Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

# User Modeling with Neural Network for Review Rating Prediction

**Duyu Tang**[†], **Bing Qin**[†*], **Ting Liu**[†], **Yuekui Yang**[♮]
[†]Harbin Institute of Technology, Harbin, China
[♮]Intelligent Computing and Search Lab, Tencent, Shenzhen, China
{dytang, qinb, tliu}@ir.hit.edu.cn, yuekuiyang@tencent.com

## Abstract

We present a neural network method for review rating prediction in this paper. Existing neural network methods for sentiment prediction typically only capture the semantics of texts, but ignore the user who expresses the sentiment. This is not desirable for review rating prediction as each user has an influence on how to interpret the textual content of a review. For example, the same word (e.g. "*good*") might indicate different sentiment strengths when written by different users. We address this issue by developing a new neural network that takes user information into account. The intuition is to factor in user-specific modification to the meaning of a certain word. Specifically, we extend the lexical semantic composition models and introduce a user-word composition vector model (UWCVM), which effectively captures how user acts as a function affecting the continuous word representation. We integrate UWCVM into a supervised learning framework for review rating prediction, and conduct experiments on two benchmark review datasets. Experimental results demonstrate the effectiveness of our method. It shows superior performances over several strong baseline methods.

## 1 Introduction

Sentiment analysis and opinion mining [Pang and Lee, 2008; Liu, 2012] has attracted a lot of attentions from both industry and research communities in recent years. A fundamental problem in sentiment analysis is to inference the sentiment polarity (e.g. "*thumbs up*" or "*thumbs down*") of a document [Pang *et al.*, 2002]. In this paper, we target at a finer grained document-level problem, known as **review rating prediction** [Pang and Lee, 2005]. Given a review written by a user as input, it calls for inferring the author's evaluation with respect to a numeric ratings (e.g. one to five stars). Majority of existing studies follow Pang and Lee [2005] and cast this problem as a multiclass classification/regression task. They typically employ machine learning algorithms in a supervised learning manner, and build the rating predictor from reviews with accompanying ratings. Under this direction, most studies focus on designing effective context-level [Qu *et al.*, 2010] and user-level features [Gao *et al.*, 2013] for obtaining a better prediction performance.

Feature engineering is important but labor intensive. It is therefore desirable to extract and organize discriminative features (or representations) automatically from data [Bengio *et al.*, 2013]. For document-level sentiment prediction, an effective way is to learn continuous text representation with neural network. Existing neural network method typically learn continuous word representations (also known as word embeddings) from text corpus [Mikolov *et al.*, 2013; Pennington *et al.*, 2014], and then use them to calculate the representation of a document with semantic composition [Socher *et al.*, 2013; Kalchbrenner *et al.*, 2014; Kim, 2014; Li *et al.*, 2015]. Despite the apparent success of existing neural network methods, they are not effective enough if directly used for review rating prediction. The reason lies in that they typically only use textual semantics of words, but ignore the review author who expresses the sentiment. It is not desirable because different users may use different words to express sentiment, and the same word might indicate different meanings when it is written by different users. For example, a critical user might use "*good*" to express an excellent attitude, but a lenient user may use "*good*" to evaluate an ordinary product.

In this paper, we introduce a novel neural network method for review rating prediction by taking user information into account. The intuitive idea is to factor in user-specific modification to the meaning of a certain word. To this end, we extend existing lexical semantic composition methods [Clark *et al.*, 2008; Baroni and Zamparelli, 2010], and introduce a user-word composition vector model (**UWCVM**) to effectively incorporate user information. Specifically, we employ matrix-vector multiplication as the basic composition function of UWCVM. We represent each word as a continuous vector and each user as a matrix which maps the original word vector to the modified representation. Matrix-vector multiplication is tailored for this scenario, since it can be thought of as a matrix modifying a vector in the field of vector-based compositional semantics [Mitchell and Lapata, 2010]. We integrate UWCVM into a feed-forward neural network for review rating prediction, as illustrated in Figure 1. As is shown, a document composition vector model (**DCVM**) takes the mod-

*Corresponding author.

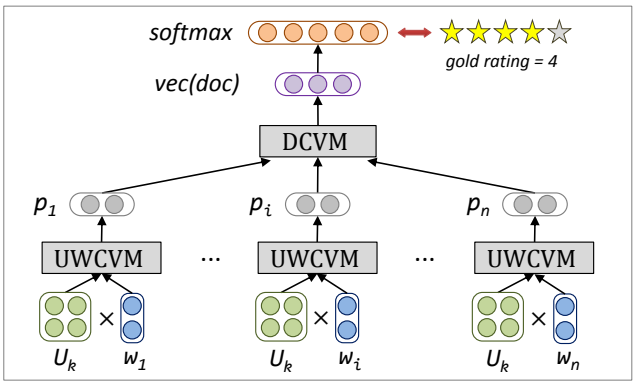


Figure 1: The proposed neural network method for review rating prediction. **UWCVM** means the user-word composition vector model that captures user-specific modification to word meanings. **DCVM** stands for the document composition vector model, which composes the document representation based on modified word vectors. $U$ and $w$ represent user and word, respectively. $p$ means the user modified word representation.

ified word vector as input and produces the representation of a review. The result of DCVM is leveraged as features to build the rating predictor without any feature engineering. The user representation and parameters of neural networks are trained in an end-to-end fashion with back propagation.

We conduct experiments to evaluate the effectiveness of the proposed method for review rating prediction. We use two benchmark datasets: one from movie reviews in Rotten Tomatoes and another from restaurant reviews in Yelp Dataset Challenge 2013. Extensive experimental results show that (1) the proposed method outperforms several strong baseline methods which only use textual semantics; (2) for the task of review rating prediction, matrix-vector multiplication is more effective to model user-word composition than vector concatenation or addition methods. The main contributions presented in this work are listed as follows:

- We represent user-word composition as matrix-vector multiplication, regarding each user as a matrix that modifies the meaning of a certain word.
- To our knowledge, this is the first neural network method that incorporates user information for review rating prediction.
- We report empirical results on two benchmark datasets. The proposed method performs better than strong baseline methods on the Yelp dataset.

## 2 Related Work

### 2.1 Review Rating Prediction

Review rating prediction is a fundamental task in sentiment analysis. It goes beyond the binary sentiment classification (*thumbs up* or *thumbs down*) and targets at predicting the numeric rating (e.g 1∼5 stars) of a given review. Pang and Lee [2005] pioneer this field by regarding review rating prediction as a classification/regression problem. They build the rating predictor with machine learning method under a supervised *metric labeling* framework. Following Pang and Lee [2005]'s work, most studies focus on designing effective textural features of reviews, since the performance of a rating predictor is heavily dependent on the choice of feature representation of data. For example, Qu et al. [2010] introduce the bag-of-opinion feature, which consists of sentiment, modification and negation words.

Beyond textural features, user information is also investigated in the literature of sentiment analysis. For review rating prediction, Gao et al. [2013] develop user-specific features to capture the user leniency; Li et al. [2014] incorporate textual topic and user-word factors through topic modeling. [Wu and Ester, 2015] leverage user information with a combination between collaborative filtering and aspect based opinion mining. Tan et al. [2011] use user information for Twitter sentiment analysis. Unlike most previous studies that use handcrafted textual or user-relevant features, we learn explanatory features automatically from data for review rating prediction. Unlike Li et al. [2014] and Diao et al. [Diao *et al.*, 2014] that models user information via topic modeling, we integrate user-word composition in a neural network approach.

### 2.2 Deep Learning for Sentiment Prediction

Deep learning has been proven to be effective for many sentiment analysis tasks [Socher *et al.*, 2013; Tang *et al.*, 2014a; Xu *et al.*, 2014]. For sentence/document sentiment prediction, the magic of deep learning is to learn continuous representations of texts with different grains (e.g. *word*, *phrase*, *sentence* and *document*). Existing neural network methods typically include two stages. They first learn word embedding[1] from text corpora, and then utilize semantic composition models [Mitchell and Lapata, 2010] to compose the representation of a document based on the representations of the words it contains. For learning word embedding, Mikolov et al. [2013] introduce a context-prediction method resulting in *word2vec*. Pennington et al. [2014] take consideration of global word-word co-occurrence. Maas et a. [2011] and Tang et al. [2014b] propose to learn sentiment-specific word vectors with topic modeling and neural networks, respectively. For learning semantic composition, Glorot et al. [2011] use stacked denoising autoencoder; Socher et al. [2013] introduce a family of recursive deep neural networks (RNN); Li [2014] extend Recursive Neural Network by using feature weight tuning to control how much one specific unit contributes to the higher-level representation; [Kalchbrenner *et al.*, 2014; Kim, 2014] use convolution neural networks; Le and Mikolov [2014] introduce Paragraph Vector. Li et al. [2015] compare the effectiveness of recursive neural network and recurrent neural network on five NLP tasks including sentiment classification.

Unlike most previous deep learning approaches that only consider the semantics of texts, we take user information into account. Our approach of modeling user-word composition via matrix-vector multiplication is inspired by the lexical composition models of [Clark *et al.*, 2008; Baroni and

[1]Word embedding is a continuous word representation that encodes each word in a low-dimensional and real valued vector.

Zamparelli, 2010; Socher *et al.*, 2012]. They regard the compositional modifier as a matrix, and use matrix-vector multiplication as the composition function. For example, Clark et al. [2008] learn adjective-noun composition. They represent words by vectors and adjectives by matrices which map the original noun representation to the modified representation. Continuous user representation is also exploited in [Kiros *et al.*, 2014; Perozzi *et al.*, 2014].

## 3 Methodology

In this section, we describe the proposed neural network method for review rating prediction. We give an overview of our method before presenting the details of two semantic composition models, UWCVM and DCVM. We then describe the use of our method for review rating prediction in a supervised *metric labeling* framework.

### 3.1 An Overview of the Neural Network Method

Given a review $r_{kj}$ comprised of $n$ words $\{w_1, w_2 \ ... \ w_n\}$ written by user $u_k$ as the input, review rating prediction aims at inferring the numeric rating (1∼4 or 1∼5 stars) of $r_{kj}$. We cast review rating prediction as a multi-class classification problem by inferring a discrete rating score.

An overview of the proposed neural network method is illustrated in Figure 1. As is shown, our method includes two composition models, the user-word composition vector model (**UWCVM**) and the document composition vector model (**DCVM**). UWCVM aims at modifying the original word vectors with user information. DCVM takes the modified word vectors as input, and produces review representation which is regarded as the feature for predicting review rating. We utilize existing machine learning algorithms to train the rating predictor in a supervised *metric labeling* framework [Pang and Lee, 2005].

### 3.2 User-Word Composition Vector Model

We describe UWCVM which models user-specific modification to the continuous representation of a word. To this end, a first attempt might consider learning user-specific word embeddings only from the texts expressed by a certain user. However, it is impractical as the parameter space is as huge as $\mathbb{R}^{d\times|V_w|\times|V_u|}$, where $d$ is the dimension of each word vector, $|V_w|$ and $|V_u|$ are the sizes of the word vocabulary and user vocabulary, respectively. Another downside is that there might be not enough contexts to effectively train the user-specific word embeddings for inactive users.

We explore vector-based compositional semantics and model user modification to word meaning with a computational composition approach in this paper. Under this perspective, additive and multiplicative composition functions are representative solutions [Mitchell and Lapata, 2010]. Given two vectors $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ as the input, **additive** composition assumes that the output vector $\boldsymbol{p}$ is a linear function of Cartesian product of $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$, as described below.

$$\boldsymbol{p} = \boldsymbol{A} \times \boldsymbol{v_1} + \boldsymbol{B} \times \boldsymbol{v_2} \tag{1}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are the matrices parameters that encode the contributes of $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ to $p$. Weighted sum $\boldsymbol{p} = \alpha\boldsymbol{v_1} + \beta\boldsymbol{v_2}$ and addition $\boldsymbol{p} = \boldsymbol{v_1} + \boldsymbol{v_2}$ are simpler cases of additive composition functions. **Multiplicative** composition assumes that the output $\boldsymbol{p}$ is a linear function of the tensor product of $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$, as shown below.

$$\boldsymbol{p} = \boldsymbol{T} \times \boldsymbol{v_1} \times \boldsymbol{v_2} = \boldsymbol{U_1} \times \boldsymbol{v_2} \tag{2}$$

where $\boldsymbol{T}$ is a tensor of rank 3 that projects the tensor product of $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ to $\boldsymbol{p}$. The partial product of $\boldsymbol{T}$ with $\boldsymbol{v_1}$ can be considered as producing a matrix $\boldsymbol{U_1}$.

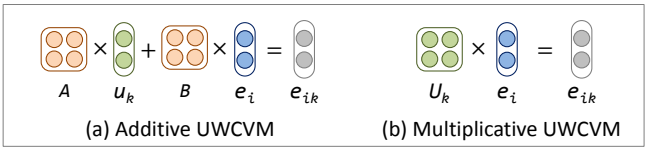


Figure 2: The additive and multiplicative composition functions as **UWCVM**.

In this paper, we exploit multiplicative composition function as the user-word composition model[2], as illustrated in Figure 2(b). The reason is that multiplicative composition can be regarded as one component $\boldsymbol{U_1}$ modifying another $\boldsymbol{v_2}$, which exactly meet our needs of user modifying word meaning. It has been successfully leveraged to model adjective-noun composition [Clark *et al.*, 2008; Baroni and Zamparelli, 2010] and adverb-adjective composition [Socher *et al.*, 2012].

Formally, we represent each word $w_i$ as a continuous vector $\boldsymbol{e_i} \in \mathbb{R}^d$, and each user $u_k$ as a matrix $\boldsymbol{U_k} \in \mathbb{R}^{d\times d}$, where $d$ is the dimension of continuous vector. In practice, the number of the parameters is still too large if we encode each user by a $d \times d$ matrix for the common vector length $d = 50$ or 100. To reduce the parameter size, we represent user representation by a low-rank plus diagonal approximation: $\boldsymbol{U_k} = \boldsymbol{U_{k1}} \times \boldsymbol{U_{k2}} + diag(\boldsymbol{u'})$, where $\boldsymbol{U_{k1}} \in \mathbb{R}^{d\times r}$, $\boldsymbol{U_{k2}} \in \mathbb{R}^{r\times d}$, $\boldsymbol{u'} \in \mathbb{R}^d$. We regard $\boldsymbol{u'}$ as a shared background representation for each user. It is tailored for the Out-Of-Vocabulary situation, where a user in testing process is never seen in the training data. After conducting matrix-vector multiplication, we append an activation layer (e.g. $tanh$) for adding the non-linearity property. Accordingly, the final modified word vector $\boldsymbol{p_i}$ for the original word vector $\boldsymbol{e_i}$ is calculated as:

$$\begin{aligned}\boldsymbol{p_i} &= tanh(\boldsymbol{e_{ik}}) = tanh(\boldsymbol{U_k} \times \boldsymbol{e_i}) \\ &= tanh((\boldsymbol{U_{k1}} \times \boldsymbol{U_{k2}} + diag(\boldsymbol{u'})) \times \boldsymbol{e_i})\end{aligned} \tag{3}$$

### 3.3 Document Composition Vector Model

Document composition vector model (DCVM) takes the user-modified word vectors as input, and produces the representation for each review/document. Document-level semantic composition is an important research topic in sentiment analysis, and many neural models have been proposed in the literature [Socher *et al.*, 2013; Kalchbrenner *et al.*, 2014]. However, it is out of the scope of this work to compare them. In

[2]We also tried additive composition functions by representing both words and users as vectors, as shown in Figure 2(a).

this paper, we exploit a simple and effective approach [Hermann and Blunsom, 2014], which recursively uses $biTanh$ function to produce the document representation.

$$biTanh(\boldsymbol{p}) = \sum_{i=1}^{n} tanh(\boldsymbol{p}_{i-1} + \boldsymbol{p}_i) \quad (4)$$

Specifically, we first use $biTanh$ to calculate the vector for each sentence by regarding the user-modified word vectors as input. We then feed the sentence vectors to $biTanh$ for generating the document vector $\boldsymbol{vec(doc)}$. Essentially, the recursive use of $biTanh$ can be viewed as two pairs of bag-of-word convolutional neural network, whose window size is two and parameters are clamped as addition and $tanh$.

### 3.4 Rating Prediction with Metric Labeling

We apply the learned review representation to review rating prediction in a supervised *metric labeling* framework [Pang and Lee, 2005]. It consists of two cascaded stages. **In the first stage**, we train an initial predictor by only using the representation of a certain user-review pair. In the experiment, we use $softmax$ to predict the probabilities for classes (e.g. one to five stars). It is calculated as $softmax_i = \frac{exp(\boldsymbol{z}_i)}{\sum_{i'} exp(\boldsymbol{z}_{i'})}$, where $\boldsymbol{z} \in \mathbb{R}^C$ is a linear vector transformed from the user-enhanced review representation $\boldsymbol{vec(doc)}$.

$$\boldsymbol{z} = \boldsymbol{W} \times \boldsymbol{vec(doc)} + \boldsymbol{b} \quad (5)$$

where $\boldsymbol{W} \in \mathbb{R}^{C \times d}$ and $\boldsymbol{b} \in \mathbb{R}^C$ are the parameters, $C$ is the number of the rating classes. We define $f(r, l)$ as the probability of predicting review $r$ as rating $l$. For each review $r$, we use cross-entropy as the training objective function:

$$L(r) = -\sum_{l \in L_s} f^g(r, l) \cdot log(f(r, l)) + \lambda_\theta \cdot |\boldsymbol{\theta}|_F^2 \quad (6)$$

where $L_s$ is the set of possible rating classes, $\boldsymbol{f^g}$ is the gold rating distribution[3] and $\boldsymbol{f}$ is the predicted rating distribution. $|\boldsymbol{\theta}|_F^2 = \sum_i \boldsymbol{\theta}_i^2$ is a Frobenius norm regularization term and $\boldsymbol{\theta} = [\boldsymbol{U_{k1}}; \boldsymbol{U_{k2}}; \boldsymbol{u'}; \boldsymbol{W}; \boldsymbol{b}]$ stands for the parameters.

**In the second stage**, we apply the initial classifier and explicitly encode the idea of "*similar items, similar labels*" with *metric labeling* [Pang and Lee, 2005]. Let $dist(l_1, l_2) = |l_1 - l_2|$ be a distance metric between labels $l_1$ and $l_2$, and let $nn(r_{kj})$ be the $M$ nearest neighbors of $r_{kj}$ according to a review similarity function $sim$. The objective of *metric labeling* is to minimize the following equation:

$$\sum_{r_{kj}}^{T} [-f(r_{kj}, l_{r_{kj}}) + \lambda_{nn} \cdot \sum_{r' \in nn(r_{kj})} dist(l_{r_{kj}}, l_{r'}) \cdot sim(r_{kj}, r')]$$

where $T$ is the dev dataset of user-review pairs and $\lambda_{nn}$ is the trade-off parameter. We use cosine similarity between the learned review representations as $sim$.

An advantage of our method is that: the learned review representation can be not only regarded as the feature of $r_{kj}$ to build the initial predictor, but also leveraged to calculate the similarity between reviews $sim(r_{kj}, r')$ without using any hand-crafted features.

[3]The gold rating distribution of a review has a *1-of-K* coding scheme. It has the same dimension as the number of rating classes, and only the dimension corresponding to the ground truth is 1, with all others being 0.

### 3.5 Model Training

We train the rating predictor in a supervised learning framework from the reviews with accompanying ratings. We take the derivative of the loss with respect to the whole set of parameters through back-propagation, and use stochastic gradient descent with mini-batch to update the parameters. Word vectors are learned with word2vec[4]. We empirically set the vector dimension $d$ as 100, the rank of user matrix $r$ as 3. The values of $W$, $b$ and $u'$ are randomly initialized with the fan-in trick. We use *dropout* [Srivastava *et al.*, 2014] to avoid the neural network being over-fitting. Hyper parameters are tuned on the development dataset.

## 4 Experiment

We conduct experiments for review rating prediction to empirically evaluate the proposed method. We describe the experiment setting and the results in this section.

### 4.1 Experiment Setting

We conduct experiments on two benchmark datasets, *Yelp13* and *RT05*. *Yelp13* is a large-scale dataset consisting of restaurant reviews from Yelp. It is released by the third round of the Yelp Dataset Challenge in 2013. *RT05* is a movie review dataset downloaded from Rotten Tomatoes. The statistical information of *Yelp13* and *RT05* are detailed in Table 1. For *Yelp13* dataset, human labeled ratings are regarded as gold standards for model training[5].

| Dataset | #users | #reviews | scale | $len_{avg}$ | $\lvert V \rvert$ |
|---|---|---|---|---|---|
| *Yelp13* | 70,817 | 335,018 | 1∼5 | 75.1 | 137,816 |
| *RT05* | 4 | 5,006 | 1∼4 | 429.1 | 55,449 |

Table 1: Statistical information of datasets. #users and #reviews are the number of users and reviews, respectively. #$len_{avg}$ is the average length of the review in each dataset, $|V|$ is the vocabulary size of words.

We conduct experiments in a supervised learning framework. On *Yelp13*, we split the original corpus into train, dev and test sets with a 80:10:10 split. We train the rating predictor on the training set, tune parameters on the dev set and evaluate on the test set. On *RT05*, we use 10-fold cross-validation as in previous studies. We conduct multi-label classification on these two datasets. Since rating scores stand for sentiment intensities, we use mean absolute error (MAE) and root mean squared error (RMSE) as the evaluation metrics (as in other work like [Li *et al.*, 2014]) to measure the divergences between predicted ratings and gold ratings.

$$MAE = \frac{\sum_i |gold_i - pred_i|}{N}$$

$$RMSE = \sqrt{\frac{\sum_i (gold_i - pred_i)^2}{N}}$$

[4]https://code.google.com/p/word2vec/

[5]We do not consider the cases that rating does not match with review texts [Zhang *et al.*, 2014].

## 4.2 Baseline Methods

We compare our method with the following baseline methods for review rating prediction:

• *Majority*: It is a heuristic method that assigns the majority rating score in the training set to each review in the test dataset.

• *BOW*: We represent each review with bag-of-words (BOW) [Pang and Lee, 2005], and build the rating predictor with Supported Vector Machine[6].

• *BOW+BOO*: Qu et al. [2010] propose to represent each document with bag-of-opinion (BOO). We use the concatenation of BOW and BOO as features. The sentiment lexicons are from BingLiu[7] and MPQA [Wilson *et al.*, 2005]. The modifier and negation words come from the Sentiment Symposium Tutorial. We train the rating predictor with SVM.

• *VecAvg*: We calculate the representation of a review by averaging the vectors of the words it contains. We use the word vectors learned from word2vec, and build the classifier with SVM [Fan *et al.*, 2008].

• *RAE*: Recursive AutoEncoder (RAE) has proven effective to learn compositionality for sentiment analysis. We train RAE using the word vectors pre-trained with word2vec. We do not compare with *RNTN* [Socher *et al.*, 2013] because it depends on a parsed tree structure, which cannot be accurately obtained for the document-level reviews.

• *PVDM*: Le and Mikolov [2014] propose the Distributed Memory Model of Paragraph Vectors (PVDM), which is a state-of-the-art performer on several sentiment analysis benchmark datasets. We set the window size of PVDM as 9 in the experiments.

• *CNN*: Convolution neural network is a state-of-the-art performer on sentence-level sentiment analysis tasks [Kalchbrenner *et al.*, 2014; Kim, 2014].

## 4.3 Results and Analysis

Table 2 shows the experimental results of the baseline methods as well as our method on two datasets. Our neural network method that uses user-word composition is abbreviated as UWRL. UWRL$^\dagger$ stands for our neural network method plus metric labeling.

From Table 2, we can see that the performances of these methods are consistent on two datasets. *Majority* performs very poor as it does not capture any text-level or user-level information. *BOW* only uses the surface form of words in the review. However, it loses the ordering of words and it also ignores the semantics of words. *BOW+BOO* performs slightly better than *BOW* because *BOO* benefits from the sentiment, negation and modifier words from external resources. We also run standard collective filtering baseline on Yelp dataset. However, its performance is poor and comparable with bag-of-word baseline.

*VecAvg* is a straight-forward method that uses word embeddings as the features without any feature engineering. From Table 2, we find that *VecAvg* does not yield obvious improvement over the traditional *BOW*. The reason is that the *average* function loses the word orders and does not well capture the complex linguistic phenomena in sentiment analysis. We also compare with several sophisticated composition methods including *RAE*, *CNN* and *PVDM*, and find that all of them outperform the *VecAvg* baseline. *CNN* is the strongest baseline on both datasets. The results indicate the importance of semantic composition for review rating prediction. The proposed method *UWRL* slightly outperforms text-based neural network algorithms as we simultaneously capture text-level and user-level semantics (p-value $< 0.05$ with t-test on *Yelp13* between *CNN* and *UWRL*). After incorporating *metric labeling*, *UWRL*$^\dagger$ captures the idea of "*similar items, similar labels*" and thus obtains further improvements.

| Method | Yelp13 | | RT05 | |
|---|---|---|---|---|
| | MAE | RMSE | MAE | RMSE |
| Majority | 1.232 | 1.626 | 0.724 | 0.984 |
| BOW | 0.787 | 1.218 | 0.562 | 0.833 |
| BOW + BOO | 0.731 | 1.124 | 0.553 | 0.819 |
| VecAvg | 0.759 | 1.176 | 0.561 | 0.826 |
| RAE | 0.700 | 1.104 | 0.521 | 0.798 |
| PVDM | 0.698 | 1.098 | 0.516 | 0.793 |
| CNN | 0.644 | 0.986 | 0.483 | 0.759 |
| UWRL | **0.626** | **0.973** | **0.469** | **0.752** |
| UWRL$^\dagger$ | **0.618** | **0.962** | **0.464** | **0.748** |

Table 2: Experimental results (lower is better) for review rating prediction on two benchmark datasets. Our method is abbreviated as UWRL and UWRL$^\dagger$.

## 4.4 The Effect of User-Word Composition

We investigate the effect of different user-word composition functions for review rating prediction. We compare the matrix-vector multiplication function (*mvMultip*) with the following strategies: *No-User*: $p = e_i$ , *Concat*: $p = [e_i; u_k]$, *Average*: $p = 1/2 \cdot (e_i + u_k)$ , *ElemMultip*: $p = e_i \odot u_k$, *WAdd*: $p = A \times e_i + B \times u_k$, where $e_i$ and $u_k$ stand for the word vector and user vector, respectively. We conduct experiments on the development set of *Yelp13*. The results are given in Figure 3.

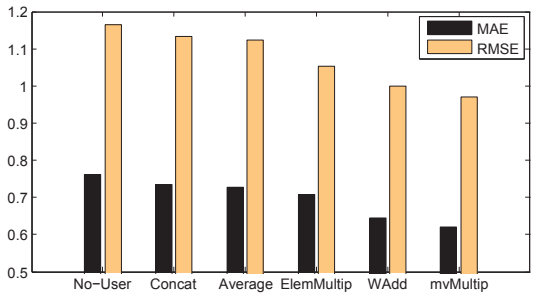


Figure 3: Experimental results on the development dataset of *Yelp13* with different user-word compositions.

We can find that capturing user information always out-

[6]In this experiment, we use SVM as baseline because the it performs better than the discretized regression [Pang and Lee, 2005] with a set of fixed decision thresholds {e.g. 0.5, 1.5, 2.5, ...}.

[7]http://www.cs.uic.edu/liub/FBS/sentiment-analysis.html

performs *No-User*, which shows the effectiveness of user information for review rating prediction. The performances of *Concat* and *Average* are relatively low because they do not well exploit the interaction between the user vector and word vector. Among all these composition functions, matrix-vector multiplication yields the best performance.

### 4.5 The Effect of User Activity

We explore the effect of the user activity for review rating prediction on the development set of *Yelp13*. We use the entire training dataset to train the model, and test on several subsets of the development dataset that correspond to different user activities. For example, the tick "100" on x-axis means that we only test on the users that have posted no less than 100 reviews.

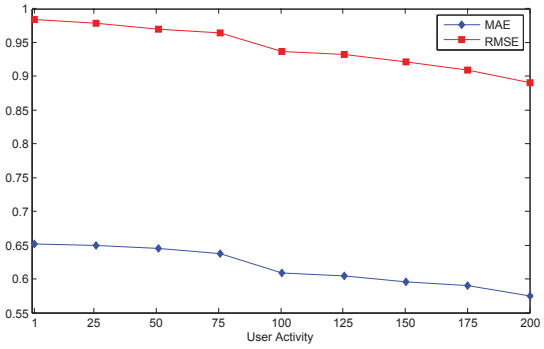


Figure 4: Experimental results on the development dataset of *Yelp13* with different user activity thresholds.

The results are illustrated in Figure 4. We can see that the performance of rating prediction consistently improves when larger user activity threshold is considered. This is because the user representations can be better estimated when more user-relevant reviews are utilized.

### 4.6 The Effect of Word Embedding

We investigate the effect of word embedding on review rating prediction. We try the randomly initialized word vectors (*Random*), the word vectors learned from *SkipGram* and the sentiment-specific word embeddings learned from *SSWE* [Tang *et al.*, 2014b] and *SSPE* [Tang *et al.*, 2014a].

From Figure 5, we find that all pre-trained word vectors outperform randomly initialized word vectors. Compared with *SkipGram*, three *SSWE* methods do not yield significant improvements. This is caused by the fact that *SSWE* assign the document polarity to each word sequence it contains for training the word vectors. The assumption is reasonable for tweets as they are short, but it is unsuitable for the document-level reviews where negation and contrast phenomenons are frequently appeared. *SSPE* performs slightly better than others, as it optimizes the word vectors by using a global document vector to predict the sentiment of a review. However, sentiment embeddings do not obtain significant performance boost than word2vec in this experiment. This calls for more powerful algorithms to learn sentiment embeddings from document level corpus.

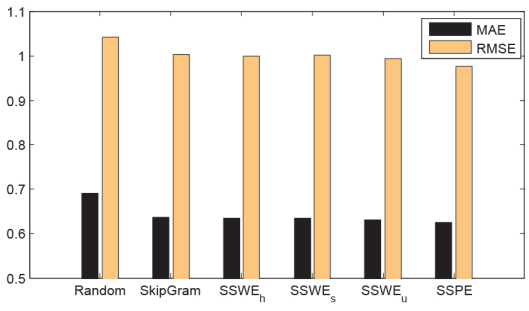


Figure 5: Experimental results on the development dataset of *Yelp13* with different word embeddings.

## 5 Conclusion

We introduce a neural network method that incorporates user information for review rating prediction in this paper. We model user-specific modification to the meaning of a certain word with a user-word composition vector model (UWCVM), and investigate both additive and multiplicative composition functions for UWCVM. We show that matrix-vector multiplication is more effective than vector concatenation or addition methods for review rating prediction. We conduct experiments on two benchmark datasets, and compare against multiple baseline methods. Experimental results show that, the proposed method performs better than several strong baseline methods which only use textural semantics.

## Acknowledgments

We gratefully acknowledge the fruitful discussions with Yaming Sun, Jing Liu, Nan Yang, Yongfeng Zhang and Wei Song. We thank the anonymous reviewers for their helpful feedbacks. This work was partly supported by National Natural Science Foundation of China (No. 61133012 and No. 61273321), the National High Technology Development 863 Program of China (No. 2015AA015407). Duyu Tang also thanks Baidu Fellowship and IBM Ph.D. Fellowship programs for their supports.

## References

[Baroni and Zamparelli, 2010] Marco Baroni and Roberto Zamparelli. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of EMNLP*, 2010.

[Bengio *et al.*, 2013] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. PAMI*, 2013.

[Clark *et al.*, 2008] Stephen Clark, Bob Coecke, and Mehrnoosh Sadrzadeh. A compositional distributional model of meaning. In *Quantum Interaction*, 2008.

[Diao *et al.*, 2014] Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J Smola, Jing Jiang, and Chong Wang. Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In *SIGKDD*. ACM, 2014.

[Fan *et al.*, 2008] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *JMLR*, 2008.

[Gao *et al.*, 2013] Wenliang Gao, Naoki Yoshinaga, Nobuhiro Kaji, and Masaru Kitsuregawa. Modeling user leniency and product popularity for sentiment classification. *Proceedings of IJCNLP*, 2013.

[Glorot *et al.*, 2011] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. domain adaptation for large-scale sentiment classification:a deep learning approach. *ICML*, 2011.

[Hermann and Blunsom, 2014] Karl Moritz Hermann and Phil Blunsom. Multilingual models for compositional distributed semantics. In *Proceedings of ACL*, 2014.

[Kalchbrenner *et al.*, 2014] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A sentence model based on convolutional neural networks. In *ACL*. 2014.

[Kim, 2014] Yoon Kim. Convolutional neural networks for sentence classification. In *Conference on EMNLP*, pages 1746–1751, 2014.

[Kiros *et al.*, 2014] Ryan Kiros, Richard Zemel, and Ruslan R Salakhutdinov. A multiplicative model for learning distributed text-based attribute representations. In *NIPS*, pages 2348–2356, 2014.

[Le and Mikolov, 2014] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. *Proceeding of ICML*, 2014.

[Li *et al.*, 2014] Fangtao Li, Sheng Wang, Shenghua Liu, and Ming Zhang. Suit: A supervised user-item based topic model for sentiment analysis. In *AAAI*, 2014.

[Li *et al.*, 2015] Jiwei Li, Dan Jurafsky, and Eudard Hovy. When are tree structures necessary for deep learning of representations? *arXiv preprint:1503.00185*, 2015.

[Li, 2014] Jiwei Li. Feature weight tuning for recursive neural networks. *Arxiv preprint*, 1412.3714, 2014.

[Liu, 2012] Bing Liu. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167, 2012.

[Maas *et al.*, 2011] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the ACL*, 2011.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *The Conference on NIPS*, 2013.

[Mitchell and Lapata, 2010] Jeff Mitchell and Mirella Lapata. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429, 2010.

[Pang and Lee, 2005] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*, 2005.

[Pang and Lee, 2008] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, 2008.

[Pang *et al.*, 2002] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *EMNLP*, 2002.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *SIGKDD*, pages 701–710. ACM, 2014.

[Qu *et al.*, 2010] Lizhen Qu, Georgiana Ifrim, and Gerhard Weikum. The bag-of-opinions method for review rating prediction from sparse text patterns. In *COLING*, 2010.

[Socher *et al.*, 2012] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *Proceedings of EMNLP*, 2012.

[Socher *et al.*, 2013] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on EMNLP*, pages 1631–1642, 2013.

[Srivastava *et al.*, 2014] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15, 2014.

[Tan *et al.*, 2011] Chenhao Tan, Lillian Lee, Jie Tang, Long Jiang, Ming Zhou, and Ping Li. User-level sentiment analysis incorporating social networks. In *SIGKDD*, 2011.

[Tang *et al.*, 2014a] Duyu Tang, Furu Wei, Bing Qin, Ming Zhou, and Ting Liu. Building large-scale twitter-specific sentiment lexicon: A representation learning approach. In *COLING*, pages 172–182, 2014.

[Tang *et al.*, 2014b] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL*, pages 1555–1565, 2014.

[Wilson *et al.*, 2005] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *EMNLP*, 2005.

[Wu and Ester, 2015] Yao Wu and Martin Ester. Flame: A probabilistic model combining aspect based opinion mining and collaborative filtering. In *WSDM*, pages 199–208. ACM, 2015.

[Xu *et al.*, 2014] Liheng Xu, Kang Liu, and Jun Zhao. Joint opinion relation detection using one-class deep neural network. In *COLING*, pages 677–687, 2014.

[Zhang *et al.*, 2014] Yongfeng Zhang, Haochen Zhang, Min Zhang, Yiqun Liu, and Shaoping Ma. Do users rate or review?: boost phrase-level sentiment labeling with review-level sentiment classification. In *SIGIR*, pages 1027–1030. ACM, 2014.