The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

SNNN: Promoting Word Sentiment and Negation in Neural Sentiment Classification

Qinmin Hu, Jie Zhou, Qin Chen, Liang He
Shanghai Key Laboratory of Multidimensional Information Processing
School of Computer Science and Software Engineering
East China Normal University, Shanghai, 200062, China
{qmhu,lh}@cs.ecnu.edu.cn, {jzhou, qchen}@ica.stc.sh.cn

Abstract

We mainly investigate word influence in neural sentiment classification, which results in a novel approach to promoting word sentiment and negation as attentions. Particularly, a sentiment and negation neural network (SNNN) is proposed, including a sentiment neural network (SNN) and a negation neural network (NNN). First, we modify the word level by embedding the word sentiment and negation information as the extra layers for the input. Second, we adopt a hierarchical LSTM model to generate the word-level, sentence-level and document-level representations respectively. After that, we enhance word sentiment and negation as attentions over the semantic level. Finally, the experiments conducting on the IMDB and Yelp data sets show that our approach is superior to the state-of-the-art methods. Furthermore, we draw the interesting conclusions that (1) LSTM performs better than CNN and RNN for neural sentiment classification; (2) word sentiment and negation are a strong alliance as attentions, while overfitting occurs when they are simultaneously applied at the embedding layer; and (3) word sentiment/negation can be singly implemented for better performance as both embedding layer and attention at the same time.

Introduction and Motivation

Many approaches in sentiment classification, utilize a supervised classifier and rely on extensive feature engineering (Go, Bhayani, and Huang 2009; Barbosa and Feng 2010; Pak and Paroubek 2010; Jiang et al. 2011; Mukherjee, Bhattacharyya, and others 2012; Hamdan, Béchet, and Bellot 2013; Mohammad, Kiritchenko, and Zhu 2013; Cheng et al. 2017). However, feature engineering costs extensive labour work and needs specific domain knowledge. Therefore, feature learning is an alternative way to learn discriminative features automatically from data. The work presented by (Socher et al. 2013; Yessenalina and Cardie 2011; Hu et al. 2016) proved that the features of a sentence/document could be learnt through its word embedding. Existing approaches of learning word embedding (Collobert et al. 2011; Mikolov et al. 2013b; Yang, Hu, and He 2015) then focused on modeling the syntactic context. After that, people turn to neural network for its learning ability of text representation (Glorot, Bordes, and Bengio 2011; Zhai and Zhang 2016; Socher et al. 2011a; 2011b; 2012; 2013; Kim 2014; Tang, Qin, and Liu 2015b; Yang et al. 2016; Chen et al. 2016; Ren et al. 2016a)

Tang et al. (Tang, Qin, and Liu 2015b) proposed a neural network model to learn vector-based document representation in a CNN based sentiment classification, where the authors found that neural gates outperformed the traditional recurrent neural network. Then, Chen et al. (Chen et al. 2016) brought a hierarchical neural network to incorporate global user and product information as attentions. They mainly challenged Tang's work that the characteristics of the user and product information should be reflected on the semantic level, instead of the word level. Based on these two pieces of the state-of-the-art work, we aim to investigate further word influence in neural sentiment classification. The motivation is that it is theoretically feasible and sound by adding more word information from multiple dimensions in the word level as the input, since the quality of document/sentence representation highly depends on word representations.

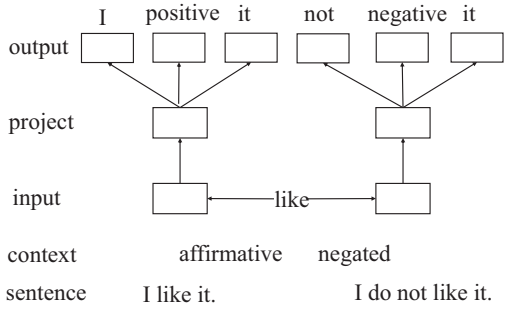


Figure 1: A perfect example of sentiment representation: (1) the correct sentiment is obtained as output on both sentences; (2) "not" at the output level is ideally represented the negated word "not" of the input level.

Figure 1 presents a perfect example to show that the correct sentiment is obtained at the output level on both sentences (Hu et al. 2016). Theoretically, we say the surrounding words and word sentiment are predicted in affirmative (negated) context, when the affirmative (negated) words are mapped to the affirmative (negated) representations.

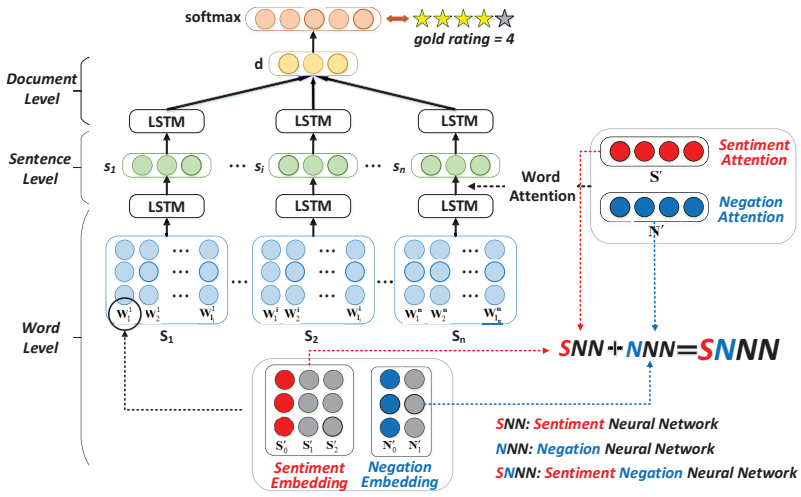


Figure 2: The architecture of the Sentiment Negation Neural Network (SNNN) model

	Never
Word2vec	ever,rarely,**already**,**gladly**,eventually,havent, **voluntarily**,hadnt,**previously**,**everytime**
LSTM+A	ever,rarely,havent,not,**already**,havne't,no,hadnt, **previously**, **everytime**
LSTM+NA	ever,rarely,not,nor,no,hardly,nothing,nobody, neither,seldom

Table 1: Top 10 neighbors of "never" represented by word2vec, LSTM+A and LSTM+NA: (1) error neighbors are emphasized as bold; (2) "LSTM+A" adopts LSTM with general attention; (3) "LSTM+NA" is our proposed approach with negation attention, which enhances negation and effects the ranking and selection of neighbors.

However, negation representation contains many noises under different projection methods. We demonstrate another example in Table 1 to show how the negation neighbors of "never" are represented by word2vec, LSTM+A and LSTM+NA. We can see that the top 10 neighbors represented by "word2vec" have five errors which are emphasized as bold, and our proposed approach with negation attention "LSTM+NA" is ideally good to represent "never". This can be explained that the negation attention based hierarchical neural network "LSTM+NA" enhances negation and effects the ranking and selection of neighbors. Hence, we desire to promote negation as an extra dimension in word representation for better sentence/document representation.

Figure 1 also shows the semantic meaning of two sentences is decided by the keyword "like" and its negation. Generally, we discuss semantics over the sentence level and treat most words as neutral. However, there are some words like "like", "good" and "bad" which bring the strong sentiment information by themselves and are definitely the central words on the semantic level. Therefore, we believe the sentiment of words themselves should be emphasized in word representation.

Motivated by the above examples, we establish a novel approach to promoting word sentiment and negation as attentions for neural sentiment classification. In Figure 2, our proposed sentiment and negation neural network (SNNN) consists of two parts: sentiment neural network (SNN) and negation neural network (NNN). First, we add extra layers in the word level, where the input not only includes all the words themselves, but also has word sentiment and negation information. Second, we adopt a hierarchical Long Short-Term Memory Network (LSTM) model to generate the word-level, sentence-level and document-level representations. After that, we introduce word sentiment and negation information as attentions over the word level to capture the semantic components. Finally, we conduct the experiments on four large-scale review datasets from IMDB and Yelp Dataset Challenges.

In summary, our contributions can be presented as follows.

(1) We propose a novel SNNN model for sentiment classification, in which word sentiment and negation enrich word representation for better document rating performance.

(2) We promote word sentiment and negation based attention, which is a higher level than the traditional attention based neural network that only considers the local text information.

(3) We conduct empirical study on four large-scale data sets to show that our approach outperforms the state-of-the-art methods.

(4) We draw some interesting conclusions: (a) LSTM is the best framework candidate for neural sentiment classification; (b) word sentiment and negation are a strong alliance as attentions, while overfitting occurs when they are simultaneously applied at the embedding layer; (c) word sentiment/negation can be singly implemented for better performance as both embedding layer and attention.

Related Work

Traditional machine learning methods and neural network are two popular ways for sentiment classification (Go, Bhayani, and Huang 2009; Barbosa and Feng 2010;

Pak and Paroubek 2010; Jiang et al. 2011; Mukherjee, Bhattacharyya, and others 2012; Chen et al. 2015; 2014; Deng, Yu, and Yang 2016; Hamdan, Béchet, and Bellot 2013; Mohammad, Kiritchenko, and Zhu 2013; Tang, Qin, and Liu 2015b; Chen et al. 2016; Ren et al. 2016b; 2016a; Chen et al. 2017b; Li et al. 2017; Hu et al. 2016). However, the machine learning methods with a supervised classifier have relied on extensive feature engineering. Tang et al. (Tang et al. 2014) was the first who did not concentrate on the labor-intensive feature engineering, but developed three neural networks to learn discriminative features automatically from a large number of labelled tweets.

Neural Sentiment Classification

After that, researchers started to study sentiment neural network for its learning ability of text representation. (Glorot, Bordes, and Bengio 2011) and (Zhai and Zhang 2016) adopted Stacked Denoising Autoencoder(SDA) in sentiment classification. Socher (Socher et al. 2011a; 2011b; 2012; 2013) proposed a series of recursive neural network models for sentence/document representations. Moreover, (Kim 2014; Tang, Qin, and Liu 2015b) achieved good performance in sentiment classification by applying convolution neural network (CNN) to learn sentence representations. (Yang et al. 2016; Tang, Qin, and Liu 2015b; Chen et al. 2016) proposed hierarchical models to obtain document level sentiment classification. Furthermore, they used attention mechanism to find meaningful words from sentences in a document.

However, many of existing neural network sentiment classification methods ignore the word level sentiment and negation, which has crucial effects on the sentiment polarities.

Negation in Sentiment Classification

Negation plays an important role in sentiment classification, since it can modify the sentiment of its scope. As early as (Pang, Lee, and Vaithyanathan 2002), when a word was detected as negated, they added the prefix NOT_ to the word as a new bag-of-words feature to determine the sentiment in negated context. The scope of negation defined in Pang's work was from the first word following the negation word until the first punctation or the end of sentence, where the negation words were collected manually. After that, this simple negation detection approach has been followed by many others(Polanyi and Zaenen 2006; Kennedy and Inkpen 2006; Kiritchenko, Zhu, and Mohammad 2014).

There are some interesting negation models. (Hu et al. 2016) proposed an advance Skip-gram model to incorporate both word sentiment and negation information into an embedding space. (Liu and Seneff 2009; Taboada et al. 2011) introduced a *shifting* hypothesis which assumed that negators changed the sentiment values by a constant amount. (Socher et al. 2013; Zhu et al. 2014) developed deep models based on recursive neural network to address negation through semantic composition. Especially, for short informal texts, such as tweets, (Kiritchenko, Zhu, and Mohammad 2014) proposed a simple corpus-based statistical approach to estimate the sentiment scores of words in affirmative and negated context.

SNNN: Word Sentiment and Negation Neural Network

The goal of our research problem is to analyze the overall sentiment polarity of a document, which results in a sentiment and negation neural network (SNNN) for sentiment classification. This novel SNNN mainly includes a sentiment neural network (SNN) and a negation neural network (NNN), where word sentiment and negation are creatively promoted as attentions in a Hierarchical Long Short-Term Memory (LSTM) network.

Mathematically, we define the problem as follows to give a formal representation.

Definition 1 *Let C be a sentiment class space and p_c be the probability of a sentiment class c. For a document D with n sentences $\{S_1, S_2, ..., S_n\}$, where the i^{th} sentence S_i consists of l_i words as $\{w_1^i, w_2^i, ..., w_{l_i}^i\}$, the research target is to compute the predicted sentiment class for D*

$$arg \max_{c \in C} \{p_c\}$$

SNN: Sentiment Neural Network

SNN is to add the word sentiment information into sentiment classification. First of all, it is a crucial step to learn the word embedding, which is a dense, low-dimensional and real-valued vector for a word. We then utilize hierarchical LSTM to learn the word orientations as positive, negative and neutral in word embedding, and capture the semantics representations of sentences and documents. The input layer of the word level is modified, where a word sentiment layer is added to represent the word in sentiment orientations in Figure 2.

In the word level, each word w_j^i of a sentence S_i is embedded into a low dimensional semantic vector $\mathbf{w}_j^i \in R^{di}$ (Bengio et al. 2003), where di is the dimension of the word vector. For each iteration, given the word embedding $\mathbf{w}_j^i$ as the input, the corresponding cell state $\mathbf{c}_j^i$ and hidden state $\mathbf{h}_j^i$ can be updated with the previous cell state $\mathbf{c}_{j-1}^i$ and hidden state $\mathbf{h}_{j-1}^i$.

There are three gates as the input gate **i**, the forget gate **f** and the output gate **o**, where they are generated by the sigmoid function σ over the ensemble of input $\mathbf{w}_j^i$ and the preceding hidden state $\mathbf{h}_{j-1}^i$ (Chen et al. 2016; 2017a). Hence, we describe the equations as:

$$\begin{aligned} Gate &= < \mathbf{i}_j^i, \mathbf{f}_j^i, \mathbf{o}_j^i > \\ Gate^T &= \sigma(\mathbf{W} \cdot [\mathbf{h}_{j-1}^i, \mathbf{w}_j^i] + \mathbf{b}) \end{aligned} \tag{1}$$

$$\hat{\mathbf{c}}_j^i = tanh(\mathbf{W} \cdot [\mathbf{h}_{j-1}^i, \mathbf{w}_j^i] + \mathbf{b}), \tag{2}$$

$$\mathbf{c}_j^i = \mathbf{f}_j^i \bigodot \mathbf{c}_{j-1}^i + \mathbf{i}_j^i \bigodot \hat{\mathbf{c}}_j^i, \tag{3}$$

$$\mathbf{h}_j^i = \mathbf{o}_j^i \bigodot tanh(\mathbf{c}_j^i), \tag{4}$$

where W is the weight matrices and b is bias vector.

In order to obtain the sentence representation s_i, the hidden states $[\mathbf{h}_1^i, \mathbf{h}_2^i, ..., \mathbf{h}_{l_i}^i]$ are usually fed to an average/min/max pooling layer. After that, we make the same sentence embeddings $[s_1, s_2, ..., s_n]$ in a similar way into LSTM. So does the document representation d.

Datasets	#docs	#s/d	#w/d	$\|V\|$	#class	Class Distribution
Yelp 2013	335,018	8.90	151.6	211,245	5	.09/.09/0.14/.33/.36
Yelp 2014	1,125,457	9.22	156.9	476,191	5	.10/.09/0.15/.30/.36
Yelp 2015	1,569,264	8.97	151.9	612,636	5	.10/.09/0.14/.30/.37
IMDB	348,415	14.02	325.6	115,831	10	.07/.04/0.5/.05/.08/.11/.15/.17/.12/.18

Table 2: Statistical information of Yelp 2013-2015 and IMDB datasets: #docs is the number of documents, #s/d and #w/d represent average numbers of sentences and words per document, $|V|$ is the vocabulary size of words, and #class is the number of classes.

#	Baselines	Descriptions
(1)	Majority	takes the majority sentiment label in the training set to all documents in the test set.
(2)	SVM+Ngrams	trains a SVM classifier by obtaining unigrams, bigrams and trigrams as features (Fan et al. 2008).
(3)	TextFeatures	trains a SVM classifier using text features (Kiritchenko, Zhu, and Mohammad 2014).
(4)	AvgerageSG	trains a SVM classifier by averaging word embeddings to get document representation (Mikolov et al. 2013a).
(5)	SSWE	trains a SVM classifier by learning SSWE to get document representation (Tang et al. 2014).
(6)	JMARS	collaboratively filters topic modeling of a review for document level sentiment classification(Diao et al. 2014).
(7)	Paragraph Vector	obtains a sentiment classifier on the document level by implementing PVDM (Le and Mikolov 2014).
(8)	CNN	adopts a convolutional neural network(CNN) model for sentiment analysis (Kim 2014).
(9)	Conv-GRNN	learns sentence representation with CNN, and encodes sentence and paragraph relations with GRNN (Tang, Qin, and Liu 2015a).
(10)	LSTM-GRNN	learns sentence representation with LSTM, and encodes sentence and paragraph relations with GRNN (Tang, Qin, and Liu 2015a).
(11)	NSC+UPA	uses user product attention (UPA) in neural sentiment classification(NSC) (Chen et al. 2016).
#	Approaches	Descriptions
(1)	LSTMSN	embeds word sentiment and negation in the word level and implements LSTM without attention.
(2)	LSTMSN+A	embeds word sentiment and negation in the word level and implements LSTM with standard attention.
(3)	LSTM+SNA	implements LSTM with word sentiment and negation attention.

Table 3: Descriptions of the state-of-the-art baselines and the proposed approaches

NNN: Negation Neural Network

As we present in our motivation, negation has its unique position in the word level. Table 1 presents an example of "never", where different top 10 neighbors are represented by word2vec, LSTM+A and LSTM+NA respectively: (1) "word2vec" has five errors as bold; (2) "LSTM+A" without emphasizing negation contains three noises; and (3) "LSTM+NA" which promotes negation as attention manipulates the ranking and selection of neighbors. This example fosters negation as an extra dimension to enrich the input and as attention to enhance the weights.

Furthermore, we believe that word negative sentiment and negation are superficially independent in LSTM. The sentiment of a word is reflected on the semantic level only when this word is in a sentence/document. However, the negation word, such as "no" and "not", stands for themselves, instead of the negative information. Their word representations in LSTM are totally different such that the sentence and document representations are very different.

Therefore, we propose an NNN which follows SNN. An LSTM model is also applied and the input layer of the word level is modified by adding the negation information as an extra layer in Figure 2. The similar equations are not repeated here.

Word Sentiment and Negation Attention

The existing work promoted attentions based on the importance of the word (Yang et al. 2016; 2017). Here we obtain attentions at a higher level based on word sentiment and word negation. Hence, in the word level, we adopt the word sentiment/negation attention mechanism to extract sentiment/negation-specific words of a sentence. Then, we aggregate the word sentiment/negation representation as "sentiment/negation attention" in Figure 2 to form the sentence representation. Formally, we use the following weighted sum of the hidden states to express the enhanced sentence representation as:

$$\mathbf{s}_i = \sum_{j=1}^{l_i} \alpha_j^i \mathbf{h}_j^i, \tag{5}$$

where α_j^i stands for the importance of the j_{th} sentiment/negation-specific word.

After that, we define the attention weight α_j^i for each hidden state combining with word sentiment/negation information as:

$$\alpha_j^i = \frac{exp(e(\mathbf{h}_j^i, \mathbf{sem}_j^i))}{\sum_{k=1}^{l_i} exp(e(\mathbf{h}_k^i, \mathbf{sem}_k^i))}, \tag{6}$$

where e is a score function which measures the importance of the sentiment/negation-specific word which composes of the sentence representation, and $\mathbf{sem}_j^i$ is the continuous and real valued vector of the w_j^i's sentiment/negation embedding.

The corresponding score function e is given as:

$$\begin{aligned} &e(\mathbf{h}_j^i, \mathbf{sem}_j^i) = \\ &\mathbf{v}^T tanh(\mathbf{W}_H \mathbf{h}_j^i + \mathbf{W}_{Sem} \mathbf{sem}_j^i + \mathbf{b}), \end{aligned} \tag{7}$$

where $\mathbf{W}_{Sem}$ is the weight matrix, v is the weight vector, $\mathbf{v}^T$ denotes its transpose and b is bias vector.

Sentiment Classification

The final step of this work is to obtain document representation hierarchically which is extracted from word and

	Yelp 2013		Yelp 2014		Yelp 2015		IMDB	
	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE
Majority	0.356	3.06	0.361	3.28	0.369	3.30	0.179	17.46
SVM + Unigrams	0.589	0.79	0.600	0.78	0.611	0.75	0.399	4.23
SVM + Bigrams	0.576	0.75	0.616	0.65	0.624	0.63	0.409	3.74
SVM + TextFeatures	0.598	0.68	0.618	0.68	0.624	0.60	0.405	3.56
SVM + AverageSG	0.543	1.11	0.557	1.08	0.568	1.04	0.319	5.57
SVM + SSWE	0.535	1.12	0.543	1.13	0.554	1.11	0.262	9.16
JMARS	N/A	0.97	N/A	1.00	N/A	-	N/A	4.97
Paragraph Vector	0.577	0.86	0.592	0.70	0.605	0.61	0.341	4.69
Convolutional NN	0.597	0.76	0.610	0.68	0.615	0.68	0.376	3.30
Conv-GRNN	0.637	0.56	0.655	0.51	0.660	0.50	0.425	2.71
LSTM-GRNN	**0.651**	0.50	0.671	0.48	0.676	0.49	0.453	3.00
NSC+UPA*	0.650	0.48	0.667	**0.43**	-	-	0.533	**1.64**
LSTMSN	0.619	0.57	0.640	0.52	0.665	0.51	0.456	2.21
LSTMSN + A	0.633	0.53	0.648	0.52	0.678	0.51	0.462	2.43
LSTM + SNA	0.649	**0.47**	**0.672**	0.44	**0.704**	**0.46**	**0.535**	1.93

Table 4: Experimental results on Yelp 2013-2015 and IMDB: (1) for accuracy, higher is better; (2) for MSE, lower is better; (3) * indicates we convert their RMSE results (Chen et al. 2016) into MSE, for the comparison purpose.

sentence representations, and then classify the document into our target class space C in Definition 1. Formally, we use a non-linear layer for this transformation:

$$\hat{\mathbf{d}} = tanh(\mathbf{W}_c\mathbf{d} + \mathbf{b}_c). \tag{8}$$

where $\mathbf{W}_c$ is the weight matrix of class c and $\mathbf{b}_c$ is the bias vector.

After that, we add a softmax layer to compute the document sentiment distribution as in Definition 1:

$$p_c = \frac{exp(\hat{d_c})}{\sum_{k=1}^{C} exp(\hat{d_k})} \tag{9}$$

Finally, the loss function for optimization is defined as cross-entropy error between the gold sentiment distribution and the proposed sentiment distribution at training:

$$L = -\sum_{d \in D}\sum_{c=1}^{C} p_c^g(d) \cdot log(p_c(d)), \tag{10}$$

where p_c^g is the probability of the gold sentiment class c in a space of $\{0, 1\}$, D stands for the training documents.

Empirical Study

In this section, we first describe the datasets and the experimental settings. Then, the empirical results are reported.

Datasets and Experimental Settings

We conduct experiments to evaluate the effectiveness of our proposed approach on four datasets: Yelp 2013-2015 and IMDB, which are the same as (Tang, Qin, and Liu 2015a). The statistics of the datasets are summarized in Table 2. For data training, development and testing purposes, we divide the data with the proportion of 8:1:1 and the NLTK [1] tool has been adopted on all datasets for tokenization and sentence splitting. Two evaluation metrics of Accuracy, which measures the overall sentiment classification performance, and MSE, which measures the divergences between predicted sentiment classes and ground truth classes, are defined as:

$$Accuracy = \frac{T}{N} \tag{11}$$

$$MSE = \frac{\sum_{i=1}^{N}(gd_i - pr_i)^2}{N}, \tag{12}$$

where T is the value of the predicted sentiment rating, N is the amount of documents, and gd_i, pr_i stand for the gold sentiment and predicted sentiment ratings.

In order to better compare with the existing Chen's and Tang's work (Chen et al. 2016; Tang, Qin, and Liu 2015a), we train our data with the same settings as Chen and Tang. The details are referred to (Chen et al. 2016; Tang, Qin, and Liu 2015a) because of the page limit.

Experimental Results

Table 3 introduces the descriptions of baselines and our proposed approaches with multiple embedding and attention configurations. The existing state-of-the-art baselines are from Tang's and Chen's work (Chen et al. 2016; Tang, Qin, and Liu 2015a), where we endorse all the baselines they have adopted, including their own methods.

The experimental results are shown in Table 4. We can see that our approach "LSTM+SNA" implementing LTSM with word sentiment and negation attention achieves the best results in most cases, especially on Yelp 2015, than all baselines, including two latest state-of-the-art baselines of "LSTM-GRNN" (Tang, Qin, and Liu 2015a) and "NSC-UPA" (Chen et al. 2016).

Influence of SNNN

"LSTM+SNA", which implements our proposed SNNN in Table 4, outperforms the baselines, including two latest state-of-the-art baselines (Chen et al. 2016; Tang, Qin, and Liu 2015a). Based on the descriptions of the baselines, we observe that LSTM is the best framework candidate for neural sentiment classification, especially with attention, where

[1] http://www.nltk.org/

	Yelp 2013		Yelp 2014		Yelp 2015		IMDB	
	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE
Basic neural network model								
LSTM	0.626	0.55	0.632	0.52	0.675	0.53	0.432	2.18
LSTM + A	0.634	0.54	0.640	0.53	0.678	0.52	0.462	1.96
NNN: word negation neural network								
LSTMN	0.641	0.50	0.655	0.48	0.687	0.50	0.445	2.09
LSTMN + A	0.648	0.49	0.665	0.46	0.690	0.49	0.471	2.04
LSTM + NA	**0.649**	**0.47**	0.665	0.46	0.695	0.48	0.473	1.95
LSTMN + NA	0.648	0.49	0.664	0.46	0.691	0.48	0.475	2.09
SNN: word sentiment neural network								
LSTMS	0.644	0.50	0.653	0.46	0.688	0.49	0.442	2.25
LSTMS + A	0.646	0.51	0.662	0.47	0.695	0.48	0.483	2.24
LSTM + SA	0.649	0.49	0.663	0.47	0.695	0.48	0.473	1.96
LSTMS + SA	0.649	0.50	0.671	**0.43**	0.695	0.48	0.473	2.01
SNNN: word sentiment and negation neural network								
LSTMSN	0.619	0.57	0.640	0.52	0.665	0.51	0.456	2.21
LSTMSN + A	0.633	0.53	0.648	0.52	0.678	0.51	0.462	2.43
LSTM + SNA	**0.649**	**0.47**	**0.672**	0.44	**0.704**	**0.46**	**0.535**	**1.93**

Table 5: Experimental results over basic LSTM, NNN, SNN and SNNN: for accuracy, higher is better; for MSE, lower is better.

"LSTM-GRNN", "NSC-UPA" and "LSTM+SNA" achieve the best results over four data sets.

The above conclusion is confirmed by the unstable performance of MSE on IMDB in Table 4. Since the basic LSTM model almost achieves the best MSE result, we jump into the theory of LSTM and find that LSTM is good at capturing the long document representation which exactly fits the IMDB data.

We also find that word sentiment and negation (SN) attention ("LSTM+SNA") obtains better performance than SN as the embedding layer ("LSTMSN+A"). Then, we say word sentiment and negation should be promoted as attention, although SN has shown the superior as an embedding layer. This conclusion is consistent to Chen's work (Chen et al. 2016) where they put the user product information as attention instead of just improving their weights when embedded.

Influence of Negation

In order to evaluate the effectiveness of negation in neural sentiment classification, we make the complimentary experiments in Table 5. We config four NNN runs as "LSTMN", "LSTMN+A", "LSTM+NA" and "LSTMN+NA". Second, we compare these four combinations with "LSTM" and "LSTM+A" which do not consider negation at all.

We draw a conclusion that negation plays an important role as an embedding layer in the word level, since "LSTMN" outperforms "LSTM", and "LSTMN+A" is more successful than "LSTM+A". What's more, negation works very well as attention, because "LSTM+NA" makes great progress over "LSTM+A".

Note that "LSTMN+NA" does not outperform "LSTM+NA" such that we say that negation should not be embedded and be an attention at the same time.

Influence of Word Sentiment

Table 5 generates four SNN runs of "LSTMS", "LSTMS+A", "LSTM+SA" and "LSTMN+SA". Their performance is compared with "LSTM" and "LSTM+A" which do not take word sentiment into account.

We can draw the same conclusion as negation that word sentiment is important, not only as the embedding layer, but also as attention at the word level. It is worth to point out that there is no big performance gap between word sentiment embedding and negation embedding individually.

Influence of Embedding

We plot the results in Table 5 into Figure 3. Figure 3a demonstrates the approaches without attention, and Figure 3b shows those with attention over four data sets. The x axis includes a basic LTSM, LSTMN, LSTMS and LSTMSN with/without attention. The y axis indicates the values of Accuracy and MSE.

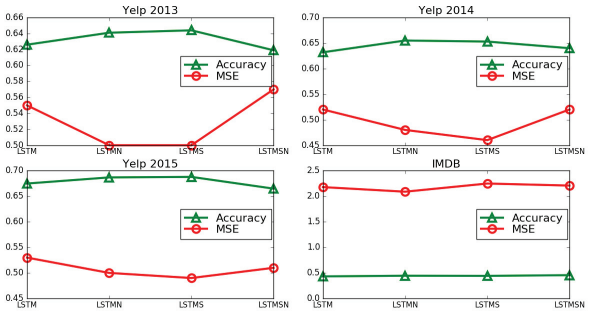
Focusing on the data in Figure 3a, we notice that "LSTMS" and "LSTMN" outperform "LSTM" and "LSTMSN" in terms of both Accuracy and MSE on three Yelp data sets. "LSTMSN" gets the worst results. The same conclusion can be drawn in Figure 3b. The exceptions happen on the IMDB data set, Accuracy has no change on both Figure 3a and Figure 3b, and MSE gets worse than LSTM/LSTM+A.

Therefore, we believe that overfitting occurs when both word sentiment and negation are embedded at the same time in LSTM/LSTM+A, but not when single word sentiment/negation is applied. The reason we analyze is that there is too much information embedded in the input.

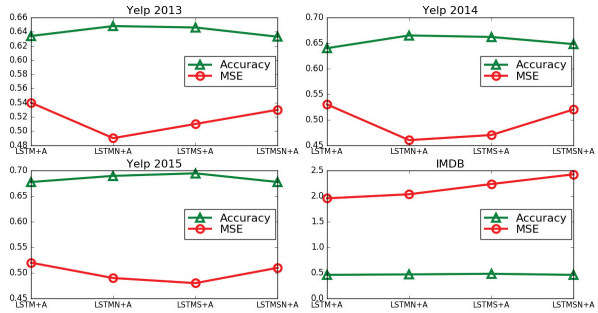
Hence, we make a conclusion that word sentiment and negation can not be embedded simultaneously, while each single of them can be implemented for better performance.

Influence of Attention

From Both Table 4 and Table 5, we find that runs with attention conquer those without attention. First of all, word sentiment/negation attention has better performance than those without the corresponding attention. Second, at the basic LSTM model, "LSTM+A" beats "LSTM" well. Fi-



(a) The influence of embedding with LSTM



(b) The influence of embedding with LSTM+A

Figure 3: The influence of embedding: word sentiment and negation can not be embedded simultaneously, while each single of them can be implemented for better performance.

nally, "LSTM+SNA" is the best one, compared to eleven state-of-the-art baselines, and the other modified proposed approaches.

In order to validate the ability to capture word sentiment and negation of "LSTM+SNA", we take a review instance in Yelp 2015 for example in Figure 4. We visualize that "LSTM+SNA" can select the words like "great", "unfortunate" and "don't", which are stronger sentiment words and negation words. This confirms our motivation of investigating word sentiment and negation at the word level, especially as attention.

Note that overfitting does not happen when both word sentiment and negation are promoted as attentions, since attention is basically to emphasize the sentiment/negation specific words that are important to the meaning of sentence, instead of embedding more information as the input. The performance curve which does not drop also supports the conclusion.

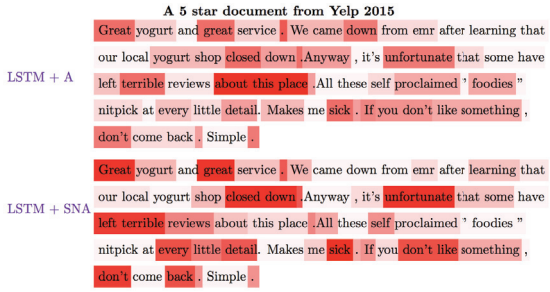


Figure 4: An example of LSTM + A and LSTM + SNA: influence of attention

Conclusions and Future Work

Our conclusion is four-fold. First, we propose a hierarchical neural network for sentiment classification, where word sentiment and negation are promoted as a higher level for the attention based model. Second, we conduct experiments on four large-scale data sets to show that our approach outperforms the state-of-the-art methods, which empirically proves word sentiment and negation enrich word representation for better document rating performance. Third, we obtain some interesting conclusions as (1) LSTM performs better than CNN and RNN for neural sentiment classification; (2) word sentiment and negation should be treated independently, although they are a strong alliance as attentions, but can not be simultaneously applied at the embedding layer; (3) word sentiment/negation can be singly implemented as both embedding layer and attention at the same time.

In the future, we will continue to focus on word influence from multiple dimensions. We will further characterize the higher level attentions on sentence representation.

Acknowledgements

This research is funded by the National Natural Science Foundation of China (No. 61572193 and 61602179) and the Science and Technology Commission of Shanghai Municipality (No. 16511102702). We also thank the anonymous reviewers for their valuable comments.

References

Barbosa, L., and Feng, J. 2010. Robust sentiment detection on twitter from biased and noisy data. In *COLING Posters*, 36–44. Association for Computational Linguistics.

Bengio, Y.; Ducharme, R.; Vincent, P.; and Jauvin, C. 2003. A neural probabilistic language model. *Journal of machine learning research* 3(Feb):1137–1155.

Chen, Q.; Yang, Y.; Hu, Q.; and He, L. 2014. Locating query-oriented experts in microblog search. In *Proceedings of Workshop on Semantic Matching in Information Retrieval co-located with the 37th international ACM SIGIR conference on research and development in information retrieval, SMIR@SIGIR 2014, Queensland, Australia, July 11, 2014.*, 16–23.

Chen, Q.; Wang, B.; Huang, B.; Hu, Q.; and He, L. 2015. ECNU at TREC 2015: Microblog track. In *Proceedings of The Twenty-Fourth Text REtrieval Conference, TREC 2015, Gaithersburg, Maryland, USA, November 17-20, 2015*.

Chen, H.; Sun, M.; Tu, C.; Lin, Y.; and Liu, Z. 2016. Neural sentiment classification with user and product attention. In *EMNLP*.

Chen, Q.; Hu, Q.; Huang, J. X.; He, L.; and An, W. 2017a. Enhancing recurrent neural networks with positional attention for question answering. In *SIGIR*, 993–996.

Chen, T.; Xu, R.; He, Y.; and Wang, X. 2017b. Improving sentiment analysis via sentence type classification using bilstm-crf and cnn. *Expert Systems with Applications* 72:221–230.

Cheng, K.; Li, J.; Tang, J.; and Liu, H. 2017. Unsupervised sentiment analysis with signed social networks. In *AAAI*, 3429–3435.

Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; and Kuksa, P. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12(Aug):2493–2537.

Deng, Z.; Yu, H.; and Yang, Y. 2016. Identifying sentiment words using an optimization model with L1 regularization. In *AAAI*, 115–121.

Diao, Q.; Qiu, M.; Wu, C.-Y.; Smola, A. J.; Jiang, J.; and Wang, C. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In *SIGKDD*, 193–202. ACM.

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, C.-J. 2008. Liblinear: A library for large linear classification. *Journal of machine learning research* 9(Aug):1871–1874.

Glorot, X.; Bordes, A.; and Bengio, Y. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, 513–520.

Go, A.; Bhayani, R.; and Huang, L. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford* 1(12).

Hamdan, H.; Béchet, F.; and Bellot, P. 2013. Experiments with dbpedia, wordnet and sentiwordnet as resources for sentiment analysis in micro-blogging. In *SemEval@ NAACL-HLT*, 455–459.

Hu, Q.; Pei, Y.; Chen, Q.; and He, L. 2016. SG++: word representation with sentiment and negation for twitter sentiment classification. In *SIGIR*, 997–1000.

Jiang, L.; Yu, M.; Zhou, M.; Liu, X.; and Zhao, T. 2011. Target-dependent twitter sentiment classification. In *ACL-HLT*, 151–160. Association for Computational Linguistics.

Kennedy, A., and Inkpen, D. 2006. Sentiment classification of movie reviews using contextual valence shifters. *Computational intelligence* 22(2):110–125.

Kim, Y. 2014. Convolutional neural networks for sentence classification. In *EMNLP*.

Kiritchenko, S.; Zhu, X.; and Mohammad, S. M. 2014. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research* 50:723–762.

Le, Q., and Mikolov, T. 2014. Distributed representations of sentences and documents. In *ICML*, 1188–1196.

Li, N.; Zhai, S.; Zhang, Z.; and Liu, B. 2017. Structural correspondence learning for cross-lingual sentiment classification with one-to-many mappings. In *AAAI*, 3490–3496.

Liu, J., and Seneff, S. 2009. Review sentiment scoring via a parse-and-paraphrase paradigm. In *EMNLP*, 161–169. Association for Computational Linguistics.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013b. Distributed representations of words and phrases and their compositionality. In *NIPS*, 3111–3119.

Mohammad, S. M.; Kiritchenko, S.; and Zhu, X. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *arXiv preprint arXiv:1308.6242*.

Mukherjee, S.; Bhattacharyya, P.; et al. 2012. Sentiment analysis in twitter with lightweight discourse analysis. In *COLING*, 1847–1864.

Pak, A., and Paroubek, P. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10.

Pang, B.; Lee, L.; and Vaithyanathan, S. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *EMNLP*, 79–86. Association for Computational Linguistics.

Polanyi, L., and Zaenen, A. 2006. Contextual valence shifters. *Computing Attitude and Affect in Text* 20:1–10.

Ren, Y.; Zhang, Y.; Zhang, M.; and Ji, D. 2016a. Context-sensitive twitter sentiment classification using neural network. In *AAAI*, 215–221.

Ren, Y.; Zhang, Y.; Zhang, M.; and Ji, D. 2016b. Improving twitter sentiment classification using topic-enriched multi-prototype word embeddings. In *AAAI*, 3038–3044.

Socher, R.; Lin, C. C.; Manning, C.; and Ng, A. Y. 2011a. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, 129–136.

Socher, R.; Pennington, J.; Huang, E. H.; Ng, A. Y.; and Manning, C. D. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *EMNLP*, 151–161. Association for Computational Linguistics.

Socher, R.; Huval, B.; Manning, C. D.; and Ng, A. Y. 2012. Semantic compositionality through recursive matrix-vector spaces. In *EMNLP*, 1201–1211. Association for Computational Linguistics.

Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D.; Ng, A. Y.; Potts, C.; et al. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, volume 1631, 1642. Citeseer.

Taboada, M.; Brooke, J.; Tofiloski, M.; Voll, K.; and Stede, M. 2011. Lexicon-based methods for sentiment analysis. *Computational linguistics* 37(2):267–307.

Tang, D.; Wei, F.; Yang, N.; Zhou, M.; Liu, T.; and Qin, B. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL*, 1555–1565.

Tang, D.; Qin, B.; and Liu, T. 2015a. Document modeling with gated recurrent neural network for sentiment classification. In *EMNLP*, 1422–1432.

Tang, D.; Qin, B.; and Liu, T. 2015b. Learning semantic representations of users and products for document level sentiment classification. In *ACL*, 1014–1023.

Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; and Hovy, E. 2016. Hierarchical attention networks for document classification. In *NAACL-HLT*, 1480–1489.

Yang, M.; Tu, W.; Wang, J.; Xu, F.; and Chen, X. 2017. Attention based lstm for target dependent sentiment classification. In *AAAI*, 5013–5014.

Yang, H.; Hu, Q.; and He, L. 2015. Learning topic-oriented word embedding for query classification. In *PAKDD*, 188–198. Springer.

Yessenalina, A., and Cardie, C. 2011. Compositional matrix-space models for sentiment analysis. In *EMNLP*, 172–182. Association for Computational Linguistics.

Zhai, S., and Zhang, Z. M. 2016. Semisupervised autoencoder for sentiment analysis. In *AAAI*, 1394–1400.

Zhu, X.; Guo, H.; Mohammad, S.; and Kiritchenko, S. 2014. An empirical study on the effect of negation words on sentiment. In *ACL*, 304–313.