Learning Semantic Representations of Users and Products for Document Level Sentiment Classification

Duyu Tang, Bing Qin*, Ting Liu
Harbin Institute of Technology, Harbin, China
{dytang, qinb, tliu}@ir.hit.edu.cn

Abstract

Neural network methods have achieved promising results for sentiment classification of text. However, these models only use semantics of texts, while ignoring users who express the sentiment and products which are evaluated, both of which have great influences on interpreting the sentiment of text. In this paper, we address this issue by incorporating user- and product- level information into a neural network approach for document level sentiment classification. Users and products are modeled using vector space models, the representations of which capture important global clues such as individual preferences of users or overall qualities of products. Such global evidence in turn facilitates embedding learning procedure at document level, yielding better text representations. By combining evidence at user-, product- and document-level in a unified neural framework, the proposed model achieves state-of-the-art performances on IMDB and Yelp datasets[1].

1 Introduction

Document-level sentiment classification is a fundamental problem in the field of sentiment analysis and opinion mining (Pang and Lee, 2008; Liu, 2012). The task is to infer the sentiment polarity or intensity (e.g. 1-5 or 1-10 stars on review sites) of a document. Dominating studies follow Pang et al. (2002; 2005) and regard this problem as a multi-class classification task. They usually use machine learning algorithms, and build sentiment classifier from documents with accompanying sentiment labels. Since the performance of a machine learner is heavily dependent on the choice of data representations (Domingos, 2012), many works focus on designing effective features (Pang et al., 2002; Qu et al., 2010; Kiritchenko et al., 2014) or learning discriminative features from data with neural networks (Socher et al., 2013; Kalchbrenner et al., 2014; Le and Mikolov, 2014).

Despite the apparent success of neural network methods, they typically only use text information while ignoring the important influences of users and products. Let us take reviews with respect to 1-5 rating scales as an example. A critical user might write a review "*it works great*" and mark 4 stars, while a lenient user might give 5 stars even if he posts an (almost) identical review. In this case, user preference affects the sentiment rating of a review. Product quality also has an impact on review sentiment rating. Reviews towards high-quality products (e.g. *Macbook*) tend to receive higher ratings than those towards low-quality products. Therefore, it is feasible to leverage individual preferences of users and overall qualities of products to build a smarter sentiment classifier and achieve better performance[2].

In this paper, we propose a new model dubbed User Product Neural Network (**UPNN**) to capture user- and product-level information for sentiment classification of documents (e.g. reviews). UPNN takes as input a variable-sized document as well as the user who writes the review and the product which is evaluated. It outputs sentiment polarity label of a document. Users and products are encoded in continuous vector spaces, the representations of which capture important global clues such

*Corresponding author.

[1]The codes and datasets are available at http://ir.hit.edu.cn/~dytang/

[2]One can manually design a small number of user and product features (Gao et al., 2013). However, we argue that they are not effective enough to capture sophisticated semantics of users and products.

as user preferences and product qualities. These representations are further integrated with continuous text representation in a unified neural framework for sentiment classification.

We apply UPNN to three datasets derived from IMDB and Yelp Dataset Challenge. We compare to several neural network models including recursive neural networks (Socher et al., 2013), paragraph vector (Le and Mikolov, 2014), sentiment-specific word embedding (Tang et al., 2014b), and a state-of-the-art recommendation algorithm JMARS (Diao et al., 2014). Experimental results show that: (1) UPNN outperforms baseline methods for sentiment classification of documents; (2) incorporating representations of users and products significantly improves classification accuracy. The main contributions of this work are as follows:

• We present a new neural network method (UPNN) by leveraging users and products for document-level sentiment classification.

• We validate the influences of users and products in terms of sentiment and text on massive IMDB and Yelp reviews.

• We report empirical results on three datasets, and show that UPNN outperforms state-of-the-art methods for sentiment classification.

2 Consistency Assumption Verification

We detail the effects of users and products in terms of sentiment (e.g. 1-5 rating stars) and text, and verify them on review datasets.

We argue that the influences of users and products include the following four aspects.

• **user-sentiment consistency**. A user has specific preference on providing sentiment ratings. Some users favor giving higher ratings like 5 stars and some users tend to give lower ratings. In other words, sentiment ratings from the same user are more consistent than those from different users.

• **product-sentiment consistency**. Similar with user-sentiment consistency, a product also has its "preference" to receive different average ratings on account of its overall quality. Sentiment ratings towards the same product are more consistent than those towards different products.

• **user-text consistency**. A user likes to use personalized sentiment words when expressing opinion polarity or intensity. For example, a strict user might use "*good*" to express an excellent attitude, but a lenient user may use "*good*" to evaluate an ordinary product.

Algorithm 1 Consistency Assumption Testing

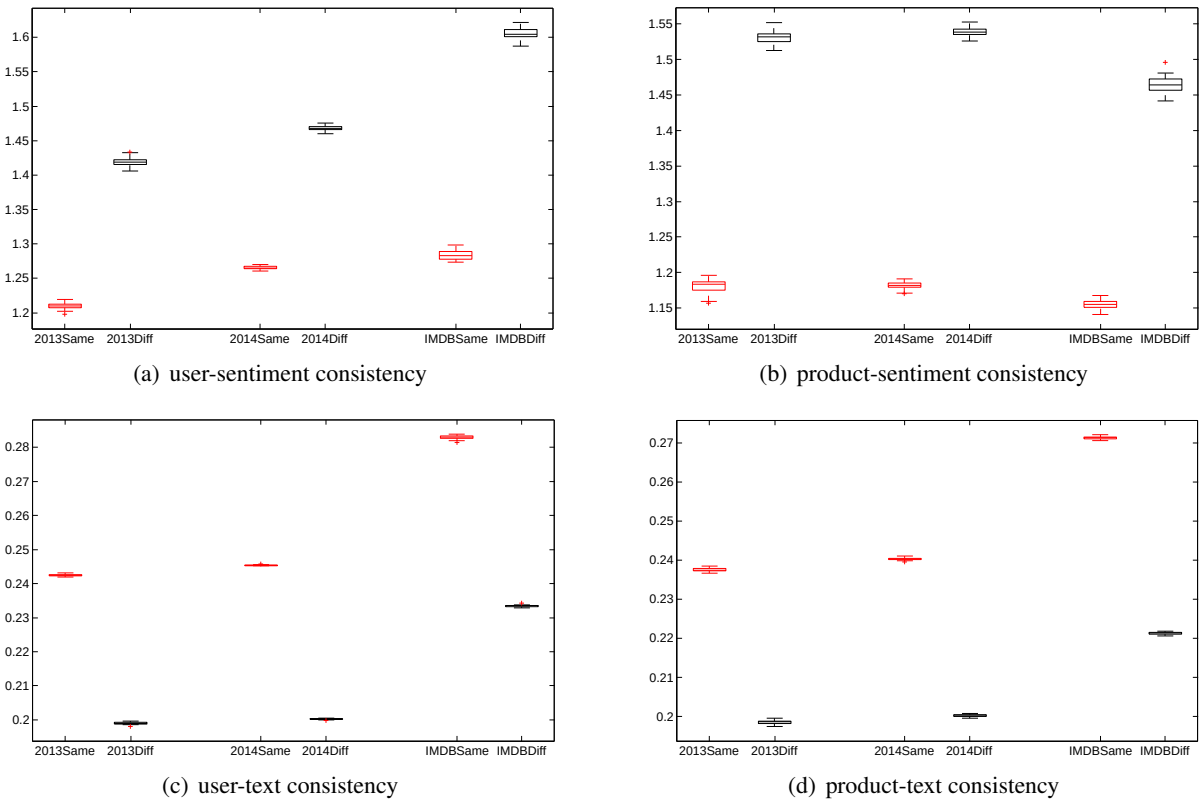
Input: data X, number of users/products m, number of iterations n
Output: $meaSame_k$, $meaDiff_k$, $1 \leq k \leq n$
for $k = 1$ **to** n **do**
  $meaSame_k = 0$, $meaSame_k = 0$
  for $i = 1$ **to** m **do**
    Sample x_i, x_i^+, x_i^- from X
    $meaSame_k$ += $measure(x_i, x_i^+)$
    $meaDiff_k$ += $measure(x_i, x_i^-)$
  end for
  $meaSame_k$ /= m, $meaDiff_k$ /= m
end for

• **product-text consistency**. Similar with user-text consistency, a product also has a collection of product-specific words suited to evaluate it. For example, people prefer using "*sleek*" and "*stable*" to evaluate a smartphone, while like to use "*wireless*" and "*mechanical*" to evaluate a keyboard.

We test four consistency assumptions mentioned above with the same testing criterion, which is formalized in Algorithm 1. For each consistency assumption, we test it for n = 50 iterations on each of IMDB, Yelp Dataset Challenge 2013 and 2014 datasets. Taking user-sentiment consistency as an example, in each iteration, we randomly select two reviews x_i, x_i^+ written by the same user u_i, and a review x_i^- written by another randomly selected user. Afterwards, we calculate the measurements of $(x_i,\ x_i^+)$ and $(x_i,\ x_i^-)$, and aggregate these statistics for m users. In user-sentiment assumption test, we use absolute rating difference $||rating_a - rating_b||$ as the measurement between two reviews a and b. We illustrate the results in Figure 1 (a)[3], where $2013same/2014same/amzsame$ (red plots) means that two reviews are written by a same user, and $2013diff/2014diff/amzdiff$ (black plots) means that two reviews are written by different users. We can find that: the absolute rating differences between two reviews written by a same user are lower than those written by different users (t-test with p-value < 0.01). In other words, sentiment ratings from the same user are more consistent than those from different users. This validates the user-sentiment consistency.

For testing product-sentiment consistency, we

[3]Since the rating scale of IMDB (1-10) is different from Yelp (1-5), we divide the rating difference of IMDB reviews by two for better visualizing and analyzing the results.



(a) user-sentiment consistency

(b) product-sentiment consistency

(c) user-text consistency

(d) product-text consistency

Figure 1: Assumption testing of user-sentiment, product-sentiment, user-text and product-text consistencies. We test them on the datasets from IMDB and Yelp Dataset Challenge in 2013 and 2014.

use absolute rating difference as the measurement. The reviews x_i, x_i^+ are towards a same product p_i, and x_i^- is towards another randomly selected product. From Figure 1 (b), we can see that sentiment ratings towards the same product are more consistent than those towards different products. In order to verify the assumptions of user-text and product-text consistencies, we use cosine similarity between bag-of-words of two reviews as the measurement. Results are given in Figure 1 (c) and (d). We can see that the textual similarity between two reviews written by a same user (or towards a same product) are higher than those written by different users (or towards different products).

3 User Product Neural Network (UPNN) for Sentiment Classification

We present the details of User Product Neural Network (UPNN) for sentiment classification. An illustration of UPNN is given in Figure 2. It takes as input a *review*, the *user* who posts the review, and the *product* which is evaluated. UPNN captures four kinds of consistencies which are verified in Section 2. It outputs the sentiment category (e.g. 1-5 stars) of a review by considering not only the semantics of review text, but also the information of user and product. In following subsections, we first describe the use of neural network for modeling semantics of variable-sized documents. We then present the methods for incorporating user and product information, followed by the use of UPNN in a supervised learning framework for sentiment classification.

3.1 Modeling Semantics of Document

We model the semantics of documents based on the principle of compositionality (Frege, 1892), which states that the meaning of a longer expression (e.g. a sentence or a document) comes from the meanings of its words and the rules used to combine them. Since a document consists of a list of sentences and each sentence is made up of a list of words, we model the semantic representation of a document in two stages. We first produce continuous vector of each sentence from word representations. Afterwards, we feed sentence vectors as inputs to compose document representation.

For modeling the semantics of words, we represent each word as a low dimensional, continu-

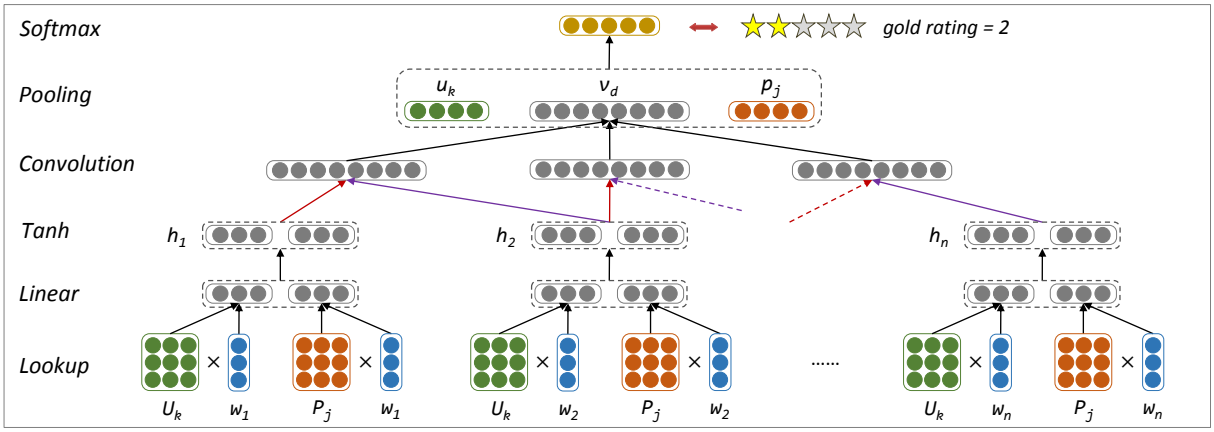


Figure 2: An illustration of the neural network approach for sentiment classification. w_i means the i-th word of a review text. u_k and p_j are continuous vector representations of user k and product j for capturing user-sentiment and product-sentiment consistencies. U_k and P_j are continuous matrix representations of user k and product j for capturing user-text and product-text consistencies.

ous and real-valued vector, also known as word embedding (Bengio et al., 2003). All the word vectors are stacked in a word embedding matrix $L_w \in \mathbb{R}^{d \times |V|}$, where d is the dimension of word vector and $|V|$ is the size of word vocabulary. These word vectors can be randomly initialized from a uniform distribution, regarded as a parameter and jointly trained with other parameters of neural networks. Alternatively, they can be pre-trained from text corpus with embedding learning algorithms (Mikolov et al., 2013; Pennington et al., 2014; Tang et al., 2014b), and applied as initial values of word embedding matrix. We adopt the latter strategy which better exploits the semantic and grammatical associations of words.

To model semantic representations of sentences, convolutional neural network (CNN) and recursive neural network (Socher et al., 2013) are two state-of-the-art methods. We use CNN (Kim, 2014; Kalchbrenner et al., 2014) in this work as it does not rely on external parse tree. Specifically, we use multiple convolutional filters with different widths to produce sentence representation. The reason is that they are capable of capturing local semantics of n-grams of various granularities, which are proven powerful for sentiment classification. The convolutional filter with a width of 3 essentially captures the semantics of trigrams in a sentence. Accordingly, multiple convolutional filters with widths of 1, 2 and 3 encode the semantics of unigrams, bigrams and trigrams in a sentence.

An illustration of CNN with three convolutional filters is given in Figure 3. Let us denote a sentence consisting of n words as $\{w_1, w_2, ...w_i, ...w_n\}$. Each word w_i is mapped to its embedding representation $e_i \in \mathbb{R}^d$. A convolutional filter is a list of linear layers with shared parameters. Let l_{cf} be the width of a convolutional filter, and let W_{cf}, b_{cf} be the shared parameters of linear layers in the filter. The input of a linear layer is the concatenation of word embeddings in a fixed-length window size l_{cf}, which is denoted as $I_{cf} = [e_i; e_{i+1}; ...; e_{i+l_{cf}-1}] \in \mathbb{R}^{d \cdot l_{cf}}$. The output of a linear layer is calculated as

$$O_{cf} = W_{cf} \cdot I_{cf} + b_{cf} \quad (1)$$

where $W_{cf} \in \mathbb{R}^{len \times d \cdot l_{cf}}$, $b_{cf} \in \mathbb{R}^{len}$, len is the output length of linear layer. In order to capture the global semantics of a sentence, we feed the output of a convolutional filter to an *average* pooling layer, resulting in an output vector with fixed-length. We further add hyperbolic tangent functions ($tanh$) to incorporate element-wise nonlinearity, and *fold* (*average*) their outputs to generate sentence representation.

We feed sentence vectors as the input of an *average* pooling layer to obtain the document representation. Alternative document modeling approaches include CNN or recurrent neural network. However, we prefer *average* pooling for its computational efficiency and good performance in our experiment.

3.2 Modeling Semantics of Users and Products

We integrate semantic representations of users and products in UPNN to capture user-sentiment,

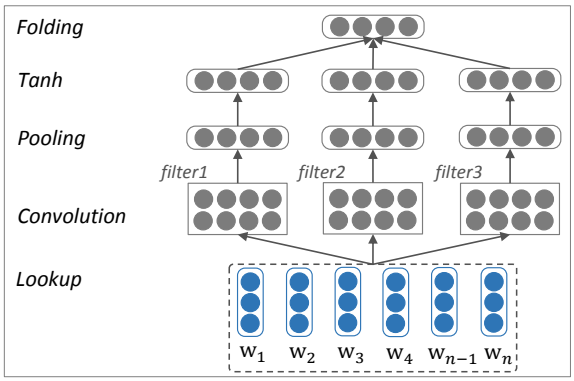


Figure 3: Convolutional neural network with multiple convolutional filters for sentence modeling.

product-sentiment, user-text and product-text consistencies.

For modeling **user-sentiment** and **product-sentiment** consistencies, we embed each user as a continuous vector $u_k \in \mathbb{R}^{d_u}$ and embed each product as a continuous vector $p_j \in \mathbb{R}^{d_p}$, where d_u and d_p are dimensions of user vector and product vector, respectively. The basic idea behind this is to map users with similar rating preferences (e.g. prefer assigning 4 stars) into close vectors in user embedding space. Similarly, the products which receive similar averaged ratings are mapped into neighboring vectors in product embedding space.

In order to model **user-text** consistency, we represent each user as a continuous matrix $U_k \in \mathbb{R}^{d_U \times d}$, which acts as an operator to modify the semantic meaning of a word. This is on the basis of vector based semantic composition (Mitchell and Lapata, 2010). They regard compositional modifier as a matrix X_1 to modify another component x_2, and use matrix-vector multiplication $y = X_1 \times x_2$ as the composition function. Multiplicative semantic composition is suitable for our need of user modifying word meaning, and it has been successfully utilized to model adjective-noun composition (Clark et al., 2008; Baroni and Zamparelli, 2010) and adverb-adjective composition (Socher et al., 2012). Similarly, we model **product-text** consistency by encoding each product as a matrix $P_j \in \mathbb{R}^{d_P \times d}$, where d is the dimension of word vector, d_P is the output length of product-word multiplicative composition. After conducting user-word multiplication and product-word multiplication operations, we concatenate their outputs and feed them to CNN (detailed in Section 3.1) for producing user and product enhanced document representation.

3.3 Sentiment Classification

We apply UPNN to document level sentiment classification under a supervised learning framework (Pang and Lee, 2005). Instead of using hand-crafted features, we use continuous representation of documents, users and products as discriminative features. The sentiment classifier is built from documents with gold standard sentiment labels.

As is shown in Figure 2, the feature representation for building rating predictor is the concatenation of three parts: continuous user representation u_k, continuous product representation p_j and continuous document representation v_d, where v_d encodes user-text consistency, product-text consistency and document level semantic composition. We use $softmax$ to build the classifier because its outputs can be interpreted as conditional probabilities. $Softmax$ is calculated as given in Equation 2, where C is the category number (e.g. 5 or 10).

$$softmax_i = \frac{exp(x_i)}{\sum_{i'=1}^{C} exp(x_{i'})} \quad (2)$$

We regard cross-entropy error between gold sentiment distribution and predicted sentiment distribution as the loss function of $softmax$. We take the derivative of loss function through back-propagation with respect to the whole set of parameters $\theta = [W_{cf}^{1,2,3}; b_{cf}^{1,2,3}; u_k; p_j; U_k; P_j; W_{softmax}, b_{softmax}]$, and update parameters with stochastic gradient descent. We set the widths of three convolutional filters as 1, 2 and 3. We learn 200-dimensional sentiment-specific word embeddings (Tang et al., 2014b) on each dataset separately, randomly initialize other parameters from a uniform distribution $U(-0.01, 0.01)$, and set learning rate as 0.03.

4 Experiment

We conduct experiments to evaluate UPNN by applying it to sentiment classification of documents.

4.1 Experimental Setting

Existing benchmark datasets for sentiment classification such as Stanford Sentiment Treebank (Socher et al., 2013) typically only have text information, but do not contain users who express the sentiment or products which are evaluated. Therefore, we build the datasets by ourselves. In order to obtain large scale corpora without manual annotation, we derive three datasets from IMDB (Diao

Dataset	#users	#products	#reviews	#docs/user	#docs/product	#sents/doc	#words/doc
IMDB	1,310	1,635	84,919	64.82	51.94	16.08	394.6
Yelp 2014	4,818	4,194	231,163	47.97	55.11	11.41	196.9
Yelp 2013	1,631	1,633	78,966	48.42	48.36	10.89	189.3

Table 1: Statistical information of IMDB, Yelp 2014 and Yelp 2013 datasets used for sentiment classification. The rating scale of IMDB dataset is 1-10. The rating scale of Yelp 2014 and Yelp 2013 datasets is 1-5. $|V|$ is the vocabulary size of words in each dataset. #users is the number of users, #docs/user means the average number of documents per user posts in the corpus.

et al., 2014) and Yelp Dataset Challenge[4] in 2013 and 2014. Statistical information of the generated datasets are given in Table 1.

We split each corpus into training, development and testing sets with a 80/10/10 split, and conduct tokenization and sentence splitting with Stanford CoreNLP (Manning et al., 2014). We use standard *accuracy* (Manning and Schütze, 1999; Jurafsky and Martin, 2000) to measure the overall sentiment classification performance, and use MAE and $RMSE$ to measure the divergences between predicted sentiment ratings (pr) and ground truth ratings (gd).

$$MAE = \frac{\sum_i |gd_i - pr_i|}{N} \quad (3)$$

$$RMSE = \sqrt{\frac{\sum_i (gd_i - pr_i)^2}{N}} \quad (4)$$

4.2 Baseline Methods

We compare UPNN with the following baseline methods for document-level sentiment classification.

(1) *Majority* is a heuristic baseline method, which assigns the majority sentiment category in training set to each review in the test dataset.

(2) In *Trigram*, we use unigrams, bigrams and trigrams as features and train classifier with supported vector machine (SVM) (Fan et al., 2008).

(3) In *TextFeature*, we implement hand-crafted text features including word/character ngrams, sentiment lexicon features, negation features, etc al. (Kiritchenko et al., 2014).

(4) We extract user-leniency features (Gao et al., 2013) and corresponding product features (denoted as *UPF*) from training data, and concatenate them with the features in baseline (2) and (3).

(5) We learn word embeddings from training and development sets with *word2vec* (Mikolov et al., 2013), average word embeddings to get document representation, and train a SVM classifier.

(6) We learn sentiment-specific word embeddings (SSWE) from training and development sets, and use max/min/average pooling (Tang et al., 2014b) to generate document representation.

(7) We represent sentence with RNTN (Socher et al., 2013) and compose document representation with recurrent neural network. We average hidden vectors of recurrent neural network as the features for sentiment classification.

(8) We re-implement PVDM in Paragraph Vector (Le and Mikolov, 2014) because its codes are not officially provided. The window size is tuned on development set.

(9) We compare with a state-of-the-art recommendation algorithm JMARS (Diao et al., 2014), which leverages user and aspects of a review with collaborative filtering and topic modeling.

4.3 Model Comparisons

Experimental results are given in Table 2. The results are separated into two groups: the methods above only use texts of review, and the methods below also use user and product information.

From the first group, we can see that majority performs very poor because it does not capture any text or user information. SVM classifiers with trigrams and hand-crafted text features are powerful for document level sentiment classification and hard to beat. We compare the word embedding learnt from each corpus with off-the-shell general word embeddings[5]. Results show that tailored word embedding from each corpus performs slightly better than general word embeddings (about 0.01 improvement in terms of accuracy). SSWE performs better than context-based word embedding by incorporating sentiment information of texts. Setting a large window size (e.g. 15) is crucial for effectively training SSWE from documents with accompanying senti-

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

[5] We compare with Glove embeddings learnt from Wikipedia and Twitter http://nlp.stanford.edu/projects/glove/

	IMDB			Yelp 2014			Yelp 2013		
	Acc	MAE	RMSE	Acc	MAE	RMSE	Acc	MAE	RMSE
Majority	0.196	1.838	2.495	0.392	0.779	1.097	0.411	0.744	1.060
Trigram	0.399	1.147	1.783	0.577	0.487	0.804	0.569	0.513	0.814
TextFeature	0.402	1.134	1.793	0.572	0.490	**0.800**	0.556	0.520	0.845
AvgWordvec + SVM	0.304	1.361	1.985	0.530	0.562	0.893	0.526	0.568	0.898
SSWE + SVM	0.312	1.347	1.973	0.557	0.523	0.851	0.549	0.529	0.849
Paragraph Vector	0.341	1.211	1.814	0.564	0.496	0.802	0.554	0.515	0.832
RNTN + Recurrent	0.400	1.133	1.764	0.582	**0.478**	0.821	0.574	0.489	**0.804**
UPNN (no UP)	**0.405**	**1.030**	**1.629**	**0.585**	0.483	0.808	**0.577**	**0.485**	0.812
Trigram + UPF	0.404	1.132	1.764	0.576	0.471	0.789	0.570	0.491	0.803
TextFeature + UPF	0.402	1.129	1.774	0.579	0.476	0.791	0.561	0.509	0.822
JMARS	N/A	1.285	1.773	N/A	0.710	0.999	N/A	0.699	0.985
UPNN (full)	**0.435**	**0.979**	**1.602**	**0.608**	**0.447**	**0.764**	**0.596**	**0.464**	**0.784**

Table 2: Sentiment classification on IMDB, Yelp 2014 and Yelp 2013 datasets. Evaluation metrics are accuracy (Acc, higher is better), MAE (lower is better) and RMSE (lower is better). Our full model is UPNN (full). Our model without using user and product information is abbreviated as UPNN (no UP). The best method in each group is in **bold**.

ment labels. *RNTN+Reccurent* is a strong performer by effectively modeling document representation with semantic composition. Our text based model (UPNN no UP) performs slightly better than *RNTN+Reccurent*, trigram and text features.

From the second group, we can see that concatenating user product feature (*UPF*) with existing feature sets does not show significant improvements. This is because the dimension of existing feature sets is typically huge (e.g. 1M trigram features in Yelp 2014), so that concatenating a small number of *UPF* features does not have a great influence on the whole model. We do not evaluate JMARS in terms of *accuracy* because JMARS outputs real-valued ratings. Our full model UPNN yields the best performance on all three datasets. Incorporating semantic representations of user and product significantly (t-test with p-value $<$ 0.01) boosts our text based model (UPNN no UP). This shows the effectiveness of UPNN over standard trigrams and hand-crafted features when incorporating user and product information.

4.4 Model Analysis: Effect of User and Product Representations

We investigate the effects of vector based user and product representations (u_k, p_j) as well as matrix based user and product representations (U_k, P_j) for sentiment classification. We remove vector based representations (u_k, p_j) and matrix based representations (U_k, P_j) from UPNN separately, and conduct experiments on three datasets. From Table 3, we can find that vector based representations (u_k, p_j) are more effective than matrix based representations (U_k, P_j). This is because u_k and p_j encode user-sentiment and product-sentiment consistencies, which are more directly associated with sentiment labels than user-text (U_k) and product-text (P_j) consistencies. Another reason might be that the parameters of vector representations are less than the matrix representations, so that the vector representations are better estimated. We also see the contribution from each of user and product by removing (U_k, u_k) and (P_j, p_j) separately. Results are given in Table 3. It is interesting to find that user representations are obviously more effective than product representations for review rating prediction.

4.5 Discussion: Out-Of-Vocabulary Users and Products

Out-of-vocabulary (OOV) situation occurs if a user or a product in testing/decoding process is never seen in training data. We give two natural solutions (avg UP and unk UP) to deal with OOV users and products. One solution (avg UP) is to regard the averaged representations of users/products in training data as the representation of OOV user/product. Another way (unk UP) is to learn a shared "unknown" user/product representation for low-frequency users in training data, and apply it to OOV user/product.

	IMDB			Yelp 2014			Yelp 2013		
	Acc	MAE	RMSE	Acc	MAE	RMSE	Acc	MAE	RMSE
UPNN (full)	0.435	0.979	1.602	0.608	0.447	0.764	0.596	0.464	0.784
UPNN $- u_k - p_j$	0.409	1.021	1.622	0.585	0.483	0.808	0.572	0.491	0.823
UPNN $- U_k - P_j$	0.426	0.993	1.607	0.597	0.465	0.789	0.585	0.482	0.802
UPNN $- U_k - u_k$	0.324	1.209	1.743	0.577	0.475	0.778	0.566	0.505	0.828
UPNN $- P_j - p_j$	0.397	1.075	1.712	0.595	0.462	0.776	0.590	0.476	0.802

Table 3: Influence of user and product representations. For user k and product j, u_k and p_j are their continuous vector representations, U_k and P_j are their continuous matrix representations (see Figure 2).

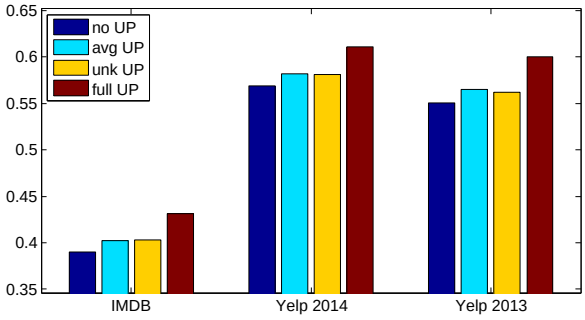


Figure 4: Accuracy of OOV user and product on OOV test set.

In order to evaluate the two strategies for OOV problem, we randomly select 10 percent users and products from each development set, and mask their user and product information. We run avg UP, unk UP together with UPNN (no UP) which only uses text information, and UPNN (full) which learns tailored representation for each user and product. We evaluate classification accuracy on the extracted OOV test set. Experimental results are given in Figure 5. We can find that these two strategies perform slightly better than UPNN (no UP), but still worse than the full model.

5 Related Work

5.1 Sentiment Classification

Sentiment classification is a fundamental problem in sentiment analysis, which targets at inferring the sentiment label of a document. Pang and Lee (2002; 2005) cast this problem a classification task, and use machine learning method in a supervised learning framework. Goldberg and Zhu (2006) use unlabelled reviews in a graph-based semi-supervised learning method. Many studies design effective features, such as text topic (Ganu et al., 2009), bag-of-opinion (Qu et al., 2010) and sentiment lexicon features (Kiritchenko et al., 2014). User information is also used for sentiment classification. Gao et al. (2013) design user-specific features to capture user leniency. Li et al. (2014) incorporate textual topic and user-word factors with supervised topic modeling. Tan et al. (2011) and Hu et al. (2013) utilize user-text and user-user relations for Twitter sentiment analysis. Unlike most previous studies that use hand-crafted features, we learn discriminative features from data. We differ from Li et al. (2014) in that we encode four kinds of consistencies and use neural network approach. User representation is also leveraged for recommendation (Weston et al., 2013), web search (Song et al., 2014) and social media analytics (Perozzi et al., 2014).

5.2 Neural Network for Sentiment Classification

Neural networks have achieved promising results for sentiment classification. Existing neural network methods can be divided into two groups: word embedding and semantic composition. For learning word embeddings, (Mikolov et al., 2013; Pennington et al., 2014) use local and global contexts, (Maas et al., 2011; Labutov and Lipson, 2013; Tang et al., 2014b; Tang et al., 2014a; Zhou et al., 2015) further incorporate sentiment of texts. 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). RNN is extended with adaptive composition functions (Dong et al., 2014), global feedbackward (Paulus et al., 2014), feature weight tuning (Li, 2014), and also used for opinion relation detection (Xu et al., 2014). Li et al. (2015) compare the effectiveness of recursive neural network and recurrent neural network on five NLP tasks including sentiment classification. (Kalchbrenner et al., 2014; Kim, 2014; Johnson and Zhang, 2014) use convolutional neural networks. Le and Mikolov (2014) introduce

Paragraph Vector. Unlike existing neural network approaches that only use the semantics of texts, we take consideration of user and product representations and leverage their connections with text semantics for sentiment classification. This work is an extension of our previous work (Tang et al., 2015), which only takes consideration of user-word association.

6 Conclusion

In this paper, we introduce User Product Neural Network (UPNN) for document level sentiment classification under a supervised learning framework. We validate user-sentiment, product-sentiment, user-text and product-text consistencies on massive reviews, and effectively integrate them in UPNN. We apply the model to three datasets derived from IMDB and Yelp Dataset Challenge. Empirical results show that: (1) UPNN outperforms state-of-the-art methods for document level sentiment classification; (2) incorporating continuous user and product representations significantly boosts sentiment classification accuracy.

Acknowledgments

The authors give great thanks to Furu Wei, Lei Cui, Nan Yang, Jiwei Li, Yaming Sun, Mao Zheng and anonymous reviewers for their valuable comments. We would like to thank Qiming Diao for providing the IMDB dataset as well as the codes of JMARS. This work was 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 is supported by Baidu Fellowship and IBM Ph.D. Fellowship.

References

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *EMNLP*, pages 1183–1193.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Stephen Clark, Bob Coecke, and Mehrnoosh Sadrzadeh. 2008. A compositional distributional model of meaning. In *Quantum Interaction Symposium*, pages 133–140.

Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J Smola, Jing Jiang, and Chong Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In *SIGKDD*, pages 193–202. ACM.

Pedro Domingos. 2012. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87.

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2014. Adaptive multi-compositionality for recursive neural models with applications to sentiment analysis. In *AAAI*, pages 1537–1543.

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

Gottlob Frege. 1892. On sense and reference. *Ludlow (1997)*, pages 563–584.

Gayatree Ganu, Noemie Elhadad, and Amélie Marian. 2009. Beyond the stars: Improving rating predictions using review text content. In *WebDB*.

Wenliang Gao, Naoki Yoshinaga, Nobuhiro Kaji, and Masaru Kitsuregawa. 2013. Modeling user leniency and product popularity for sentiment classification. In *ICJNLP*, pages 1107–1111.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, pages 513–520.

Andrew B Goldberg and Xiaojin Zhu. 2006. Seeing stars when there aren't many stars: graph-based semi-supervised learning for sentiment categorization. In *GraphBased Method for NLP*, pages 45–52.

Xia Hu, Lei Tang, Jiliang Tang, and Huan Liu. 2013. Exploiting social relations for sentiment analysis in microblogging. In *WSDM*, pages 537–546.

Rie Johnson and Tong Zhang. 2014. Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint: 1412.1058*.

Dan Jurafsky and James H Martin. 2000. *Speech & language processing*. Pearson Education India.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *ACL*, pages 655–665.

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

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. 2014. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, pages 723–762.

Igor Labutov and Hod Lipson. 2013. Re-embedding words. In *Annual Meeting of the Association for Computational Linguistics*.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *ICML*, pages 1188–1196.

Fangtao Li, Sheng Wang, Shenghua Liu, and Ming Zhang. 2014. Suit: A supervised user-item based topic model for sentiment analysis. In *AAAI*, pages 1636–1642.

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

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

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

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *ACL*, pages 142–150.

Christopher D Manning and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*. MIT press.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL*, pages 55–60.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119.

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

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*, pages 115–124.

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

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *EMNLP*, pages 79–86.

Romain Paulus, Richard Socher, and Christopher D Manning. 2014. Global belief recursive neural networks. In *NIPS*, pages 2888–2896.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

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

Lizhen Qu, Georgiana Ifrim, and Gerhard Weikum. 2010. The bag-of-opinions method for review rating prediction from sparse text patterns. In *COLING*, pages 913–921.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *EMNLP*, pages 1201–1211.

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

Yang Song, Hongning Wang, and Xiaodong He. 2014. Adapting deep ranknet for personalized search. In *WSDM*, pages 83–92. ACM.

Chenhao Tan, Lillian Lee, Jie Tang, Long Jiang, Ming Zhou, and Ping Li. 2011. User-level sentiment analysis incorporating social networks. In *SIGKDD*, pages 1397–1405. ACM.

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

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

Duyu Tang, Bing Qin, Ting Liu, and Yuekui Yang. 2015. User modeling with neural network for review rating prediction. In *IJCAI*.

Jason Weston, Ron J Weiss, and Hector Yee. 2013. Nonlinear latent factorization by embedding multiple user interests. In *RecSys*, pages 65–68. ACM.

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

Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. 2015. Representation learning for aspect category detection in online reviews. In *AAAI*.