Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification*

Duyu Tang[†], Furu Wei[‡] , Nan Yang[♮], Ming Zhou[‡], Ting Liu[†], Bing Qin[†]
[†]Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
[‡]Microsoft Research, Beijing, China
[♮]University of Science and Technology of China, Hefei, China
{dytang, tliu, qinb}@ir.hit.edu.cn
{fuwei, v-nayang, mingzhou}@microsoft.com

Abstract

We present a method that learns word embedding for Twitter sentiment classification in this paper. Most existing algorithms for learning continuous word representations typically only model the syntactic context of words but ignore the sentiment of text. This is problematic for sentiment analysis as they usually map words with similar syntactic context but opposite sentiment polarity, such as *good* and *bad*, to neighboring word vectors. We address this issue by learning sentiment-specific word embedding (**SSWE**), which encodes sentiment information in the continuous representation of words. Specifically, we develop three neural networks to effectively incorporate the supervision from sentiment polarity of text (e.g. sentences or tweets) in their loss functions. To obtain large scale training corpora, we learn the sentiment-specific word embedding from massive distant-supervised tweets collected by positive and negative emoticons. Experiments on applying SSWE to a benchmark Twitter sentiment classification dataset in SemEval 2013 show that (1) the SSWE feature performs comparably with hand-crafted features in the top-performed system; (2) the performance is further improved by concatenating SSWE with existing feature set.

1 Introduction

Twitter sentiment classification has attracted increasing research interest in recent years (Jiang et al., 2011; Hu et al., 2013). The objective is to classify the sentiment polarity of a tweet as positive, negative or neutral. The majority of existing approaches follow Pang et al. (2002) and employ machine learning algorithms to build classifiers from tweets with manually annotated sentiment polarity. Under this direction, most studies focus on designing effective features to obtain better classification performance. For example, Mohammad et al. (2013) build the top-performed system in the Twitter sentiment classification track of SemEval 2013 (Nakov et al., 2013), using diverse sentiment lexicons and a variety of hand-crafted features.

Feature engineering is important but labor-intensive. It is therefore desirable to discover explanatory factors from the data and make the learning algorithms less dependent on extensive feature engineering (Bengio, 2013). For the task of sentiment classification, an effective feature learning method is to compose the representation of a sentence (or document) from the representations of the words or phrases it contains (Socher et al., 2013b; Yessenalina and Cardie, 2011). Accordingly, it is a crucial step to learn the word representation (or word embedding), which is a dense, low-dimensional and real-valued vector for a word. Although existing word embedding learning algorithms (Collobert et al., 2011; Mikolov et al., 2013) are intuitive choices, they are not effective enough if directly used for sentiment classification. The most serious problem is that traditional methods typically model the syntactic context of words but ignore the sentiment information of text. As a result, words with opposite polarity, such as *good* and *bad*, are mapped into close vectors. It is meaningful for some tasks such as pos-tagging (Zheng et al., 2013) as the two words have similar usages and grammatical roles, but it becomes a disaster for sentiment analysis as they have the opposite sentiment polarity.

In this paper, we propose learning sentiment-specific word embedding (**SSWE**) for sentiment analysis. We encode the sentiment information in-

* This work was done when the first and third authors were visiting Microsoft Research Asia.

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pages 1555–1565,
Baltimore, Maryland, USA, June 23-25 2014.

to the continuous representation of words, so that it is able to separate *good* and *bad* to opposite ends of the spectrum. To this end, we extend the existing word embedding learning algorithm (Collobert et al., 2011) and develop three neural networks to effectively incorporate the supervision from sentiment polarity of text (e.g. sentences or tweets) in their loss functions. We learn the sentiment-specific word embedding from tweets, leveraging massive tweets with emoticons as distant-supervised corpora without any manual annotations. These automatically collected tweets contain noises so they cannot be directly used as gold training data to build sentiment classifiers, but they are effective enough to provide weakly supervised signals for training the sentiment-specific word embedding.

We apply SSWE as features in a supervised learning framework for Twitter sentiment classification, and evaluate it on the benchmark dataset in SemEval 2013. In the task of predicting positive/negative polarity of tweets, our method yields 84.89% in macro-F1 by only using SSWE as feature, which is comparable to the top-performed system based on hand-crafted features (84.70%). After concatenating the SSWE feature with existing feature set, we push the state-of-the-art to 86.58% in macro-F1. The quality of SSWE is also directly evaluated by measuring the word similarity in the embedding space for sentiment lexicons. In the accuracy of polarity consistency between each sentiment word and its top N closest words, SSWE outperforms existing word embedding learning algorithms.

The major contributions of the work presented in this paper are as follows.

- We develop three neural networks to learn sentiment-specific word embedding (SSWE) from massive distant-supervised tweets without any manual annotations;
- To our knowledge, this is the first work that exploits word embedding for Twitter sentiment classification. We report the results that the SSWE feature performs comparably with hand-crafted features in the top-performed system in SemEval 2013;
- We release the sentiment-specific word embedding learned from 10 million tweets, which can be adopted off-the-shell in other sentiment analysis tasks.

2 Related Work

In this section, we present a brief review of the related work from two perspectives, Twitter sentiment classification and learning continuous representations for sentiment classification.

2.1 Twitter Sentiment Classification

Twitter sentiment classification, which identifies the sentiment polarity of short, informal tweets, has attracted increasing research interest (Jiang et al., 2011; Hu et al., 2013) in recent years. Generally, the methods employed in Twitter sentiment classification follow traditional sentiment classification approaches. The lexicon-based approaches (Turney, 2002; Ding et al., 2008; Taboada et al., 2011; Thelwall et al., 2012) mostly use a dictionary of sentiment words with their associated sentiment polarity, and incorporate negation and intensification to compute the sentiment polarity for each sentence (or document).

The learning based methods for Twitter sentiment classification follow Pang et al. (2002)'s work, which treat sentiment classification of texts as a special case of text categorization issue. Many studies on Twitter sentiment classification (Pak and Paroubek, 2010; Davidov et al., 2010; Barbosa and Feng, 2010; Kouloumpis et al., 2011; Zhao et al., 2012) leverage massive noisy-labeled tweets selected by positive and negative emoticons as training set and build sentiment classifiers directly, which is called *distant supervision* (Go et al., 2009). Instead of directly using the distant-supervised data as training set, Liu et al. (2012) adopt the tweets with emoticons to smooth the language model and Hu et al. (2013) incorporate the emotional signals into an unsupervised learning framework for Twitter sentiment classification.

Many existing learning based methods on Twitter sentiment classification focus on feature engineering. The reason is that the performance of sentiment classifier being heavily dependent on the choice of feature representation of tweets. The most representative system is introduced by Mohammad et al. (2013), which is the state-of-the-art system (the top-performed system in SemEval 2013 Twitter Sentiment Classification Track) by implementing a number of hand-crafted features. Unlike the previous studies, we focus on learning discriminative features automatically from massive distant-supervised tweets.

2.2 Learning Continuous Representations for Sentiment Classification

Pang et al. (2002) pioneer this field by using bag-of-word representation, representing each word as a one-hot vector. It has the same length as the size of the vocabulary, and only one dimension is 1, with all others being 0. Under this assumption, many feature learning algorithms are proposed to obtain better classification performance (Pang and Lee, 2008; Liu, 2012; Feldman, 2013). However, the one-hot word representation cannot sufficiently capture the complex linguistic characteristics of words.

With the revival of interest in deep learning (Bengio et al., 2013), incorporating the continuous representation of a word as features has been proving effective in a variety of NLP tasks, such as parsing (Socher et al., 2013a), language modeling (Bengio et al., 2003; Mnih and Hinton, 2009) and NER (Turian et al., 2010). In the field of sentiment analysis, Bespalov et al. (2011; 2012) initialize the word embedding by Latent Semantic Analysis and further represent each document as the linear weighted of ngram vectors for sentiment classification. Yessenalina and Cardie (2011) model each word as a matrix and combine words using iterated matrix multiplication. Glorot et al. (2011) explore Stacked Denoising Autoencoders for domain adaptation in sentiment classification. Socher et al. propose Recursive Neural Network (RNN) (2011b), matrix-vector RNN (2012) and Recursive Neural Tensor Network (RNTN) (2013b) to learn the compositionality of phrases of any length based on the representation of each pair of children recursively. Hermann et al. (2013) present Combinatory Categorial Autoencoders to learn the compositionality of sentence, which marries the Combinatory Categorial Grammar with Recursive Autoencoder.

The representation of words heavily relies on the applications or tasks in which it is used (Labutov and Lipson, 2013). This paper focuses on learning sentiment-specific word embedding, which is tailored for sentiment analysis. Unlike Maas et al. (2011) that follow the probabilistic document model (Blei et al., 2003) and give an sentiment predictor function to each word, we develop neural networks and map each ngram to the sentiment polarity of sentence. Unlike Socher et al. (2011c) that utilize manually labeled texts to learn the meaning of phrase (or sentence) through compositionality, we focus on learning the meaning of word, namely word embedding, from massive distant-supervised tweets. Unlike Labutov and Lipson (2013) that produce task-specific embedding from an existing word embedding, we learn sentiment-specific word embedding from scratch.

3 Sentiment-Specific Word Embedding for Twitter Sentiment Classification

In this section, we present the details of learning sentiment-specific word embedding (**SSWE**) for Twitter sentiment classification. We propose incorporating the sentiment information of sentences to learn continuous representations for words and phrases. We extend the existing word embedding learning algorithm (Collobert et al., 2011) and develop three neural networks to learn SSWE. In the following sections, we introduce the traditional method before presenting the details of SSWE learning algorithms. We then describe the use of SSWE in a supervised learning framework for Twitter sentiment classification.

3.1 C&W Model

Collobert et al. (2011) introduce C&W model to learn word embedding based on the syntactic contexts of words. Given an ngram *"cat chills on a mat"*, C&W replaces the center word with a random word w^r and derives a **corrupted** ngram *"cat chills w^r a mat"*. The training objective is that the original ngram is expected to obtain a higher language model score than the corrupted ngram by a margin of 1. The ranking objective function can be optimized by a hinge loss,

$$loss_{cw}(t, t^r) = max(0, 1 - f^{cw}(t) + f^{cw}(t^r)) \quad (1)$$

where t is the original ngram, t^r is the corrupted ngram, $f^{cw}(\cdot)$ is a one-dimensional scalar representing the language model score of the input ngram. Figure 1(a) illustrates the neural architecture of C&W, which consists of four layers, namely $lookup \rightarrow linear \rightarrow hTanh \rightarrow linear$ (from bottom to top). The original and corrupted ngrams are treated as inputs of the feed-forward neural network, respectively. The output f^{cw} is the language model score of the input, which is calculated as given in Equation 2, where L is the lookup table of word embedding, w_1, w_2, b_1, b_2 are the parameters of linear layers.

$$f^{cw}(t) = w_2(a) + b_2 \quad (2)$$

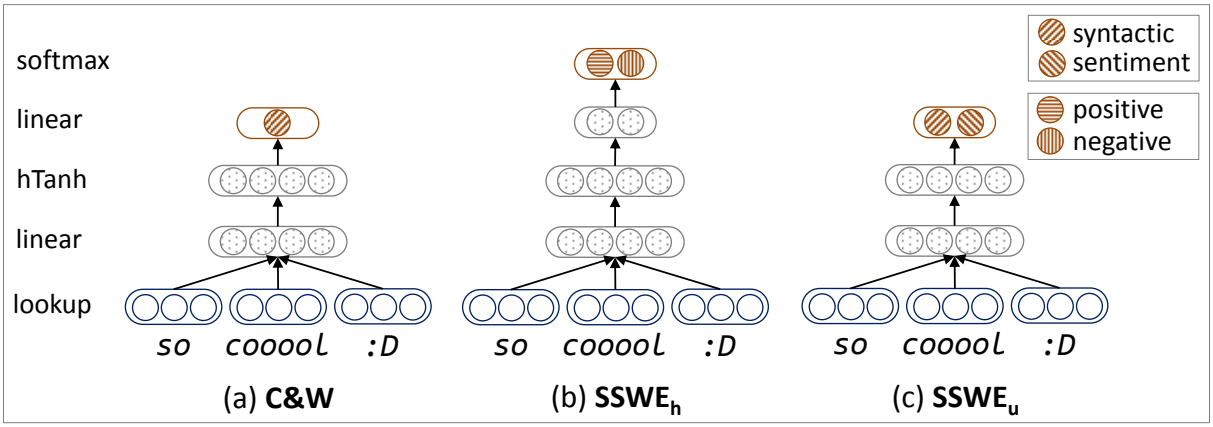

Figure 1: The traditional C&W model and our neural networks (SSWE$_h$ and SSWE$_u$) for learning sentiment-specific word embedding.

$$a = hTanh(w_1 L_t + b_1) \quad (3)$$

$$hTanh(x) = \begin{cases} -1 & \text{if } x < -1 \\ x & \text{if } -1 \leq x \leq 1 \\ 1 & \text{if } x > 1 \end{cases} \quad (4)$$

3.2 Sentiment-Specific Word Embedding

Following the traditional C&W model (Collobert et al., 2011), we incorporate the sentiment information into the neural network to learn sentiment-specific word embedding. We develop three neural networks with different strategies to integrate the sentiment information of tweets.

Basic Model 1 (SSWE$_h$). As an unsupervised approach, C&W model does not explicitly capture the sentiment information of texts. An intuitive solution to integrate the sentiment information is predicting the sentiment distribution of text based on input ngram. We do not utilize the entire sentence as input because the length of different sentences might be variant. We therefore slide the window of ngram across a sentence, and then predict the sentiment polarity based on each ngram with a shared neural network. In the neural network, the distributed representation of higher layer are interpreted as features describing the input. Thus, we utilize the continuous vector of top layer to predict the sentiment distribution of text.

Assuming there are K labels, we modify the dimension of top layer in C&W model as K and add a $softmax$ layer upon the top layer. The neural network (**SSWE$_h$**) is given in Figure 1(b). $Softmax$ layer is suitable for this scenario because its outputs are interpreted as conditional probabilities. Unlike C&W, SSWE$_h$ does not generate any corrupted ngram. Let $\boldsymbol{f}^g(t)$, where K denotes the number of sentiment polarity labels, be the gold K-dimensional multinomial distribution of input t and $\sum_k \boldsymbol{f}^g_k(t) = 1$. For positive/negative classification, the distribution is of the form [1,0] for positive and [0,1] for negative. The cross-entropy error of the $softmax$ layer is :

$$loss_h(t) = -\sum_{k=\{0,1\}} \boldsymbol{f}^g_k(t) \cdot log(\boldsymbol{f}^h_k(t)) \quad (5)$$

where $\boldsymbol{f}^g(t)$ is the gold sentiment distribution and $\boldsymbol{f}^h(t)$ is the predicted sentiment distribution.

Basic Model 2 (SSWE$_r$). SSWE$_h$ is trained by predicting the positive ngram as [1,0] and the negative ngram as [0,1]. However, the constraint of SSWE$_h$ is too strict. The distribution of [0.7,0.3] can also be interpreted as a positive label because the positive score is larger than the negative score. Similarly, the distribution of [0.2,0.8] indicates negative polarity. Based on the above observation, the hard constraints in SSWE$_h$ should be relaxed. If the sentiment polarity of a tweet is positive, the predicted positive score is expected to be larger than the predicted negative score, and the exact reverse if the tweet has negative polarity.

We model the relaxed constraint with a ranking objective function and borrow the bottom four layers from SSWE$_h$, namely $lookup \rightarrow linear \rightarrow hTanh \rightarrow linear$ in Figure 1(b), to build the relaxed neural network (**SSWE$_r$**). Compared with SSWE$_h$, the $softmax$ layer is removed because SSWE$_r$ does not require probabilistic interpretation. The hinge loss of SSWE$_r$ is modeled as de-

scribed below.

$$loss_r(t) = max(0, 1 - \delta_s(t)\boldsymbol{f}_0^r(t) + \delta_s(t)\boldsymbol{f}_1^r(t)\ ) \quad (6)$$

where $\boldsymbol{f}_0^r$ is the predicted positive score, $\boldsymbol{f}_1^r$ is the predicted negative score, $\delta_s(t)$ is an indicator function reflecting the sentiment polarity of a sentence,

$$\delta_s(t) = \begin{cases} 1 & \text{if } \boldsymbol{f}^g(t) = [1, 0] \\ -1 & \text{if } \boldsymbol{f}^g(t) = [0, 1] \end{cases} \quad (7)$$

Similar with SSWE$_h$, SSWE$_r$ also does not generate the corrupted ngram.

Unified Model (SSWE$_u$). The C&W model learns word embedding by modeling syntactic contexts of words but ignoring sentiment information. By contrast, SSWE$_h$ and SSWE$_r$ learn sentiment-specific word embedding by integrating the sentiment polarity of sentences but leaving out the syntactic contexts of words. We develop a unified model (**SSWE$_u$**) in this part, which captures the sentiment information of sentences as well as the syntactic contexts of words. SSWE$_u$ is illustrated in Figure 1(c).

Given an original (or corrupted) ngram and the sentiment polarity of a sentence as the input, SSWE$_u$ predicts a two-dimensional vector for each input ngram. The two scalars ($\boldsymbol{f}_0^u$, $\boldsymbol{f}_1^u$) stand for language model score and sentiment score of the input ngram, respectively. The training objectives of SSWE$_u$ are that (1) the original ngram should obtain a higher language model score $\boldsymbol{f}_0^u(t)$ than the corrupted ngram $\boldsymbol{f}_0^u(t^r)$, and (2) the sentiment score of original ngram $\boldsymbol{f}_1^u(t)$ should be more consistent with the gold polarity annotation of sentence than corrupted ngram $\boldsymbol{f}_1^u(t^r)$. The loss function of SSWE$_u$ is the linear combination of two hinge losses,

$$loss_u(t, t^r) = \alpha \cdot loss_{cw}(t, t^r) + (1 - \alpha) \cdot loss_{us}(t, t^r) \quad (8)$$

where $loss_{cw}(t, t^r)$ is the syntactic loss as given in Equation 1, $loss_{us}(t, t^r)$ is the sentiment loss as described in Equation 9. The hyper-parameter α weighs the two parts.

$$loss_{us}(t, t^r) = max(0, 1 - \delta_s(t)\boldsymbol{f}_1^u(t) + \delta_s(t)\boldsymbol{f}_1^u(t^r)\ ) \quad (9)$$

Model Training. We train sentiment-specific word embedding from massive distant-supervised tweets collected with positive and negative emoticons[1]. We crawl tweets from April 1st, 2013 to April 30th, 2013 with TwitterAPI. We tokenize each tweet with TwitterNLP (Gimpel et al., 2011), remove the *@user* and *URLs* of each tweet, and filter the tweets that are too short (< 7 words). Finally, we collect 10M tweets, selected by 5M tweets with positive emoticons and 5M tweets with negative emoticons.

We train SSWE$_h$, SSWE$_r$ and SSWE$_u$ by taking the derivative of the loss through back-propagation with respect to the whole set of parameters (Collobert et al., 2011), and use AdaGrad (Duchi et al., 2011) to update the parameters. We empirically set the window size as 3, the embedding length as 50, the length of hidden layer as 20 and the learning rate of AdaGrad as 0.1 for all baseline and our models. We learn embedding for unigrams, bigrams and trigrams separately with same neural network and same parameter setting. The contexts of unigram (bigram/trigram) are the surrounding unigrams (bigrams/trigrams), respectively.

3.3 Twitter Sentiment Classification

We apply sentiment-specific word embedding for Twitter sentiment classification under a supervised learning framework as in previous work (Pang et al., 2002). Instead of hand-crafting features, we incorporate the continuous representation of words and phrases as the feature of a tweet. The sentiment classifier is built from tweets with manually annotated sentiment polarity.

We explore min, $average$ and max convolutional layers (Collobert et al., 2011; Socher et al., 2011a), which have been used as simple and effective methods for compositionality learning in vector-based semantics (Mitchell and Lapata, 2010), to obtain the tweet representation. The result is the concatenation of vectors derived from different convolutional layers.

$$z(tw) = [z_{max}(tw), z_{min}(tw), z_{average}(tw)]$$

where $z(tw)$ is the representation of tweet tw and $z_x(tw)$ is the results of the convolutional layer $x \in \{min, max, average\}$. Each convolutional layer

[1] We use the emoticons selected by Hu et al. (2013). The positive emoticons are :) :) :-) :D =), and the negative emoticons are :(: (:-(.

z_x employs the embedding of unigrams, bigrams and trigrams separately and conducts the matrix-vector operation of x on the sequence represented by columns in each lookup table. The output of z_x is the concatenation of results obtained from different lookup tables.

$$z_x(tw) = [w_x\langle L_{uni}\rangle^{tw}, w_x\langle L_{bi}\rangle^{tw}, w_x\langle L_{tri}\rangle^{tw}]$$

where w_x is the convolutional function of z_x, $\langle L\rangle^{tw}$ is the concatenated column vectors of the words in the tweet. L_{uni}, L_{bi} and L_{tri} are the lookup tables of the unigram, bigram and trigram embedding, respectively.

4 Experiment

We conduct experiments to evaluate SSWE by incorporating it into a supervised learning framework for Twitter sentiment classification. We also directly evaluate the effectiveness of the SSWE by measuring the word similarity in the embedding space for sentiment lexicons.

4.1 Twitter Sentiment Classification

Experiment Setup and Datasets. We conduct experiments on the latest Twitter sentiment classification benchmark dataset in SemEval 2013 (Nakov et al., 2013). The training and development sets were completely in full to task participants. However, we were unable to download all the training and development sets because some tweets were deleted or not available due to modified authorization status. The test set is directly provided to the participants. The distribution of our dataset is given in Table 1. We train sentiment classifier with LibLinear (Fan et al., 2008) on the training set, tune parameter $-c$ on the dev set and evaluate on the test set. Evaluation metric is the Macro-F1 of positive and negative categories [2].

	Positive	Negative	Neutral	Total
Train	2,642	994	3,436	7,072
Dev	408	219	493	1,120
Test	1,570	601	1,639	3,810

Table 1: Statistics of the SemEval 2013 Twitter sentiment classification dataset.

Baseline Methods. We compare our method with the following sentiment classification algorithms:

(1) *DistSuper*: We use the 10 million tweets selected by positive and negative emoticons as training data, and build sentiment classifier with LibLinear and ngram features (Go et al., 2009).

(2) *SVM*: The ngram features and Support Vector Machine are widely used baseline methods to build sentiment classifiers (Pang et al., 2002). LibLinear is used to train the SVM classifier.

(3) *NBSVM*: NBSVM (Wang and Manning, 2012) is a state-of-the-art performer on many sentiment classification datasets, which trades-off between Naive Bayes and NB-enhanced SVM.

(4) *RAE*: Recursive Autoencoder (Socher et al., 2011c) has been proven effective in many sentiment analysis tasks by learning compositionality automatically. We run RAE with randomly initialized word embedding.

(5) *NRC*: NRC builds the top-performed system in SemEval 2013 Twitter sentiment classification track which incorporates diverse sentiment lexicons and many manually designed features. We re-implement this system because the codes are not publicly available [3]. *NRC-ngram* refers to the feature set of *NRC* leaving out ngram features.

Except for $DistSuper$, other baseline methods are conducted in a supervised manner. We do not compare with RNTN (Socher et al., 2013b) because we cannot efficiently train the RNTN model. The reason lies in that the tweets in our dataset do not have accurately parsed results or fine grained sentiment labels for phrases. Another reason is that the RNTN model trained on movie reviews cannot be directly applied on tweets due to the differences between domains (Blitzer et al., 2007).

Results and Analysis. Table 2 shows the macro-F1 of the baseline systems as well as the SSWE-based methods on positive/negative sentiment classification of tweets. Distant supervision is relatively weak because the noisy-labeled tweets are treated as the gold standard, which affects the performance of classifier. The results of bag-of-ngram (uni/bi/tri-gram) features are not satisfied because the one-hot word representation cannot capture the latent connections between words. NBSVM and RAE perform comparably and have

[2]We investigate 2-class Twitter sentiment classification (positive/negative) instead of 3-class Twitter sentiment classification (positive/negative/neutral) in SemEval2013.

[3]For 3-class sentiment classification in SemEval 2013, our re-implementation of NRC achieved 68.3%, 0.7% lower than NRC (69%) due to less training data.

Method	Macro-F1
DistSuper + unigram	61.74
DistSuper + uni/bi/tri-gram	63.84
SVM + unigram	74.50
SVM + uni/bi/tri-gram	75.06
NBSVM	75.28
RAE	75.12
NRC (**Top System** in SemEval)	**84.73**
NRC - ngram	84.17
$SSWE_u$	**84.98**
$SSWE_u$+NRC	**86.58**
$SSWE_u$+NRC-ngram	**86.48**

Table 2: Macro-F1 on positive/negative classification of tweets.

a big gap in comparison with the NRC and SSWE-based methods. The reason is that RAE and NBSVM learn the representation of tweets from the small-scale manually annotated training set, which cannot well capture the comprehensive linguistic phenomenons of words.

NRC implements a variety of features and reaches 84.73% in macro-F1, verifying the importance of a better feature representation for Twitter sentiment classification. We achieve 84.98% by using only $SSWE_u$ as features without borrowing any sentiment lexicons or hand-crafted rules. The results indicate that $SSWE_u$ automatically learns discriminative features from massive tweets and performs comparable with the state-of-the-art manually designed features. After concatenating $SSWE_u$ with the feature set of *NRC*, the performance is further improved to 86.58%. We also compare $SSWE_u$ with the ngram feature by integrating SSWE into *NRC-ngram*. The concatenated features *$SSWE_u$+NRC-ngram* (86.48%) outperform the original feature set of NRC (84.73%).

As a reference, we apply $SSWE_u$ on subjective classification of tweets, and obtain 72.17% in macro-F1 by using only $SSWE_u$ as feature. After combining $SSWE_u$ with the feature set of *NRC*, we improve NRC from 74.86% to 75.39% for subjective classification.

Comparision between Different Word Embedding. We compare sentiment-specific word embedding ($SSWE_h$, $SSWE_r$, $SSWE_u$) with baseline embedding learning algorithms by only using word embedding as features for Twitter sentiment classification. We use the embedding of unigrams, bigrams and trigrams in the experiment. The embeddings of C&W (Collobert et al., 2011), word2vec[4], WVSA (Maas et al., 2011) and our models are trained with the same dataset and same parameter setting. We compare with C&W and word2vec as they have been proved effective in many NLP tasks. The trade-off parameter of ReEmb (Labutov and Lipson, 2013) is tuned on the development set of SemEval 2013.

Table 3 shows the performance on the positive/negative classification of tweets[5]. ReEmb(C&W) and ReEmb(w2v) stand for the use of embeddings learned from 10 million distant-supervised tweets with C&W and word2vec, respectively. Each row of Table 3 represents a word embedding learning algorithm. Each column stands for a type of embedding used to compose features of tweets. The column *uni+bi* denotes the use of unigram and bigram embedding, and the column *uni+bi+tri* indicates the use of unigram, bigram and trigram embedding.

Embedding	unigram	uni+bi	uni+bi+tri
C&W	74.89	75.24	75.89
Word2vec	73.21	75.07	76.31
ReEmb(C&W)	75.87	–	–
ReEmb(w2v)	75.21	–	–
WVSA	77.04	–	–
$SSWE_h$	81.33	83.16	83.37
$SSWE_r$	80.45	81.52	82.60
$SSWE_u$	**83.70**	**84.70**	**84.98**

Table 3: Macro-F1 on positive/negative classification of tweets with different word embeddings.

From the first column of Table 3, we can see that the performance of C&W and word2vec are obviously lower than sentiment-specific word embeddings by only using unigram embedding as features. The reason is that C&W and word2vec do not explicitly exploit the sentiment information of the text, resulting in that the words with opposite polarity such as *good* and *bad* are mapped to close word vectors. When such word embeddings are fed as features to a Twitter sentiment classifier, the discriminative ability of sentiment words are weakened thus the classification performance is affected. Sentiment-specific word em-

[4]Available at https://code.google.com/p/word2vec/. We utilize the Skip-gram model because it performs better than CBOW in our experiments.

[5]MVSA and ReEmb are not suitable for learning bigram and trigram embedding because their sentiment predictor functions only utilize the unigram embedding.

beddings (SSWE$_h$, SSWE$_r$, SSWE$_u$) effectively distinguish words with opposite sentiment polarity and perform best in three settings. SSWE outperforms MVSA by exploiting more contextual information in the sentiment predictor function. SSWE outperforms ReEmb by leveraging more sentiment information from massive distant-supervised tweets. Among three sentiment-specific word embeddings, SSWE$_u$ captures more context information and yields best performance. SSWE$_h$ and SSWE$_r$ obtain comparative results.

From each row of Table 3, we can see that the bigram and trigram embeddings consistently improve the performance of Twitter sentiment classification. The underlying reason is that a phrase, which cannot be accurately represented by unigram embedding, is directly encoded into the n-gram embedding as an idiomatic unit. A typical case in sentiment analysis is that the composed phrase and multiword expression may have a different sentiment polarity than the individual words it contains, such as *not [bad]* and *[great] deal of* (the word in the bracket has different sentiment polarity with the ngram). A very recent study by Mikolov et al. (2013) also verified the effectiveness of phrase embedding for analogically reasoning phrases.

Effect of α in SSWE$_u$ We tune the hyper-parameter α of SSWE$_u$ on the development set by using unigram embedding as features. As given in Equation 8, α is the weighting score of syntactic loss of SSWE$_u$ and trades-off the syntactic and sentiment losses. SSWE$_u$ is trained from 10 million distant-supervised tweets.

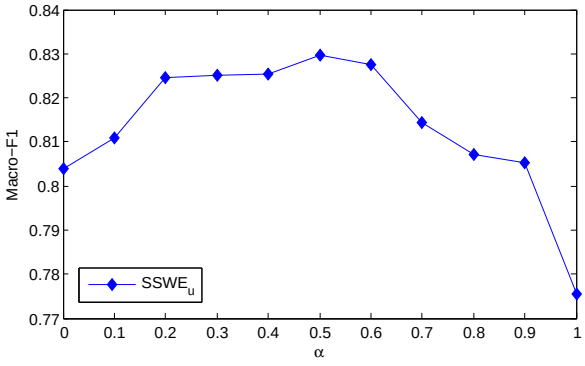

Figure 2: Macro-F1 of SSWE$_u$ on the development set of SemEval 2013 with different α.

Figure 2 shows the macro-F1 of SSWE$_u$ on positive/negative classification of tweets with different α on our development set. We can see that SSWE$_u$ performs better when α is in the range of [0.5, 0.6], which balances the syntactic context and sentiment information. The model with α=1 stands for C&W model, which only encodes the syntactic contexts of words. The sharp decline at α=1 reflects the importance of sentiment information in learning word embedding for Twitter sentiment classification.

Effect of Distant-supervised Data in SSWE$_u$ We investigate how the size of the distant-supervised data affects the performance of SSWE$_u$ feature for Twitter sentiment classification. We vary the number of distant-supervised tweets from 1 million to 12 million, increased by 1 million. We set the α of SSWE$_u$ as 0.5, according to the experiments shown in Figure 2. Results of positive/negative classification of tweets on our development set are given in Figure 3.

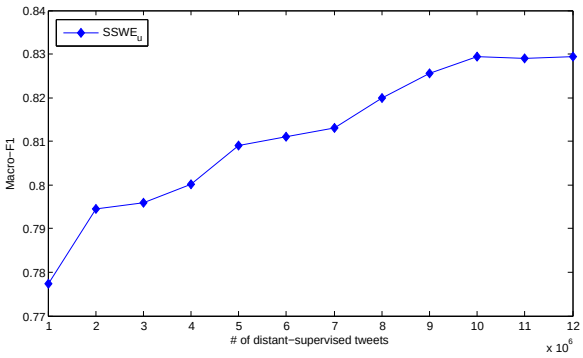

Figure 3: Macro-F1 of SSWE$_u$ with different size of distant-supervised data on our development set.

We can see that when more distant-supervised tweets are added, the accuracy of SSWE$_u$ consistently improves. The underlying reason is that when more tweets are incorporated, the word embedding is better estimated as the vocabulary size is larger and the context and sentiment information are richer. When we have 10 million distant-supervised tweets, the SSWE$_u$ feature increases the macro-F1 of positive/negative classification of tweets to 82.94% on our development set. When we have more than 10 million tweets, the performance remains stable as the contexts of words have been mostly covered.

4.2 Word Similarity of Sentiment Lexicons

The quality of SSWE has been implicitly evaluated when applied in Twitter sentiment classification in the previous subsection. We explicitly evaluate it in this section through word similarity in the em-

bedding space for sentiment lexicons. The evaluation metric is the accuracy of polarity consistency between each sentiment word and its top N closest words in the sentiment lexicon,

$$Accuracy = \frac{\sum_{i=1}^{\#Lex} \sum_{j=1}^{N} \beta(w_i, c_{ij})}{\#Lex \times N} \quad (10)$$

where $\#Lex$ is the number of words in the sentiment lexicon, w_i is the *i-th* word in the lexicon, c_{ij} is the *j-th* closest word to w_i in the lexicon with cosine similarity, $\beta(w_i, c_{ij})$ is an indicator function that is equal to 1 if w_i and c_{ij} have the same sentiment polarity and 0 for the opposite case. The higher accuracy refers to a better polarity consistency of words in the sentiment lexicon. We set N as 100 in our experiment.

Experiment Setup and Datasets We utilize the widely-used sentiment lexicons, namely *MPQA* (Wilson et al., 2005) and *HL* (Hu and Liu, 2004), to evaluate the quality of word embedding. For each lexicon, we remove the words that do not appear in the lookup table of word embedding. We only use unigram embedding in this section because these sentiment lexicons do not contain phrases. The distribution of the lexicons used in this paper is listed in Table 4.

Lexicon	Positive	Negative	Total
HL	1,331	2,647	3,978
MPQA	1,932	2,817	4,749
Joint	1,051	2,024	3,075

Table 4: Statistics of the sentiment lexicons. *Joint* stands for the words that occur in both *HL* and *MPQA* with the same sentiment polarity.

Results. Table 5 shows our results compared to other word embedding learning algorithms. The accuracy of *random* result is 50% as positive and negative words are randomly occurred in the nearest neighbors of each word. Sentiment-specific word embeddings (SSWE$_h$, SSWE$_r$, SSWE$_u$) outperform existing neural models (C&W, word2vec) by large margins. SSWE$_u$ performs best in three lexicons. SSWE$_h$ and SSWE$_r$ have comparable performances. Experimental results further demonstrate that sentiment-specific word embeddings are able to capture the sentiment information of texts and distinguish words with opposite sentiment polarity, which are not well solved in traditional neural models like C&W and word2vec. SSWE outperforms MVSA and ReEmb by exploiting more context information of words and sentiment information of sentences, respectively.

Embedding	HL	MPQA	Joint
Random	50.00	50.00	50.00
C&W	63.10	58.13	62.58
Word2vec	66.22	60.72	65.59
ReEmb(C&W)	64.81	59.76	64.09
ReEmb(w2v)	67.16	61.81	66.39
WVSA	68.14	64.07	67.12
SSWE$_h$	74.17	68.36	74.03
SSWE$_r$	73.65	68.02	73.14
SSWE$_u$	**77.30**	**71.74**	**77.33**

Table 5: Accuracy of the polarity consistency of words in different sentiment lexicons.

5 Conclusion

In this paper, we propose learning continuous word representations as features for Twitter sentiment classification under a supervised learning framework. We show that the word embedding learned by traditional neural networks are not effective enough for Twitter sentiment classification. These methods typically only model the context information of words so that they cannot distinguish words with similar context but opposite sentiment polarity (e.g. *good* and *bad*). We learn sentiment-specific word embedding (SSWE) by integrating the sentiment information into the loss functions of three neural networks. We train SSWE with massive distant-supervised tweets selected by positive and negative emoticons. The effectiveness of SSWE has been implicitly evaluated by using it as features in sentiment classification on the benchmark dataset in SemEval 2013, and explicitly verified by measuring word similarity in the embedding space for sentiment lexicons. Our unified model combining syntactic context of words and sentiment information of sentences yields the best performance in both experiments.

Acknowledgments

We thank Yajuan Duan, Shujie Liu, Zhenghua Li, Li Dong, Hong Sun and Lanjun Zhou for their great help. This research was partly supported by National Natural Science Foundation of China (No.61133012, No.61273321, No.61300113).

References

Luciano Barbosa and Junlan Feng. 2010. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of International Conference on Computational Linguistics*, pages 36–44.

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

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Analysis and Machine Intelligence*.

Yoshua Bengio. 2013. Deep learning of representations: Looking forward. *arXiv preprint arXiv:1305.0445*.

Dmitriy Bespalov, Bing Bai, Yanjun Qi, and Ali Shokoufandeh. 2011. Sentiment classification based on supervised latent n-gram analysis. In *Proceedings of the Conference on Information and Knowledge Management*, pages 375–382.

Dmitriy Bespalov, Yanjun Qi, Bing Bai, and Ali Shokoufandeh. 2012. Sentiment classification with supervised sequence embedding. In *Machine Learning and Knowledge Discovery in Databases*, pages 159–174. Springer.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Annual Meeting of the Association for Computational Linguistics*, volume 7.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of International Conference on Computational Linguistics*, pages 241–249.

Xiaowen Ding, Bing Liu, and Philip S Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of the International Conference on Web Search and Data Mining*, pages 231–240.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, pages 2121–2159.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.

Ronen Feldman. 2013. Techniques and applications for sentiment analysis. *Communications of the ACM*, 56(4):82–89.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 42–47.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. *Proceedings of International Conference on Machine Learning*.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12.

Karl Moritz Hermann and Phil Blunsom. 2013. The role of syntax in vector space models of compositional semantics. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 894–904.

Ming Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 168–177.

Xia Hu, Jiliang Tang, Huiji Gao, and Huan Liu. 2013. Unsupervised sentiment analysis with emotional signals. In *Proceedings of the International World Wide Web Conference*, pages 607–618.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. *The Proceeding of Annual Meeting of the Association for Computational Linguistics*, 1:151–160.

Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. 2011. Twitter sentiment analysis: The good the bad and the omg! In *The International AAAI Conference on Weblogs and Social Media*.

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

Kun-Lin Liu, Wu-Jun Li, and Minyi Guo. 2012. Emoticon smoothed language models for twitter sentiment analysis. In *The Association for the Advancement of Artificial Intelligence*.

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 *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *The Conference on Neural Information Processing Systems*.

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

Andriy Mnih and Geoffrey E Hinton. 2009. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088.

Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *Proceedings of the International Workshop on Semantic Evaluation*.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. In *Proceedings of the International Workshop on Semantic Evaluation*, volume 13.

Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of Language Resources and Evaluation Conference*, volume 2010.

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 *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–86.

Richard Socher, Eric H Huang, Jeffrey Pennington, Andrew Y Ng, and Christopher D Manning. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *The Conference on Neural Information Processing Systems*, 24:801–809.

Richard Socher, Cliff C Lin, Andrew Ng, and Chris Manning. 2011b. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the International Conference on Machine Learning*, pages 129–136.

Richard Socher, J. Pennington, E.H. Huang, A.Y. Ng, and C.D. Manning. 2011c. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Conference on Empirical Methods in Natural Language Processing*, pages 151–161.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013a. Parsing with compositional vector grammars. In *Annual Meeting of the Association for Computational Linguistics*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307.

Mike Thelwall, Kevan Buckley, and Georgios Paltoglou. 2012. Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology*, 63(1):163–173.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. *Annual Meeting of the Association for Computational Linguistics*.

Peter D Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pages 417–424.

Sida Wang and Christopher D Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 90–94.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 347–354.

Ainur Yessenalina and Claire Cardie. 2011. Compositional matrix-space models for sentiment analysis. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 172–182.

Jichang Zhao, Li Dong, Junjie Wu, and Ke Xu. 2012. Moodlens: an emoticon-based sentiment analysis system for chinese tweets. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for chinese word segmentation and pos tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 647–657.