Yelp website, where the aspects highlighted in a red color have positive sentiments and those highlighted in a yellow color have negative sentiments. Observing the review, we find that the number of positive aspects is almost equal to the number of negative aspects.

[1] In this study, we give items, products or service in e-commerce a general name, *business*. Each of the reviews on the e-commerce website is linked with a business. The topic mentioned in a review usually involves the features or attributes of a business. We also call entities, features, or attributes *aspects* in this paper. (See Section 3 for their formal definitions.)

This study intends to build a predictive framework based on sentiment analysis at the aspects-level to provide star ratings for non-rated reviews. Observation 1 builds the cornerstone for this study. We also seek to provide the evidence for the reliability of the observation via experiments (see Section 4.3 for details). Motivated by Observation 1, the task of predicting ratings can be formally decomposed into three steps: extracting the aspects, obtaining their sentiments, and then, predicting the rating based on the aspects. To calculate the sentiment scores of the aspects, we must derive the context of the aspects, i.e., terms that co-occur with the aspects in the same sentence. The sentiment of the aspects depends on their context. Traditional lexicon-based sentiment analysis employs a sentiment score of words provided by the lexicon to determine the sentiment scores of phrases or sentences, even documents. Prior work [4,5,25] has proven that the sentiments of words could change depending on their context. Hence, in this study, we use the term pair $\langle w_1, w_2 \rangle$ as a basic element, where both terms w_1 and w_2 are considered the context of their counterpart. We use a list of term pairs that co-occur with an aspect as the context of the aspect. Calculating the sentiment scores of the term pairs and then aggregating them, we obtain the sentiment score of the aspects. Furthermore, a cumulative logit model is developed that uses the sentiment scores of the aspects as features for predicting the ratings of the reviews. The main contributions of this study include the following:

(1) Develop a variant of the Conditional Random Field model, called SentiCRF, to build term pairs and calculate their sentiment scores.
(2) Conceive of a heuristic re-sampling algorithm to address the class imbalance that is encountered when we train the SentiCRF model.
(3) Build a framework to predict the ratings of non-rated reviews.[2]

The remainder of this paper is organized as follows. Section 2 reviews the related research on predicting the star ratings of the reviews. Section 3 introduces a predictive framework. Section 4 presents the experimental results. We also discuss interesting findings in Section 5. Section 6 offers the concluding remarks.

2. Related work

The task of predicting the star ratings of reviews originates from the sentiment classification of reviews, i.e., classifying reviews as recommended (thumbs up) or not recommended (thumbs down) [3]. Research work for review classification can be divided into machine learning methods, lexicon-based methods and hybrid methods. Some studies see review classification as a problem of text classification, which employs traditional machine learning methods, e.g., SVM (Support Vector Machine) [1] or Naive Bayesian [1], to train a predictive model based on reviews with polarity labels and then to calculate the sentiment polarity for the non-rated reviews using the model. There are also studies that employ an unsupervised learning method. For example, Turney [3] presents an effective unsupervised learning algorithm to perform the task. The polarity of a review is predicted through calculating the average semantic orientation of the phrases in the review. The first step is to use a part-of-speech tagger to identify phrases in the review that contain adjectives or adverbs. The second step is to estimate the semantic orientation of each extracted phrase. The semantic orientation of a phrase is calculated as the mutual information between the given phrase and the word “excellent” minus the mutual information between the given phrase and the word “poor”. A review is classified as recommended if the average semantic orientation of its phrases is positive.

Lexicon-based approaches to sentiment analysis are currently gaining in popularity due to their simplicity and domain independence [9]. These approaches rely on sentiment lexicons, where the words in a collection of words are marked with fixed sentiment polarities. However, the sentiment orientations of the words (positive, neutral, negative) and/or sentiment strengths could change depending on their context and targeted entities. Many approaches to automatic sentiment analysis begin with a large lexicon of words that are marked with their prior polarity. However, the contextual polarity of the phrase in which a specific instance of a word appears could be quite different from the prior polarity of the word. Positive words are used in phrases that express negative sentiments, or vice versa. Additionally, quite often words that are positive or negative out of context are neutral in context, which means that they are not even being used to express a sentiment. Wilson et al. [4] sought to distinguish between prior and contextual polarity, with a focus on understanding which features are important for the recognition of contextual polarity.

Many research efforts combine machine learning methods with a lexicon to improve the sentiment classification performance. Ortigosa et al. [11] performed sentiment classification and sentiment change detection on Facebook comments using a lexicon and the SVM. Prabowo1 and Thelwall [10] combined a rule-based method and multiple machine learning methods to improve the sentiment classification performance. Appel et al. [24] combined a sentiment lexicon, SentiWordNet [27], and fuzzy sets to estimate the semantic orientation polarity for sentences.

Compared to inferencing the sentiment polarity (e.g., “thumbs up” or “thumbs down”) of a review [3], review rating prediction [8] is a more challenging task. Most of the existing studies follow Pang and Lee [8] and cast this problem as a multiclass classification task. They typically employ machine learning algorithms in a supervised learning manner and build the rating predictor from reviews with accompanying ratings. In this direction, most studies focus on designing effective context-level features [6] and incorporating user-level features [2,7] to obtain a better prediction performance. Qu et al.

[2] We build a prototype to present our work (http://www.biswufe.cn/predratings).

[6] introduced a type of bag-of-opinions representation, where an opinion within a review consists of three components: a root word, a set of modifier words from the same sentence, and one or more negation words. Each opinion is assigned a numeric score that is learned, by ridge regression, from a large, domain-independent corpus of reviews. Li et al. [2] proposed a learning framework that incorporates reviewer and product information into a text-based predictive model for rating prediction. The reviewer, product and text features are modeled as a three-dimensional tensor. Tang et al. [7] think that each user has an impact on how to interpret the textual content of a review. They address this issue through developing a neural network that accounts for user information.

In this study, we focus on calculating the rating relative to plain text reviews that do not contain information on the reviewer. That approach makes a significant difference between our work and the work in [2,7]. Our work is close to the work in [6]. However, the proposed method in [6] requires combining the bag-of-opinions model with a unigram model that is trained on a domain-dependent corpus. The premise of using the model is to learn the dependent domain of a review.

3. A predictive framework

To better present this study, we first provide three definitions. The Yelp website[3] uses the term *business* to refer to the target of the reviews. We also employ this term to present our work.

Definition 1 (Business). This study gives the items, products or services presented in the e-commerce website, which are the targets of reviews, a general name, *business*.

Liu [17] uses the term entity to indicate the products and uses aspects to refer to the features or attributes of the products. He also uses the term general aspects to refer to the entities. This study gives a general definition relative to the aspects.

Definition 2 (Aspects). This study considers a collection of entities and their features or attributes in a review to be the aspects of the review.

Some researchers employ machine learning methods to extract aspects from reviews. For example, Poria et al. [28] propose a deep learning approach to aspect extraction in opinion mining. This study, however, uses a simple approach for aspects extraction that nouns and noun phrases (or groups) identified via a part-of-speech (POS) tagger are considered aspects in the review. Although this method is very simple, it is in fact quite effective. Some commercial companies are using this method with several improvements [17].

Definition 3 (Context of aspects). The context of an aspect, denoted by a, is a set of term pairs, T. Given a term pair $\langle w_i, w_j\rangle \in T$, it co-occurs with the aspect a in the same sentence s.t. $w_i \neq w_j$, $w_i \neq a$, $w_j \neq a$ and $\langle w_i, w_j\rangle = \langle w_j, w_i\rangle$. Each term pair $\langle w_i, w_j\rangle \in T$ is attached to both a positive and negative sentiment score $\{\lambda_p, \lambda_n\}$.

$\langle w_i, w_j\rangle = \langle w_i, w_j\rangle$ indicates that the terms within a term pair are order-independent. If we use only terms that co-occur with an aspect a as the context of a, then intuitively, we can employ a sentiment lexicon to determine the sentiment of the aspect a. However, many studies have shown that sentiments of words depend largely on their context [19,20,25]. Hence, this study explores the term pairs $\langle w_i, w_j\rangle$ where w_i and w_j co-occur within the same sentence and one term is regarded as the context of its counterpart.

Let us take an example. The business in the review in Fig. 3 is *Cheesecake Factory*. The aspects include *service, meal, menu* and *refill.* The context of the aspect "*service*" is "*particularly friendly*".

Following the motivation discussed in Section 1, this study develops a predictive framework to predict the ratings for non-rated reviews. The framework, illustrated in Fig. 4, works with the following steps.

(1) Use the reviews derived from the e-commerce websites as a training set. In the preprocessing phase, we employ a list of stop words to delete the stop words from the reviews, using Stanford POS-Tagger [22] as a tool to pick up terms, and then generating term pairs.
(2) Train a SentiCRF model, obtaining a collection of term pairs that are attached to sentiment scores.
(3) Develop a cumulative logit model (CLM).
(4) Extract aspects of a non-rated review, building their context.
(5) Map the term pairs in the context of the aspects to those in the collection generated in step (2). Calculate the sentiment for every aspect in the review based on its context. Then, build a feature vector for the review. (See Section 3.4 for details.)
(6) Finally, employ CLM to predict the star rating for the non-rated review.

Both the SentiCRF model and the cumulative logit model are key components of the predictive framework. In the remaining part of this section, we give a detailed discussion about these two components.

3.1. Pre-processing

Taboada et al. [21] give many good guides for lexicon-based sentiment analysis. Their guidance includes the following: (1) adjectives, nouns, verbs and adverbs can be used as the source of subjective content in a document; (2) a negator can

[3] http://www.yelp.com.

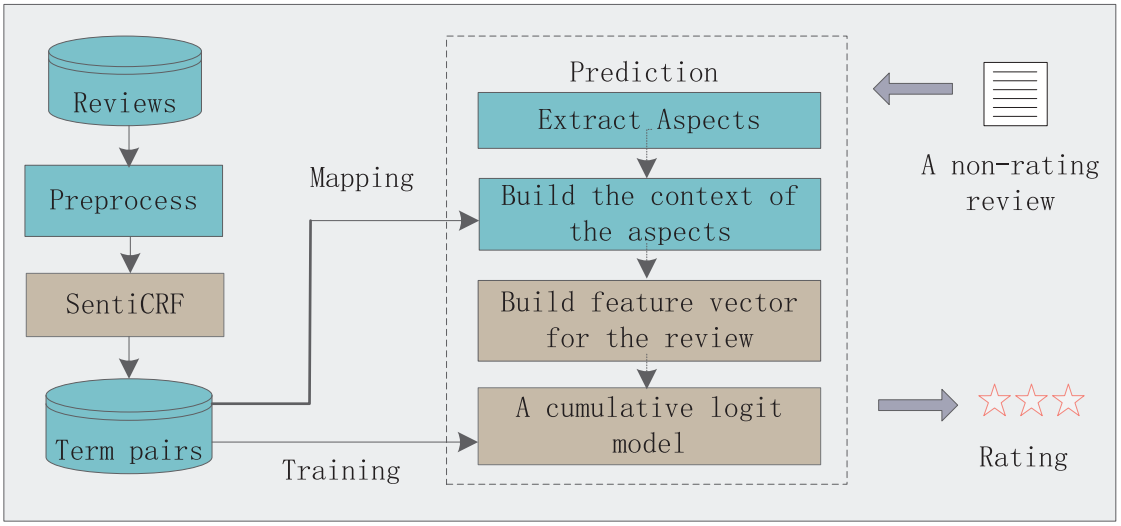

Fig. 4. The predictive framework.

Sentence: "*Service wasn't particularly friendly*"
The extracted terms are {*service*, *be*, *not*, *particularly*, *friendly*}
The extracted term pairs are {<*service*, *be*>, <*service*, *not*>, <*service*, *particularly*>, <*service*, *friendly*>, <*be*, *not*>, <*be*, *particularly*>, <*be*, *friendly*>, <*not*, *particularly*>, <*not*, *friendly*>, <*particularly*, *friendly*>}

Fig. 5. An example of extracting terms and term pairs from a sentence.

reverse the polarity of the items; and (3) irrealis indicates that the words that appear in a sentence might not be reliable for sentiment analysis. Inspired by their research, this study extracts adjectives, nouns, verbs, adverbs, negator and irrealis from reviews, and then, it generates term pairs from the sentence. Each of the terms that is extracted is stemmed.

Let us take an example. We extract the terms and term pairs from a sentence. They are shown in Fig. 5.

A very large number of term pairs are expected to be generated from the training set. It is necessary to perform feature selection. In practice, we generated above 30 million term pairs from the Yelp dataset. Employing a frequency-based feature selection method (picking up term pairs whose frequencies are more than 10), we obtain almost 3 million term pairs.

3.2. SentiCRF: a variant of a Conditional Random Field

One of the core components of the framework is a probabilistic discriminative model, called SentiCRF, which was developed to build a collection of term pairs and calculate their sentiment scores. SentiCRF is a variant of Conditional Random Field (CRF) [18]. Lafferty developed CRF to build probabilistic models to segment and label sequence data. Although the initial applications of CRFs used linear chains, there have been many later applications of CRFs that have more general graphical structures. Such structures are especially useful for relational learning. This study develops a variant of CRF to learn the relationships of both terms. The SentiCRF takes the form of

$$p(l|r) = \frac{\exp\left\{\sum_a \sum_{i,j} \sum_k \lambda_k f_k(w_i, w_j, l)\right\}}{\sum_{l'} \exp\left\{\sum_a \sum_{i,j} \sum_k \lambda_k f_k(w_i, w_j, l')\right\}} \tag{1}$$

denoted by $p(l|r)$ the probability that assigns to review r a label $l \in \{pos, neg\}$, where pos and neg refer to positive and negative sentiments, respectively. This model explores each of the aspects a in the review r. The feature function f_k is represented in the form of

$$f_k(w_i, w_j, l) = \begin{cases} 1, & \langle w_i, w_j \rangle \text{ has a label } l \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

λ_k denotes the coefficients of the feature function f_k. The notation $\langle w_i, w_j \rangle$ denotes a term pair that occurs in the context of aspect *a*.

The key point of this model lies in the estimation of the coefficients λ_k, which gauges the inclination of a term pair $\langle w_i, w_j \rangle$ toward positive or negative sentiment. We employ the gradient descent method to estimate the coefficients. The marginal log likelihood of this model is illustrated in the form of

$$L(\theta) = \sum_n \sum_a \sum_{i,j} \sum_k \lambda_k f_k(w_i, w_j, l) - \sum_n \log Z(r_n) - \sum_k \frac{\lambda_k^2}{2\sigma^2} \tag{3}$$

where there are n instances in the training set, and

$$Z = \sum_{l'} \exp\left\{ \sum_a \sum_{i,j} \sum_k \lambda_k f_k(w_i, w_j, l') \right\} \tag{4}$$

To avoid over-fitting, we also add the regularization $\sum_k \lambda_k^2 / 2\sigma^2$ to the log likelihood. When employing the gradient descent method for the problem, we require calculating the gradients using Eq. (5).

$$\frac{\partial L(\theta)}{\partial \lambda_k} = \sum_n \sum_a \sum_{i,j} f_k(w_i, w_j, l) - \sum_n \sum_{l'} p(l'|r_n) \sum_a \sum_{i,j} f_k(w_i, w_j, l') - \sum_k \frac{\lambda_k}{\sigma^2} \tag{5}$$

We further obtain the updating rule

$$\lambda_k \leftarrow \lambda_k + \alpha \frac{\partial L_n(\theta)}{\partial \lambda_k} \tag{6}$$

Each term pair $\langle w_i, w_j \rangle$ comprises two feature functions $f_{kp}(w_i, w_j, l_{pos})$ and $f_{kn}(w_i, w_j, l_{neg})$. The two coefficients λ_{kp} and λ_{kn} indicate the positive and negative sentiment scores of the term pair $\langle w_i, w_j \rangle$, respectively. At the same time, they also explicitly exhibit the co-occurrence strength of both terms w_i and w_j.

This study also learns the sentiments of a single term. We give a variant of Eq. (1), which is also a discriminative model.

$$p(l|r) = \frac{\exp\{\sum_a \sum_i \sum_m \mu_m s_m(w_i, l)\}}{\sum_{l'} \exp\{\sum_a \sum_i \sum_m \mu_m s_m(w_i, l')\}} \tag{7}$$

This model examines each aspect a in the review r. Here, w_i is a term that occurs in the context of the aspect a, with μ_m the coefficient of the status function $S_m(w_i, l)$, which is written in the form of

$$s_m(w_i, l) = \begin{cases} 1, & w_i \text{ has the label } l \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

The marginal likelihood function of the model is

$$L(\Theta) = \sum_n \sum_a \sum_i \sum_m \mu_m s_m(w_i, l) - \sum_n \log Z(r_n) - \sum_m \frac{{\mu_m}^2}{2\delta^2} \tag{9}$$

where

$$Z = \sum_{l'} \exp\left\{ \sum_a \sum_i \sum_m \mu_m s_m(w_i, l') \right\} \tag{10}$$

Suppose that the training set has n instances. We calculate the gradients

$$\frac{\partial L_n(\theta)}{\partial \mu_m} = \sum_n \sum_a \sum_i s_m(w_i, l) - \sum_n \sum_{l'} p(l'|r_n) \sum_a \sum_i s_m(w_i, l') - \sum_m \frac{\mu_m}{\delta^2} \tag{11}$$

Then, the updating formula can be written as

$$\mu_m \leftarrow \mu_m + \beta \frac{\partial L_n(\theta)}{\partial \mu_m} \tag{12}$$

Every term w_i comprises two status function $s_{mp}(w_i, l_{pos})$ and $s_{mn}(w_i, l_{neg})$. The two coefficients μ_{mp} and μ_{mn} indicate the positive and negative sentiment scores of the term.

To gain deeper insight, we further provide a discussion about the sentiment of the term pairs and terms, i.e., λ_{pos}, λ_{neg}, μ_{pos}, μ_{neg} learned from the Yelp dataset in Section 5.

3.3. Class imbalance

In this study, we seek to generate the term pairs and learn their sentiment scores from the training set. Furthermore, we leverage the sentiment scores of the term pairs to predict a rating relative to non-rated reviews. However, in the first place, we must evaluate how well these term pairs work. Therefore, we conduct the experiments on sentiment classification using the term pairs obtained from SentiCRF (see Section 4.2 for details). Here, we see the accuracy of using the term pairs to classify reviews as the performance of SentiCRF. The experimental results (Table 3) shows that the classification accuracy reaches 83%, and the accuracy on the positive reviews can even reach 99%. However, the accuracy on the negative reviews is only 24%.

Usually, the reviews dataset is a class imbalanced dataset. The Yelp dataset, which was employed for the experiment, is an obviously skewed dataset in which the positive reviews (4- or 5-star ratings) and negative reviews (1- or 2-star ratings) account for 66.7% and 19.1% of the total reviews, respectively. We can conclude that the class imbalance of the review dataset seriously harms the performance of SentiCRF.

To ensure that the term pairs that are generated from the SentiCRF are effective, we must address the class imbalance of the training set. Prior work [12] shows that the re-sampling is an effective solution relative to the issue. Re-sampling approaches include randomly over-sampling the minority class and under-sampling the prevailing class. We seek to tackle this issue with over-sampling. Taking the Yelp dataset as an example, we first simply broaden the negative reviews in the training set through repeating them three times, which makes the number of positive and negative instances in the training set approximately equal. Conducting experiments again, we find that the accuracy on the positive and negative reviews reaches 82.3% and 89.7%, respectively. This finding shows that even a simple re-sampling can effectively improve the performance of SentiCRF.

Instead of simply duplicating the original instances in the dataset, we intend to investigate the feasibility that we can better cope with the class imbalance when training the SentiCRF model. Inspired by the AdaBoost algorithm [16], we conceive of a heuristic re-sampling algorithm to further improve the performance of SentiCRF. The AdaBoost algorithm works with the following steps: initially, every instance in the training set is assigned a weight, and a new training set D_i is generated through sampling instances by their weights. Then, a classifier M_i is built using D_i. Its error is then calculated using D_i as a test set. The weights of the training instances are then adjusted according to how they were classified. Through repeating the above steps, a new training set D_{i+1} and, then, a classifier M_{i+1} are obtained. In the final step, all of the classifiers $M_1 \sim M_n$ are assigned a weight by their misclassification rate. For a new instance, every classifier gives a classification result that is then multiplied by its error rate. Having aggregated all of the classification results, the algorithm can obtain the final classification of the new instance.

We conceive of a heuristic re-sampling algorithm to build the training set. This algorithm extracts instances by their weights and re-calculates their weights relative to the error rate of the classification model in each iteration. The algorithm is formally described in Algorithm 1:

Algorithm 1 A heuristic re-sampling for improving the SentiCRF model.

Input: Training set D; the number of iterations, it
Output: a SentiCRF model
Steps:
1. Initialize all of the instances' weight to $1/|D|$
2. $n \leftarrow \text{argmax}\{|D_{pos}|, |D_{neg}|\}$
3. Initialize θ_0 of SentiCRF model
4. **For** $i = 1$ to it do
5. Sample n instances with replacement from D_{pos} and D_{neg} relative to their weights, respectively, to build $D_i = D_{pos} \cup D_{neg}$.
6. Use the parameters θ_{i-1} of model M_{i-1} as the initialized parameters θ_i of the model M_i
7. Learn parameters θ_i of SentiCRF model M_i using D_i
8. Use M_i to classify D_i, then compute the error rate, *erate*, of M_i
9. Update the weight of every misclassification instance with $w \leftarrow w \times (1 + erate(M_i))$
10. Normalize the weights of the instances in D_{pos} and D_{neg}, respectively, s.t. $\sum_{j \in D_{pos}} w_j = 1$ and $\sum_{j \in D_{neg}} w_j = 1$.
11. **End For**

The error rate is calculated using formula (13):

$$\text{erate}(M) = \frac{\sum_{j=1}^{|D|} \text{err}(X_j)}{|D|} \tag{13}$$

where

$$\text{err}(X_j) = \begin{cases} 1, & X_j \text{ is misclassified} \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

The error rate varies from 0 to 1.

In the algorithm, every model M has a set of parameters θ (the coefficients of the feature function of the term pairs or those of the status functions of the terms). Because we sample the instances in the dataset to build a new training set, it is

Table 1
Features derived from a review.

Features	Equation	Description
n_{ap}	$n_{ap} = \|\{a\|SA(a) \geq 0\}\|$	The number of aspects in the review, r, s.t. SA $\geqslant 0$.
n_{an}	$n_{an} = \|\{a\|SA(a) < 0\}\|$	The number of aspects in the review, r, s.t. SA < 0.
S_p	$s_p = \sum_{tp \in r} \lambda_p(tp)$	The sum of positive sentiment scores of all term pairs *tp* generated in the review
s_n	$s_n = \sum_{tp \in r} \lambda_n(tp)$	The sum of negative sentiment scores of all term pairs *tp* generated in the review
t_p	$t_{p=} \sum_{t \in r} \mu_p(t)$	The sum of positive sentiment scores of all terms t extracted from the review
t_n	$t_{n=} \sum_{t \in r} \mu_n(t)$	The sum of negative sentiment scores of all terms t extracted from the review
len		*The number of words in the review*

very possible that some term pairs (or terms) are excluded from the new training set. However, whether we exclude these term pairs (or terms) from model M_i or use random values as the coefficients of these term pairs (or terms) can heavily impair the performance of the model. In the algorithm, when training a model M_i, we use the parameters of the model M_{i-1} as the initial values of those of M_i (except that in the first iteration, the parameters are initialized with random values), which could cope with this problem well.

3.4. Predicting the ratings of the reviews

Constructing a feature vector X relative to a review r is an important step in the prediction. Observing a consumer review in e-commerce, we can find that the reviews always contain topics, and the reviewers tend to show their attitudes or sentiments toward these topics. In this study, aspects are referred to the topics and the context of aspects represents their sentiments. Motivated by Observation 1, we build the feature vector X via the following steps:

Let a be an aspect, let C be the context of a, and let t be a term pair within C. Each of the term pairs t contains both a positive score $\lambda_p(t)$ and a negative score $\lambda_n(t)$. Before using λ_p, λ_n, μ_p and μ_n, they are normalized using Eq. (15).

$$\begin{aligned}
\lambda_p &= \log(1+\lambda_p)/(\log(1+\lambda_p)+\log(1+\lambda_n)) \\
\lambda_n &= \log(1+\lambda_n)/(\log(1+\lambda_p)+\log(1+\lambda_n)) \\
\mu_p &= \log(1+\mu_p)/(\log(1+\mu_p)+\log(1+\mu_n)) \\
\mu_n &= \log(1+\mu_n)/(\log(1+\mu_p)+\log(1+\mu_n))
\end{aligned} \tag{15}$$

We first calculate the sentiment of an aspect a (SA for short) in a review r.

$$SA(a) = \begin{cases} 1, & f(a) \geq 0 \\ -1, & f(a) < 0 \end{cases} \tag{16}$$

where

$$f(a) = \sum_{t \in C} \log(1+\lambda_p(t)) - \sum_{t \in C} \log(1+\lambda_n(t)) \tag{17}$$

We then can build a feature vector X for review r, where

$$\begin{aligned}
X = \Bigg(& x_1 = \frac{n_{ap}+1}{n_{ap}+n_{an}+2}, x_2 = \frac{n_{an}+1}{n_{ap}+n_{an}+2}, x_3 = \frac{s_p+1}{s_p+s_n+2}, \\
& x_4 = \frac{s_n+1}{s_p+s_n+2}, x_5 = \frac{t_p+1}{t_p+t_n+2}, x_6 = \frac{t_n+1}{t_p+t_n+2}, x_7 = \log(1+len) \Bigg)
\end{aligned} \tag{18}$$

The features shown in the vector are represented in Table 1.

We introduce the features t_p and t_n to the model to address short reviews, where the texts are too short to generate effective term pairs.

In the ordinal regression problem, the response variable can take a small number of discrete, ordered values, which are often referred to as categories. Ordinal regression differs from multiclass classification due to the existence of an order relation on the response variable. Because the ratings have an order from 1-star to 5-star, we see the rating prediction as an ordinal regression problem. We build a cumulative logit model (CLM) [13], one of the models for solving the ordinal regression problem, to perform the predictive task in which an ordinal response variable Y_j can fall within $j = 1,\ldots, 5$ ratings. Then, Y_j follows a multinomial distribution with parameter π, where π_{ij} denotes the probability that the ith observation falls in the response ratings j. We define the cumulative probabilities as

$$Y_{ij} = P(Y_i \leq j) = \pi_{i1} + \ldots + \pi_{ij} \tag{19}$$

The logit function is defined as $\text{logit}(\pi) = \log[\pi/(1-\pi)]$, and the cumulative logits are defined as

$$\text{logit}(\gamma_{ij}) = \text{logit}(P(Y_i \leq j)) = \log\left[\frac{P(Y_i \leq j)}{1-P(Y_i \leq j)}\right], \quad j = 1, \ldots, J-1 \tag{20}$$

in such a way that the cumulative logits are defined for all but the last 5-star rating. A cumulative logit model is a regression model for cumulative logits:

$$\text{logit}(\gamma_{ij}) = \alpha_j - X^T\beta \tag{21}$$

where X denotes a k-dimensional vector that is derived from a review, and β denotes a coefficient vector that corresponds to X. The α parameters provide each cumulative logit with its own intercept. However, the coefficient β does not depend on j. That fact means that the effect of the explanatory variable is the same for different logit functions.

The prediction steps are shown in Algorithm 2.

Algorithm 2 Predicting ratings for non-rated reviews with a cumulative logit model.

Input: a feature vector of a review
Output: a rating
Steps:
1. Given a review r and its feature vector X, we calculate the cumulative probability by employing the trained cumulative logit model. $P(Y_i \le j) = 1/(1 + e^{-(\alpha_j - X^T\beta)})$
2. For each of the ratings $j = 1,\ldots, J$, we compute $P(rating = j) = P(Y_i \le j) - P(Y_i \le j - 1)$
3. Finally, we obtain the rating of the review r $rating(r) = \max \arg_j (P(rating = j) | j = 1, \ldots, J)$

4. Experiments

In this section, we evaluate both the feasibility and the performance of the predictive framework developed in this study using the Yelp dataset.[4] The experiments are conducted on a PC with an INTEL i7 processor and 32 GB RAM.

4.1. The Yelp review dataset

The Yelp dataset comprises 1,569,263 reviews and 61,184 businesses. The number of reviews relative to the five ratings are 159,811, 140,608, 222,719, 466,598 and 579,527, which correspond to 1-star to 5-star, respectively. The Yelp dataset is found to be a skewed dataset.

4.2. Experiments for class imbalance

In this subsection, we investigate the impact of a class imbalance on the performance of SentiCRF, and we further evaluate the effectiveness of the heuristic re-sampling algorithm to address the imbalance class through conducting experiments on the sentiment polarity classification using the term pairs obtained via SentiCRF. However, reviews with 3-star will not contribute to the sentiment polarity classification. Therefore, we build a dataset, called D, which excludes 3-star rating reviews from the Yelp dataset. We then reassign the reviews in D labels, where the reviews with a 1- or 2-star rating are given a negative label, and the reviews with a 4- or 5-star rating are given a positive label. The predictive results generated from the predictive model are also processed accordingly.

As noted in many studies, the classification performance over imbalanced data sets should not be expressed in terms of the plain accuracy [14]. The use of these simple measures might produce misleading conclusions because they do not account for the misclassification costs. They are strongly biased toward favoring the majority class. This study employs the balance accuracy (BAC) [14] to evaluate the classification performance, which is defined as an average accuracy within the decision classes.

$$\begin{aligned}
\text{ACC}(D_p) &= \frac{\text{TP}}{\text{TP} + \text{FN}} \\
\text{ACC}(D_n) &= \frac{\text{TN}}{\text{TN} + \text{FP}} \\
\text{ACC}(D) &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{TN} + \text{FP}} \\
\text{BAC}(D) &= \frac{\text{ACC}(D_p) + \text{ACC}(D_n)}{2}
\end{aligned} \tag{22}$$

The descriptions of the four parameters TP, TN, FP and FN are listed in Table 2.

We conceive of a simple classification approach for our purpose of evaluating SentiCRF. Extract all of the term pairs from a review; aggregate λ_p and λ_n of these term pairs; finally, assign a label to the review that corresponds to $\max(\Sigma\lambda_p, \Sigma\lambda_n)$.

Employing 10-fold cross-validation, the experimental results are shown in Table 3.

[4] https://www.yelp.com/dataset_challenge/dataset.

Table 2
A confusion matrix for binary classification.

	Predicted positive	Predicted negative
Positive class (D_p)	True positive (TP)	False negative (FN)
Negative class (D_n)	False positive(FP)	True negative (TN)

Table 3
Classification performance.

ACC	BAC	ACC (D_p)	ACC (D_n)
82.9%	61.45%	98.8%	24.1%

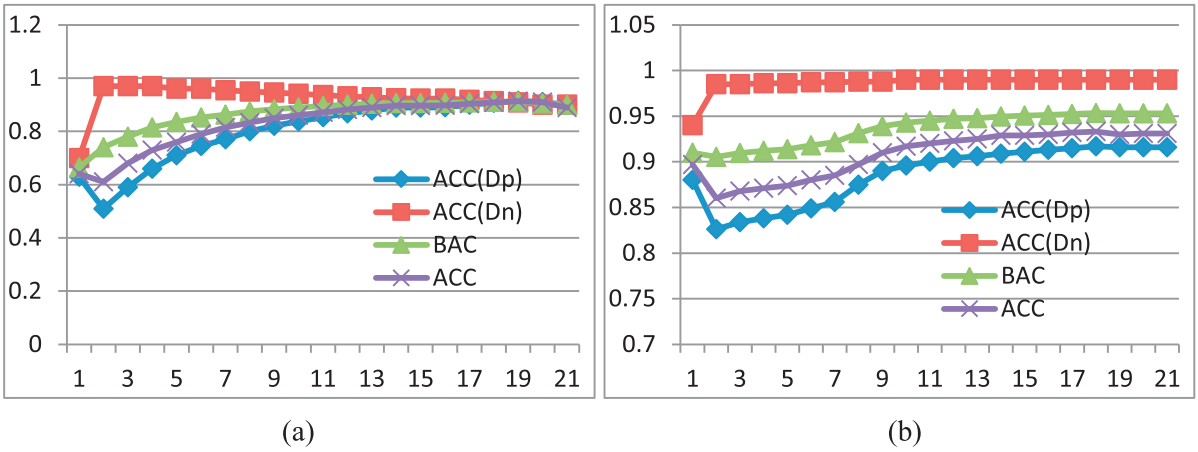

Fig. 6. The performance of Algorithm 1. (a) Using the sentiment score of terms while conducting the experiment. (b) Using the sentiment score of the term pairs while conducting the experiment.

In Table 3, although the experiments obtain an 82.9% accuracy, the performance even reaches 98.8% on positive samples, while ACC(D_n) exhibits a 24.1% (low) accuracy. Obviously, the large number of positive samples dominates the training process of the predictive model. We conclude that the imbalance of the classes has a large impact on the performance of SentiCRF.

We further explore whether Algorithm 1 might be able to tackle the impact of the class imbalance. The algorithm runs 20 iterations. In each iteration, we use the re-sampled dataset to train SentiCRF. We then make a classification decision on the training set to update the weights of the instances in the training set, and we conduct experiments on the testing set to evaluate the performance of SentiCRF. Fig. 6 illustrates the classification results on the testing set.

Fig. 6 illustrates the experimental results of employing terms and term pairs while conducting the experiments. Denote the accuracy measure by the y-axis and the number of iterations, k, by the x-axis in Fig. 6. When $k=1$, the training set is built by repeating instances with negative labels in the training set three times. Thus, the positive and negative instances in the training set have almost equal sizes. From $k=2$, we use Algorithm 1 to resample instances in the training set to build a new training set.

Fig. 6(a) and (b) shows that both BAC and ACC increase along with the increase in iterations until convergence. Algorithm 1 attempts to pick up instances that have high weights to build the training set for the next iteration. The convergence means that the algorithm has not already been able to find a clear classification boundary to perfectly divide these high error rate instances into either positive or negative classes. In other words, the heuristic re-sampling method has reached its best performance We conclude that the heuristic re-sampling algorithm is capable of effectively addressing the impact of imbalanced classes on training SentiCRF, because both BAC and ACC in Fig. 6(a) and (b) show a growing trend from the whole picture.

Table 4 lists a part of the data in Fig. 6. From this table, we can obtain a couple of interesting findings. (1) When $k=20$, both ACC and BAC on the two figures reach peaks. More specifically, the re-sampling algorithm shows significant improvement for the model that uses terms. The algorithm improves 27.2% ACC and 24.8% BAC compared with the method that simply repeats the negative samples. (2) The model that uses term pairs shows the best performance, with 95.4% BAC and 93.6% ACC, which are 4% and 2.4% higher than the model that uses terms. We can draw the following conclusions:

(1) The re-sampling algorithm proposed in this study can efficiently improve the performance of the SentiCRF model.
(2) The term pairs can accomplish sentiment analysis better than the terms.

Table 4
Classification performance.

	ACC (D_p)	ACC (D_n)	ACC	BAC
$K = 1$, with terms	62.9%	70.3%	64.0%	66.6%
$K = 1$, with term pairs	88.4%	94.4%	91.9%	91.4%
$K = 20$, with terms	90.9%	91.5%	91.2%	91.4%
$K = 20$, with term pairs	**91.7%**	**99.0%**	**93.6%**	**95.4%**

Table 5
Models for predicting ratings.

Models	Description
M0	The predictive framework in this study
M4	Use SentiWordNet [27] to build feature vectors for reviews and employ CLM to make the prediction
M5	An SVM-based multiclass classifier
M6	Combine SO and PSP to build the predictive model.
M7	Model UWRL$^+$ [7]
M8	A convolution neural network based model [23]

4.3. Experiments for predicting the ratings

This subsection investigates the performance of SentiCRF on predicting the star ratings of reviews in comparison to baseline methods. We employ the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to evaluate the experiments, which are written in the form of

$$\mathrm{MAE} = \frac{\sum_i |p_i - r_i|}{n} \tag{23}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_i (p_i - r_i)^2} \tag{24}$$

where r_i denotes the real rating of the ith review, p_i denotes the predicted rating, and n denotes the total number of reviews in the test set. Both metrics evaluate how much the predicted rating deviates from the real rating. A smaller value indicates a more accurate prediction.

We compare the predictive framework, called M0 in the experiment, with the baseline methods, which are listed in Table 5.

The first baseline method, called M4, is a simple model that uses terms that occur with aspects in the same sentence as the context of the aspects, and it employs sentiment scores that are provided by SentiWordNet to calculate the sentiments of the aspects. M4 uses x_1, x_2, x_5, x_6 and x_7 in Eq. (18) as its variables to build the feature vectors relative to the reviews. Then, it employs CLM to make the prediction.

Pang and Lee [8] see the predictive ratings of the reviews as a problem of text classification, and they propose an SVM-based method for predicting the ratings of reviews. We also build a SVM-based multiclass classifier as the baseline, called M5. The classifier is built using the libsvm software system [15]. In practice, we conceive of a strategy to make the SVM-based multiclass classifier where we generate a one-against-rest classifier relative to each of the ratings. Taking the classifier of a 3-star rating as an example, we build the one-against-rest review classifiers with the following steps:

(1) Build a training set, *D2*, and a test set, *S*. Build a dataset, $D2_p$, through selecting those reviews that have a 3-star rating in *D2*. Build a dataset $D2_n = \{D2\text{-}D2_p\}$.
(2) We obtain interest in terms that occur in $D2_p$ alone. Build a term list, *tlist*, using SentiWordNet and the stop list to select terms in $D2_p$, excluding those terms whose occurring frequency in reviews of $D2_p$ are less than 10.
(3) Use $D2_p$ as the positive samples, $D2_n$ as the negative samples, and *tlist* as the term list to build the vector space model (VSM). Compute the term frequency as the element's weights in VSM.

In the prediction phase, the multiclass classifier generates the probability relative to each rating. We pick up a rating that corresponds to the highest prediction value as the predicted result.

Pang and Lee [8] also proposed a PSP method that denotes the percentage of positive sentences in all of the subjective sentences. We employ SO + PSP as the third baseline M6, which predicts the rating of a review through calculating the percentages of positive phrases in the review. Tang et al. [7] developed a deep learning based model, called UWRL$^+$. They also conducted experiments for review rating prediction on the Yelp dataset. The experimental results are referred to as M7. Kalchbrenner built a convolution neural network (CNN) for sentence-level sentiment analysis tasks [23]. We employ the CNN for the task of predicting ratings, which is referred to as the fifth baseline M8. The experimental results are listed in Table 6.

Table 6
Experimental results for predicting the ratings of reviews.

	M0	M4	M5	M6	M7	M8
MAE	**0.581**	0.781	0.677	0.712	0.618	0.655
RMSE	**0.860**	1.056	0.931	1.091	0.962	0.981

Table 7
The models that use different variables.

Model	Variables	MAE	RMSE
M0	x_1, x_2, x_3, x_4, x_5, x_6, x_7	0.581	0.860
C1	x_3, x_4, x_5, x_6, x_7	**0.609**	**0.896**
C2	x_1, x_2, x_5, x_6, x_7	0.597	0.885
C3	x_1, x_2, x_3, x_4, x_7	0.580	0.859
C4	x_1, x_2, x_3, x_4, x_5, x_6	0.579	0.859
C5	x_1, x_2, x_3, x_4	0.579	0.858

Table 6 shows that our predictive framework outperforms M4 on both MAE and RMSE. This result means that the term pairs can better represent the context of the aspects than the terms. Using the sentiment lexicon for sentiment analysis is a convenient method but is always an oversimplified method. In addition, it can also be observed that M0 significantly outperforms M5, M6 and M7. This result indicates that the predictive framework is effective at the task of predicting ratings of reviews.

We also seek to find how the variables in CLM affect the performance of the model. The CLM model in Section 3.4 uses 7 variables (x_1,..., x_7). To investigate the importance of these variables, we construct five CLM models, listed in Table 7, which use only some of the variables.

Conducting experiments again, we list the experimental results in Table 7. Let M0 be the baseline. We suggest that if a model in Table 7 has the largest gap in performance compared to the baseline, those variables that are excluded from the model have the largest impact on the performance of CLM. Obviously, model C1 has the poorest performance, and it excludes variables x_1 and x_2. Because x_1 and x_2 are the normalized n_{ap} and n_{an}, which are the number of positive and negative aspects in a review, we can conclude that n_{ap} and n_{an} play the most important roles in the proposed model. This conclusion also gives strong evidential support for Observation 1 in Section 1.

We can also see that models C3, C4 and C5 do not have a significant difference compared to the baseline. This finding means that using t_{p}, t_{n} and the length of the review as the variables in the predictive model does not improve the performance of the model. Because there exists a high correlation within the three pair variables, $\langle x_1, x_2\rangle$, $\langle x_3, x_4\rangle$, $\langle x_5, x_6\rangle$ (see Section 5.2 for details), our model finally includes variables x_1, x_3, x_5, and x_7 alone.

5. Discussion

5.1. SentiCRF model

In Section 4, we employ the Yelp dataset to train the SentiCRF model. We list the top 20 positive sentiment and negative sentiment term pairs in Table 8. We rank the term pairs by their sentiment score, *sc*, which is calculated using the formula $sc = \log(\lambda_{\text{pos}} + 1) - \log(\lambda_{\text{neg}} + 1)$. If *sc* is larger than 0, then the term pair exhibits a positive sentiment; otherwise, it indicates a negative sentiment.

From observing Table 8, we find that '*downside*' surprisingly shows at the top of the list of positive sentiment term pairs. Here, '*downside*' means "the negative part of a situation" (from the Cambridge Dictionary). Intuitively, we consider 'downside' to be a word that has a negative sentiment. However, from retrieving the Yelp dataset, we find 7942 reviews that contain the word '*downside*', where there are 6364 reviews with 4 or 5 stars and 284 reviews with 1 or 2 stars. This finding means that the customers who use *downside* usually express a positive sentiment. For example, there is a review that includes

...The only downsides were the tight squeeze of the location,...

From this sentence, we can find that the user actually uses '*only downside*' to stress the positive sentiment expression in the review.

In addition, we believe that '*best*' is usually a strong positive sentiment word, but the term pair '⟨*best, mediocre*⟩' appears in the negative sentiment list. We pick up a review that contains both words from the Yelp dataset.

...The food was fresh but mediocre at best!...

We can find that '*mediocre at best*' is a fixed collocation that represents a negative sentiment. Because we exclude a preposition from a review during the pre-processing phrase, the term pair ⟨*at, best*⟩ is missing from the final term pairs list. Instead, ⟨*best, mediocre*⟩ becomes a frequent negative sentiment indicator.

Table 8
Top 20 positive and negative sentiment term pairs.

Top 20 positive sentiment term pairs			Top 20 negative sentiment term pairs		
Term pairs	λ_{pos}	λ_{neg}	Term pairs	λ_{pos}	λ_{neg}
⟨Only, downside⟩	24.40	1.37	⟨ever, worst⟩	5.29	213.21
⟨recommend, highly⟩	219.29	23.30	⟨experience, worst⟩	2.05	80.11
⟨Complaint, only⟩	74.37	7.37	⟨worst, service⟩	2.68	87.42
⟨delicious, everything⟩	35.08	3.03	⟨zero, star⟩	0.56	32.79
⟨surprise, pleasantly⟩	43.44	4.36	⟨have, worst⟩	16.45	353.02
⟨best, hand⟩	37.13	3.75	⟨customer, worst⟩	0.76	33.27
⟨ever, best⟩	249.76	32.07	⟨zero, give⟩	0.43	24.55
⟨worth, well⟩	71.26	9.21	⟨food, worst⟩	1.64	43.84
⟨back, definitely⟩	218.88	30.25	⟨be, tasteless⟩	5.32	105.84
⟨pleasantly, be⟩	65.62	9.08	⟨service, horrible⟩	8.08	141.68
⟨be, downside⟩	53.82	7.53	⟨horrible, customer⟩	1.66	37.13
⟨happier, not⟩	16.37	1.72	⟨flavorless, be⟩	4.77	75.95
⟨glad, find⟩	15.20	1.55	⟨best, mediocre⟩	2.15	40.97
⟨recommend, definitely⟩	63.24	9.11	⟨service, terrible⟩	8.06	119.13
⟨reasonable, very⟩	53.61	7.60	⟨unprofessional, be⟩	4.31	67.74
⟨gem, hide⟩	21.34	2.52	⟨inedible, be⟩	3.30	53.65
⟨again, definitely⟩	106.91	16.02	⟨rude, manager⟩	1.12	25.56
⟨delicious, also⟩	25.55	3.27	⟨worst, place⟩	2.66	44.78
⟨delicious, fresh⟩	40.56	5.80	⟨Horrible, food⟩	4.22	62.03
⟨friendly, professional⟩	23.09	2.95	⟨worst, be⟩	30.33	374.50

Table 9
Top 20 positive and negative sentiment terms.

Top 20 positive sentiment terms			Top 20 negative sentiment terms		
Terms	μ_{pos}	μ_{neg}	Terms	μ_{pos}	μ_{neg}
Perfect	32.39	6.61	Worst	2.62	34.97
Delicious	63.30	14.51	Horrible	3.87	31.53
Amaze	65.86	15.61	Rude	3.98	31.29
Fantastic	22.38	5.09	Terrible	3.96	27.95
Awesome	46.41	11.80	Disgust	1.22	11.55
Excellent	38.23	9.81	Bland	3.77	24.38
Perfection	5.46	0.86	Mediocre	2.38	16.83
Perfectly	15.18	3.68	Awful	2.36	16.37
Pleasantly	4.56	0.63	Poor	4.13	21.83
Gem	6.13	1.13	Tasteless	1.18	8.26
Wonderful	22.96	6.33	Worse	2.41	13.43
Highly	20.54	5.96	Flavorless	0.75	6.19
Favorite	43.01	13.35	Waste	4.42	18.87
Incredible	8.91	2.26	Inedible	0.39	4.05
Knowledgeable	7.43	1.77	Overpriced	3.72	15.94
Heaven	5.42	1.15	Stale	0.91	5.76
Yummy	13.32	3.99	Refund	1.80	8.81
Phenomenal	4.55	1.09	Refuse	1.85	8.94
Reasonable	15.78	5.36	Disappointment	1.98	9.33
Affordable	6.13	1.72	Manager	13.17	48.02

We also list the top 20 positive sentiment and negative sentiment terms in Table 9. We rank these terms according to their sentiment score, which is calculated using the formula $sc = \log(\mu_{pos} + 1) - \log(\mu_{neg} + 1)$.

Interestingly, we find '*manager*' to be in the negative list. Compared with Table 8, we can find that ⟨*rude, manager*⟩ occurs in the negative term pairs list. We might infer that when the customers write a review in Yelp, they frequently consider the managers to be a negative image.

5.2. CLM

In Section 4.3, we train a CLM and conduct the experiments relative to the rating-predictions on the Yelp data set. The parameters of the trained CLM are listed in Table 10, and its threshold coefficients are shown in Table 11.

Because $x_2 = 1 - x_1$, $x_4 = 1 - x_3$ and $x_6 = 1 - x_5$, there is a high correlation within the three pairs of variables $\langle x_1, x_2\rangle$, $\langle x_3, x_4\rangle$ and $\langle x_5, x_6\rangle$. Table 10 shows that the coefficients β_2, β_4 and β_6 cannot be correctly estimated and that the *P*-value of β_1, β_3, β_5 and β_7 in the table is smaller than 2e−16. We thus can definitely claim that the variables x_1, x_3, x_5 and x_7 are a meaningful addition to the model.

Table 10
The coefficients of the trained CLM.

Coefficients	Estimate	Standard error	Z value	Pr(>|z|)
β_1	4.440723	0.014603	304.09	<2e−16
β_2	NA	NA	NA	NA
β_3	23.827631	0.127489	186.90	<2e−16
β_4	NA	NA	NA	NA
β_5	9.309220	0.201445	46.21	<2e−16
β_6	NA	NA	NA	NA
β_7	0.158392	0.002053	77.14	<2e−16

Table 11
Threshold coefficients.

Threshold coefficients	Estimate	Standard error	Z value
α_1	16.02051	0.06599	242.8
α_2	17.38488	0.06635	262.0
α_3	18.93708	0.06664	284.2
α_4	20.95836	0.06697	313.0

Observing Table 11, we find that all of the threshold coefficients have a very low standard error. This finding indicates that the estimated threshold coefficients have a very low uncertainty.

6. Conclusions

Predicting the ratings of the reviews is a task that is both interesting and challenging. This task could be a basic and even key component in a business intelligence application. In this paper, we propose a predictive framework to meet the challenges. We develop a SentiCRF model to build a collection of term pairs and to calculate their sentiments from a training set. To predict the star rating of a non-rated review, we extract the aspects of the review and their contexts, i.e., term pairs that co-occur with the aspect in the same sentence, mapping these term pairs to the term pairs in the collection generated by the SentiCRF. Having built the feature vectors by summarizing the term pairs in the review, we employ a cumulative logit model to predict the ratings of the reviews.

We conduct experiments on the Yelp dataset, drawing the following conclusions from the experimental results:

(1) Class imbalance in the dataset has a significant impact on the learning sentiments of the term pairs. The heuristic re-sampling algorithm in this study can efficiently cope with the challenge of the class imbalance.
(2) Term pairs can better represent the context of the aspects than the terms alone.
(3) Our framework significantly outperforms the baseline methods on predicting ratings of reviews.
(4) Experimental results provide evidence for Observation 1, which is the cornerstone of this study.

We developed a supervised method to learn the sentiments of term pairs. These sentiments thus have a high correlation with the certain domain. For example, the sentiments learning from the Yelp dataset cannot be applied to make sentiment analysis for movie reviews. However, we also do not believe that there exists a general model that can do a good job on all domains. Compared to the lexicon-based method on sentiment analysis, we think that the supervised method is a better solution if sufficient training data are available.

Acknowledgments

This work was supported by National Natural Science Foundation of China (no. 71571145), Humanities and Social Science Foundation of Ministry of Education of China (no. 14YJAZH063) and the Fundamental Research Funds for the Central Universities (nos. JBK120505 and JBK150503).

References

[1] B. Pang, L. Lee, S. Vaithyanathan, Thumbs up? Sentiment classification using machine learning techniques, in: Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2002.
[2] Li, F., et al., Incorporating reviewer and product information for review rating prediction, Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI), 2011.
[3] P.D. Turney, Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews, in: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02), 2002, pp. 417–424.
[4] T. Wilson, J. Wiebe, P. Hoffmann, Recognizing contextual polarity: an exploration of features for phrase-level sentiment analysis, Comput. Ling. 35 (3) (2009) 399–433.
[5] A. Weichselbraun, S. Gindl, A. Scharl, Extracting and grounding context-aware sentiment lexicons, IEEE Intell. Syst. 28 (2) (2013) 39–46.
[6] Qu, L., Ifrim, G., Weikum, G., The bag-of-opinions method for review rating prediction from sparse text patterns, Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10), Pages 913–921.

[7] D. Tang, B. Qin, T. Liu, Y. Yang, User modeling with neural network for review rating prediction, in: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI), 2015.
[8] B. Pang, L. Lee, Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales, in: Proceedings of the ACL, 2005.
[9] M. Thelwall, K.B. Paltoglou, Sentiment strength detection for the social web, J. Am. Soc. Inf. Sci. Technol. 63 (1) (2012) 163–173.
[10] R. Prabowo1, M. Thelwall, Sentiment analysis: a combined approach, J. Inf. 3 (2009) 143–157.
[11] A. Ortigosa, J.M. Martín, R.M. Carro, Sentiment analysis in Facebook and its application to e-learning, Comput. Hum. Behav. 31 (2014) 527–541.
[12] V. García, J.S. Sánchez, R.A. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Knowl. Based Syst. 25 (2012) 13–21.
[13] P. McCullah, Regression models for ordinal data, J. R. Stat. Soc. 42 (1980) 109–142.
[14] S. Daskalaki, I. Kopanas, N. Avouris, Evaluation of classifiers for an uneven class distribution problem, Appl. Artif. Intell. 20 (2006) 381–417.
[15] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 27:1–27:27.
[16] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, J. Comput. Syst. Sci. 55 (1997) 119–139.
[17] B. Liu, Sentiment Analysis and Opinion Mining, Morgan & Claypool Publisher, 2012.
[18] J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: probabilistic models for segmenting and labeling sequence data, in: Proceedings of the 18th International Conference on Machine Learning, 2001.
[19] A. Kennedy, D. Inkpen, Sentiment classification of movie reviews using contextual valence shifters, Comput. Intell. 22 (2) (2006) 110–125.
[20] S. Li, S.Y.M. Lee, Y. Chen, C.-R. Huang, G. Zhou, Sentiment classification and polarity shifting, in: Proceedings of the 23rd International Conference on Computational Linguistics (CO LING -2010), 2010.
[21] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, Lexicon-based methods for sentiment analysis, computational linguistics, 37 (2) (2011) 267–307.
[22] K. Toutanova, D. Klein, C. Manning, Y. Singer, Feature-rich part-of-speech tagging with a cyclic dependency network, in: Proceedings of HLT-NAACL 2003, 2003, pp. 252–259.
[23] N. Kalchbrenner, E. Grefenstette, P. Blunsom, A sentence model based on convolutional neural networks, ACL, 2014.
[24] O. Appel, F. Chiclana, J. Carter, H. Fujita, A hybrid approach to the sentiment analysis problem at the sentence level, Knowl. Based Syst. 108 (2016) 110–124.
[25] A. Muhammad, N. Wiratunga, R. Lothian, Contextual sentiment analysis for social media genres, Knowl. Based Syst. 108 (2016) 92–101.
[26] T. Chen, R. Xu, Y. He, Y. Xia, X. Wang, Learning user and product distributed representations using a sequence model for sentiment analysis, IEEE Comput. Intell. Mag. 11 (3) (2016) 34–44.
[27] A. Esuli, F. Sebastiani, SentiWordNet – a publicly available lexical resource for opinion mining, in: Proceedings of the 5th Conference on Language Resources and Evaluation, 2006, pp. 417–422.
[28] S. Poria, E. Cambria, A. Gelbukh, Aspect extraction for opinion mining with a deep convolutional neural network, Knowl. Based Syst. 108 (2016) 42–49.