A topic-enhanced recurrent autoencoder model for sentiment analysis of short texts

Shaochun Wu*, Ming Gao, Qifeng Xiao and Guobing Zou

Department of Intelligent Information Processing,
Shanghai University,
Shanghai 200444, China
Email: scwu@shu.edu.cn
Email: qywtgm950120@foxmail.com
Email: xqf_shu@163.com
Email: gbzou@shu.edu.cn
*Corresponding author

Abstract: This paper presents a topic-enhanced recurrent autoencoder model to improve the accuracy of sentiment classification of short texts. First, the concept of recurrent autoencoder is proposed to tackle the problems in recursive autoencoder including 'increasing in computation complexity' and 'ignoring the natural word order'. Then, the recurrent autoencoder model is enhanced with the topic and sentiment information generated by joint sentiment-topic (JST) model. Besides, in order to identify the negations and ironies in short texts, sentiment lexicon is utilised to add feature dimensions for sentence representations. Experiments are performed to determine the feasibility and effectiveness of the model. Compared with recursive autoencoder model, the classification accuracy of our model is improved by about 7.7%.

Keywords: short texts; sentiment analysis; recurrent autoencoder; recurrent neural network; joint sentiment-topic model.

Reference to this paper should be made as follows: Wu, S., Gao, M., Xiao, Q. and Zou, G. (xxxx) 'A topic-enhanced recurrent autoencoder model for sentiment analysis of short texts', *Int. J. Internet Manufacturing and Services*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes:

This paper is a revised and expanded version of a paper entitled 'A topic-enhanced recurrent autoencoder model for sentiment analysis of short texts' presented at ICSS2018, Shanghai University, 13 May 2018.

1 Introduction

With the rapid development of the internet, various social media platforms appear one by one, such as Facebook, Twitter and so on. However, as the carriers of public opinions, these platforms have generated a large number of text messages called 'short texts'. At present, short texts have become a kind of medium of communication and emotional interaction, which profoundly influence people's daily life. Sentiment analysis of these short texts will give us a timely understanding of people's opinions and emotional tendencies in different events. In addition, it has been found that sentiment analysis of short texts can also help us predict the stock trend, crude oil price and even presidential election results. Therefore, sentiment analysis of short texts has been a hot topic in the field of text mining in recent years.

Nowadays, there are two main research approaches for sentiment analysis: lexicon-based approaches and machine learning (Giachanou and Crestani, 2016). The lexicon-based approach uses a manually or automatically built list of positive and negative terms to derive the polarity of the message. The machine learning approach employs a machine learning method and a number of different features to build a classifier that can detect the sentiment tendency of texts. Although both of these methods have had a good performance in sentiment analysis of short texts, there are still some factors that would influence the accuracy of classification, such as topics. Take the example of the following two sentences, we focus on the word 'small' and observe its sentiment tendency.

- My mobile phone is so *small* that I can put it in my pocket.
- My mobile phone is so *small* that I can't see the words on the screen.

In these two sentences, obviously, the former 'small' denotes a positive sentiment, but the latter one expresses a negative sentiment. The observation demonstrates that a word may have different sentiments on different topics. This encourages us to consider the topic information on short texts for sentiment analysis task.

In this paper, we present a hybrid approach which combines machine learning and lexicon-based methods and utilises the topic information on short texts to achieve a better performance in sentiment classification.

2 Related work

Feature learning and word representation play an important role in sentiment analysis of short texts. Word is the smallest unit of understanding natural language and its high quality of vector representation can help us to perform all kinds of natural language processing tasks, so there are amounts of research work on word embedding and parts of them have made breakthrough progress. Traditional methods of word representation include semantic lexicon-based and one-hot representation. The semantic lexicon-based approach utilises synonyms or hypernyms included in semantic lexicon like WordNet and Probase (Wu et al., 2012) to express the meaning of the word. Although the semantic lexicon-based method is simple and specific, it is difficult to calculate the similarity between the words like 'good' and 'perfect'. One-hot representation is based on the bag-of-word model and it has the same length as the vocabulary size, and only one dimension is 1, with all others being 0. Bag-of-word model is an effective tool, but it has two problems. On the one hand, the high dimension and spare information of word representation limit the classification performance. On the other hand, this model cannot capture semantic relation between words.

With the revival of deep learning, how to generate word embeddings by neural networks has become a research hotspot. Word embeddings generated by neural networks can capture the context information more flexibly and effectively, so they are widely used in various kinds of natural language processing tasks. Recently, recursive autoencoder has been applied to learn sentiment specific word embeddings (Li et al., 2015). However, traditional recursive autoencoder which consists of the recursive neural network and the autoencoder is only used in the datasets where the structure of data is already known (Socher et al., 2011a). Based on traditional recursive autoencoder model, Socher et al. (2011b) presented two significant improvements. One is that Socher's model no longer needs the given tree structure of a sentence. Instead of that, Socher's model learns the best tree structure out of all possibilities in a greed way. Another improvement is that Socher's model uses the sentiment information of a sentence to conduct semi-supervised learning. However, there are still some defects in Socher's model. For example, the computation complexity is increased because of using the greedy algorithm and the importance of natural word order is not considered for sentiment analysis. Besides, the model is also difficult to identify negations or ironies. In order to solve these problems, Socher further proposed a recursive neural tensor network model (Socher et al., 2013), but it is based on TreeBank where the sentiment polarity is tagged by artificial methods so that it cannot be widely used.

Based on the recursive autoencoder model, this paper proposes the recurrent autoencoder model to learn word embeddings according to the natural word order and reduce the computation complexity. Furthermore, the recurrent autoencoder is improved by learning the word embedding with the supervision of topic and sentiment information. Besides, we add sentiment feature dimensions for sentence representations with lexicon to identify negations and ironies so that the accuracy of sentiment classification can be improved.

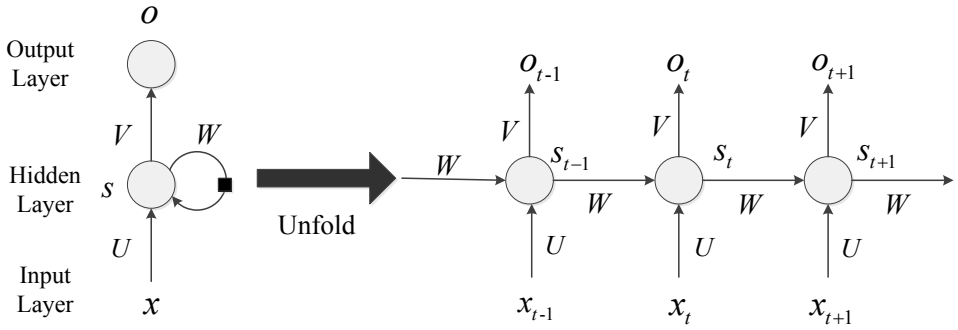
3 Topic-enhanced recurrent autoencoder model

In order to solve these two issues of Socher's recursive autoencoder model: 'increasing in computation complexity' and 'ignoring the natural word order', we first introduce traditional recurrent neural network, and then put forward the recurrent autoencoder model to learn the word embedding. Finally, we discuss how to improve the recurrent autoencoder model with topic information.

3.1 Traditional recurrent neural network

Recurrent neural network (RNN), a deep learning method, is used to deal with sequence data (Mikolov et al., 2011). Different from the traditional neural network, all inputs and outputs of RNN have a connection with each other. The state of a hidden layer is calculated based on both the previous hidden state and the input at the current step (Sutskever et al., 2011). The special structure of RNN can help us complete a lot of tasks which cannot be solved by common neural networks, including predicting the next word in this sentence and other similar tasks of processing the context-sensitive sequence data. A recurrent neural network and the unfolding in time of the computation involved in its forward computation are shown in Figure 1.

Figure 1 A recurrent neural network and the unfolding in time of the computation involved in its forward computation



In Figure 1, x_t is the input at time step t. For example, x_1 could be a word embedding corresponding to the second word of a sentence. s_t is the hidden state at time step t and it is calculated based on both the previous hidden state s_{t-1} and the input at the current step x_t. O_t is the output at step t. There is a non-reversing information flow from the input layer to the hidden layer and another information flow is from the hidden layer to the output layer. As shown in the figure, recurrent structure is existed in hidden layers.

It is assumed that a sigmoid function is used as the activation function of hidden layers and a softmax function is utilised to classify samples in output layers. The forward computation formula of RNN is as follows:

$$s_t = sigmoid(W * s_{t-1} + U * x_t + b) \tag{1}$$

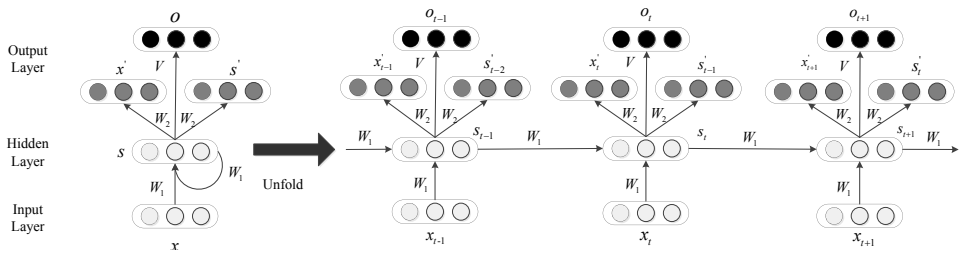
$$o_t = softmax(V * s_t + c) \tag{2}$$

where U, V, W is the matrix of parameters, b and c is the bias term of the sigmoid and the softmax function. s_{-1}, which is required to calculate the first hidden state, is typically initialised to all zeroes.

3.2 The concept of recurrent autoencoder model

Recurrent autoencoder model (RAE), which is composed of the RNN and the autoencoder, is used to learn sentiment specific word embeddings. Firstly, the word embedding of the input layer at the current step will be merged with that of the previous hidden layer through coding networks of autoencoder. Then, we use decoding networks to reconstruct the original word embeddings. Finally, by minimising the reconstruct error, the merged word embedding will maintain the messages of original word embeddings maximally. The merged word embedding will be used as input of the next hidden layer. With such an iterative process, a sentence representation will be obtained when the last word embedding is merged. A recurrent autoencoder model and the unfolding in time of the computation involved in its forward computation are shown in Figure 2.

Figure 2 A recurrent autoencoder model and the unfolding in time of the computation involved in its forward computation



In Figure 2, x'_t and s'_t is the reconstruct word embedding of the input layer and the hidden layer at time step t. There is a recurrent structure between hidden layers. The forward computation formula of the recurrent autoencoder is as follows:

$$s_t = f(W_1(x_t + s_{t-1}) + b_1) \tag{3}$$

where $f(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function, W_1 is a parameter matrix of coding networks, b_1 is a bias term.

Decoding networks are used to reconstruct original word embeddings in RAE model. The computation formula of the reconstructed data is as follows:

$$[x'_t, s'_{t-1}] = f(W_2 s_t + b_2) \tag{4}$$

where $f(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function, W_2 is a parameter matrix of decoding networks, b_2 is a bias term. Because the decoding network is a reciprocal action of the coding network, we let the parameter matrix transpose $W_2 = W'_1$. To evaluate how

much information of the original word embedding that the merged word embedding can carry, this model defines the reconstruction error as the Euclidean distance as follows.

$$E_{rec}([x_t, s_{t-1}]) = \frac{1}{2} \left\| [x_t, s_{t-1}] - [x'_t, s'_{t-1}] \right\| \tag{5}$$

In this section, we propose the RAE model to learn the word embedding. Combined with the characteristics and advantages of the RNN and the autoencoder, this model can utilise the natural word order completely and reduce the computation complexity effectively. Besides, the RAE model can also solve the problem of information loss which exists in the process of tackling long texts through RNN (Rong et al., 2013).

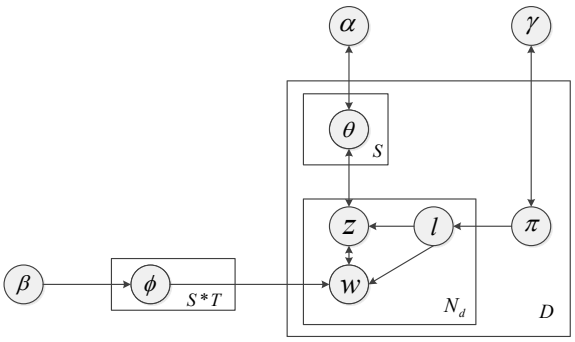
3.3 Topic-enhanced recurrent autoencoder model

Previous works have shown that integrating topic and sentiment information is very useful for some NLP tasks (Ren et al., 2016). Furthermore, researchers have found that the word embedding combined with topic messages could improve the accuracy of sentiment classification. Therefore, we design a topic-enhanced recurrent autoencoder (TRAE) model for sentiment analysis of short texts. First, we use JST model to generate the topic-sentiment combined distribution, and then utilise it to conduct the supervised learning of word embeddings.

3.3.1 Calculate the topic-sentiment combined distribution with JST model

Joint sentiment-topic (JST) model, an extension model of latent Dirichlet allocation (LDA), is used to generate the probability distribution for textual analysis and follows the bag of words hypothesis. Based on the LDA topic model, JST model adds a sentiment layer between the document and the topic layer so that it can analyse the sentiment and topic of a document. In this paper, we use the JST model to generate the topic-sentiment combined distribution. The structure of JST model is shown in Figure 3.

Figure 3 The structure of JST model



As shown in Figure 3, it is assumed that there is a corpus with a collection of D documents denoted by $C = \{d_1, d_2, ..., d_n\}$; each document in the corpus is a sequence of N_dwords denoted by $d = \{w_1, w_2, ..., w_{N_d}\}$. Also, let S be the number of sentiment

labels, and T be the total number of topics, $k = S * T$. The procedure of generating a word w_i in document d is divided into three stages. Firstly, for each document, we choose a distribution $\pi_d \sim Dir(\gamma)$; Secondly, for each sentiment label l under document d, we choose a distribution $\theta_{d,l} \sim Dir(\alpha)$; Finally, for each word w_i, we choose a sentiment label $l_i \sim \pi_d$, a topic $z_i \sim \theta_{d,l_i}$ and a word w_i from the distribution over words defined by the topic z_i and sentiment label l_i, $w_i \sim \phi_{z_i}^{l_i}$. α, β, γ are hyperparameters of Dirichlet.

In our work, Gibbs sampling is adopted to learn JST model. Then, the k-dimensional distribution φ which is combined with topics and sentiments for each document can be generated by JST model.

3.3.2 Learning word embeddings over topics and sentiments

After the topic-sentiment combined distribution for each document is obtained, we use it to conduct the learning of word embeddings. In our model, there is a softmax layer as output over each hidden layer and the computation formula of classification results is as follows.

$$O(s_t; \theta) = softmax(V * s_t + c) \tag{6}$$

where s_t denotes the word embedding of the t^{th} hidden layer, V is a matrix of parameters from the hidden layer to the output layer, c is a bias term and θ is a parameter of the softmax function. Here, we use the cross-entropy error function to calculate the topic-sentiment error between φ and $O(s_t; \theta)$ as follows.

$$E_{top-sent}(s_t, \varphi; \theta) = -\sum_{i-1}^{k} \varphi_i log O_i(s_t; \theta) \tag{7}$$

where φ_i is the i^{th} dimension of the topic-sentiment combined distribution and k is the number of classes.

In the process of learning word embeddings with TRAE, we can get reconstruction error E_{rec} and topic-sentiment error $E_{top-sent}$. Assume that m is the size of training set and the error of the sample s is computed as:

$$E(s, \varphi; \theta) = \sum_{s_t \in s} (E_{rec} + E_{top-sent}) \tag{8}$$

The formula of the final cost function is as follows.

$$J = \frac{1}{m} \sum_{s,\varphi} E(s, \varphi; \theta) + \frac{\lambda}{2} \|\theta\|^2 \tag{9}$$

In order to minimise the loss function, we use the gradient descent method and the formula of gradient is as follows.

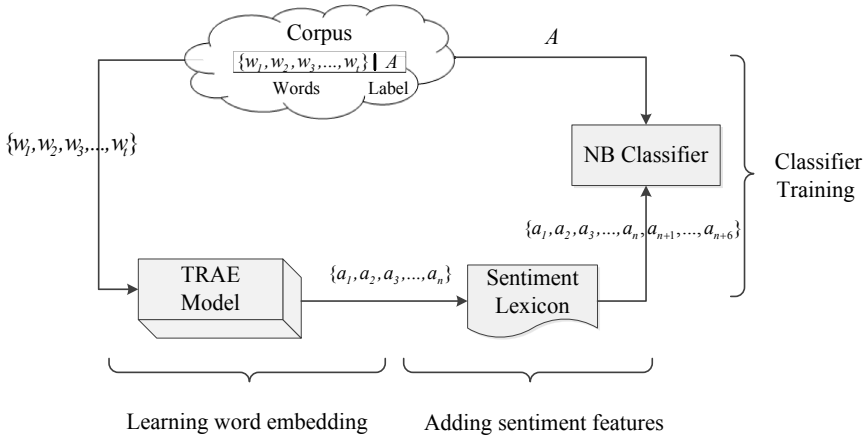
$$\frac{\partial J}{\partial \theta} = \frac{1}{m} \sum_{s,\varphi} \frac{\partial E(s, \varphi; \theta)}{\partial \theta} + \lambda \theta \tag{10}$$

Based on this gradient, we can back propagate errors to influence the model parameters and learn word embedding. When the model parameters are stable, the representation of a sentence will be obtained.

4 The union model for sentiment analysis of short texts

The last merged word embedding generated by TRAE model, namely the sentence representation, can be effectively used for sentiment analysis. However, to further improve the performance, we design a union model for sentiment analysis of short texts. The structure of the union model is shown in Figure 4.

Figure 4 The structure of the union model



As just described in Figure 4, we first use the TRAE model to learn word embeddings. Then, sentiment lexicon is used to add feature dimensions for the sentence representation so that we can identify the negations and ironies in the short text. Finally, naive Bayes classifier is used for sentiment classification.

4.1 Advanced sentence representation combined with sentiment lexicon

In order to identify negations and ironies of short texts effectively, we improve the sentence representation by adding feature dimensions with sentiment lexicon. At present, there are some famous sentiment lexicons such as SentiWordNet, MPQA and so on. In this paper, we use SentiWordNet to extract sentiment features of short texts. Assume that n-dimensional sentence representation is obtained through the TRAE model and we can utilise the sentiment lexicon to get six feature attributes as follows:

1 the percentage of positive words

2 the percentage of negative words

3 the ratio of total positive score to overall score

4 the ratio of total negative score to overall score

5 the percentage of total positive score to total negative score

6 the percentage of the words that are not in the lexicon.

Combined with these six feature attributes, dimensions of sentence representation will be increased to $n + 6$.

4.2 Naive Bayes classifier

Naive Bayes classifier, a classical algorithm for classification, has been widely used in many fields such as image processing, data mining and so on. Especially, it has a great performance in the text classification. In this paper, we use it to classify the sentiment polarity of the advanced sentence representation.

Assume that an advanced sentence representation is denoted by $x = \{a_1, a_2, \ldots, a_n, a_{n+1}, .., a_{n+6}\}$ and the set of sentiment polarities is denoted by $C = \{y_1, y_2, y_3\}$. y_1, y_2, y_3 denotes positive, negative and neutral. According to the Bayes theorem, the probability formula of the advanced sentence representation in the corresponding sentiment polarity is as follows.

$$P(y_i|x) = \frac{P(x|y_i)P(y_i)}{\sum_j P(x|y_i)P(y_i)} \tag{11}$$

$$P(x|y_i)P(y_i) = P(y_i) \prod_{j=1}^{n+6} P(a_j|y_i) \tag{12}$$

Based on the probability formula above, a train set of advanced sentence representations with sentiment labels is used to train the classifier. When the parameters of the classifier become stable, we can predict sentiment polarities of those unlabelled sentence representations.

5 Experimental design and resultant analysis

5.1 Data preparation

In the experiment, we adopted two publicly available datasets: Stanford Twitter Sentiment (STS) and SemEval-2013∼2015 (Rosenthal et al., 2014). The datasets contain a number of tweets crawled from Twitter together with tweets' sentiment and each dataset is divided into a train set and a test set. STS is combined with both positive and negative samples. Samples in SemEval are labelled as positive, negative and neutral. The specific description of these two datasets is shown in Table 1.

Table 1 The information of datasets

Dataset	*Positive*	*Negative*	*Neutral*	*Total*
STS-train	800,000	800,000	0	1,600,000
STS-test	182	177	0	359
SemEval-train	7,000	3,000	5,000	15,000
SemEval-test	1,691	1,084	2,838	5,613

Before training the model, we need to preprocess the textual data and take specific steps including word segmentation, removing stop-words, cleaning noise and so on. Then, we used Word2Vec to train word embeddings of corpus. Word2Vec, a widely adopted toolkit to generate word embedding, is provided by Google (Goldberg and Levy, 2014). Through the training for corpus with Word2Vec, we can get the word embedding with 200 dimensions.

5.2 The design of experiments

In order to test the performance of the topic-enhanced recurrent autoencoder in sentiment analysis of short texts, we adopted a series of contrast experiments. The specific design of experiments is shown in Table 2.

Table 2 The design of contrast experiments

Model	*Description*
Basic	Calculate sentence representation by averaging all word embeddings.
Recurrent neural networks	Generate sentence representation with recurrent neural networks.
Recursive autoencoder	Generate sentence representation with Socher's recursive autoencoder.
Recurrent autoencoder (RAE)	Generate sentence representation with recurrent autoencoder.
RAE + JST (TRAE)	Use JST model to add sentiment and topic information.
RAE + JST + Lexicon	Use sentiment lexicon to add features for sentence representation.

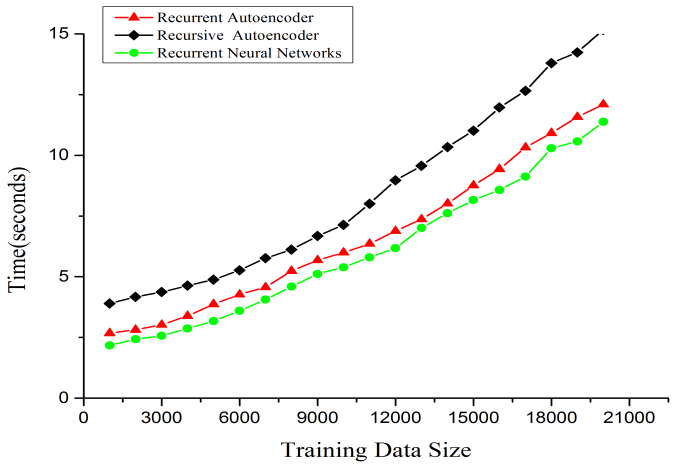
5.3 Result analysis

5.3.1 Time complexity

In the experiment, we compared the time complexities of recursive autoencoder, recurrent neural networks and recurrent autoencoder, respectively. Firstly, we used the train sets in different size to train these models. Then, we got the time complexities of different models corresponding to the training data size as shown in Figure 5.

As described in Figure 5, with the increase of training data size, the time complexities of these three models increase sharply. However, under the same size of training data, the recursive autoencoder model requires more time to learn word embeddings, because it adopts the greedy algorithm to learn the best tree structure out of all possibilities so that the computation becomes more complex and it takes more time for error convergence. Besides, the time complexity of the recurrent autoencoder is slightly higher than that of the RNNs. Since the process of data reconstruction is added in the learning of word embeddings with the recurrent autoencoder model, it requires more time to train the model compared with the RNNs.

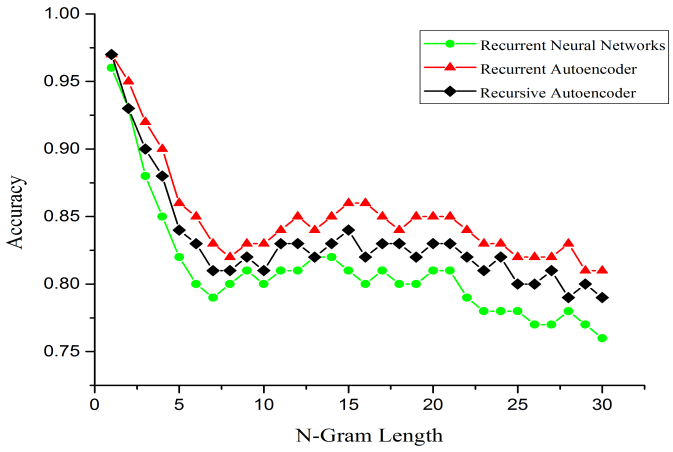
Figure 5 Time complexities of different models corresponding to the training data size (see online version for colours)



5.3.2 The accuracy of sentiment classification

In order to prove that the sentence representation built by the recurrent autoencoder model has a higher quality than that of the recursive autoencoder and the RNNs, we used the test sets with different n-gram length to take a contrast experiment of the classification accuracy. The result of the experiment is shown in Figure 6.

Figure 6 The accuracies of different models with different n-gram length (see online version for colours)



In Figure 6, with the increase of n-gram length, the accuracies of these three models first fall off sharply and then become convergent. Compared with the other two models, we found that the accuracy of the RNNs model descended obviously after the n-gram length surpasses 20. Because the traditional RNNs model is not suitable for the learning of long sentence representation. When the length of sequence data surpasses a threshold, some

problems would appear such as gradient vanishing, gradient exploding and information loss. These problems may lead to the decline in classification accuracy. Besides, in the same conditions, the accuracy of the recurrent autoencoder is the highest, which proves that the recurrent autoencoder model can improve the accuracy of sentiment classification for short texts.

In the experiment, we used the test sets of STS and SemEval to evaluate the performance of recurrent neural networks, recursive autoencoder and recurrent autoencoder, respectively. With the Naive Bayes classifier, we have got the comparative result of these three models, the specific information of the evaluation metrics is shown in Table 3.

Table 3 The design of contrast experiments

Model	*STS-test*			*SemEval-test*		
	Precision (%)	*Recall (%)*	*F-measure (%)*	*Precision (%)*	*Recall (%)*	*F-measure (%)*
Recurrent neural	80.13	79.28	79.70	81.62	80.17	80.89
Networks recursive	82.35	80.62	81.48	83.73	81.22	82.46
Recurrent autoencoder	86.67	84.66	85.65	87.95	85.34	86.63

In Table 3, we listed three evaluation metrics, including precision, recall and F-measure. From the result, we found that the recurrent autoencoder model had greater performance than the other two models in the same dataset. Compared with the recursive autoencoder model, the F-measure of the recurrent autoencoder model increases by about 4%, because the latter model combines the word embedding with the natural word order of a sentence. Based on this result, it is proved that the natural word order plays an important role in the sentiment analysis of short texts.

5.3.3 Result analysis of contrast experiments

In order to evaluate the performance of the topic-enhanced recurrent autoencoder model and the union model, we adopted a series of contrast experiments and the result is shown in Table 4.

Table 4 The design of contrast experiments

Model	*STS*		*SemEval*	
	F-measure (%)	*Improvement (%)*	*F-measure (%)*	*Improvement (%)*
Basic	75.62		76.85	
Recurrent neural networks	79.70	+4.08	80.89	+4.04
Recursive autoencoder	81.48	+1.78	82.46	+1.57
Recurrent autoencoder (RAE)	85.65	+4.17	86.63	+4.17
RAE + JST (TRAE)	87.72	+2.07	88.59	+1.96
RAE + JST + Lexicon	89.13	+1.41	90.16	+1.57

As shown in Table 4, experiment results prove that the union model has a better performance in the sentiment analysis of short texts. Compared with calculating sentence

representation by averaging all word embeddings, the sentence representation built by RNNs has a higher accuracy and its F-measure increases by about 4%. However, the recurrent autoencoder model combines the RNNs and the autoencoder to learn word embedding effectively and its F-measure is improved about 10%. Then, a topic-enhanced recurrent autoencoder is proposed and it utilises the JST model to generate the topic-sentiment combined distribution which is used for conducting the learning of word embeddings. Because of the word embedding combined with topic and sentiment information, the accuracy of sentiment classification is improved again. Finally, in the union model, we used the sentiment lexicon to add feature dimensions for the sentence representation so that negations and ironies can be identified. Compared with the basic method, F-measure is totally improved about 13%.

6 Conclusions

In this paper, a topic-enhanced recurrent autoencoder model for sentiment analysis of short texts is proposed to improve the accuracy of sentiment classification. Firstly, we present the recurrent autoencoder to solve the problems in recursive autoencoder model, including large amounts of computation and not considering the natural word order. Then, we improve the recurrent autoencoder model by using the topic-sentiment combined distribution generated by JST model to conduct the learning of word embeddings. The word embedding built by TRAE is combined with sentiment and topic information. Finally, feature dimensions of the sentence representation are added by the sentiment lexicon, which can improve the efficiency of identifying the negations and ironies in short texts. Through the contrast experiments, it is proved that the topic-enhanced recurrent autoencoder can improve the accuracy of sentiment classification for short texts. In addition, our model still needs to be improved. For example, the robustness of the model needs to be enhanced, because there are always some incorrect spellings in short texts like tweets. Furthermore, due to the quick update of the data in some social media platform, how to track sentiments over time will be our focus in the next work.

Acknowledgements

The authors thank to all anonymous reviewers for their insightful comments and useful suggestions.

References

Giachanou, A. and Crestani, F. (2016) ‘Like it or not: a survey of twitter sentiment analysis methods’, *ACM Computing Surveys*, Vol. 49, No. 2, pp.1–41.

Goldberg, Y. and Levy, O. (2014) ‘Word2Vec explained: deriving Mikolov et al.’s negative-sampling word-embedding method’, *CoRR*, abs/1402.3722.

Li, C., Xu, B., Wu, G., He, S., Tian, G. and Zhou, Y. (2015) ‘Parallel recursive deep model for sentiment analysis’, *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp.15–26.

Mikolov, T., Kombrink, S., Burget, L., Cernocký, J. and Khudanpur, S. (2011) 'Extensions of recurrent neural network language model', *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011*, 22–27 May, Prague Congress Center, Prague, Czech Republic, pp.5528–5531.

Ren, Y., Wang, R. and Ji, D. (2016) 'A topic-enhanced word embedding for Twitter sentiment classification', *Inf. Sci.*, Vol. 369, pp.188–198 [online] https://doi.org/10.1016/j.ins.2016.06.040.

Rong, W., Peng, B., Ouyang, Y., Li, C. and Xiong, Z. (2013) 'Semi-supervised dual recurrent neural network for sentiment analysis', *IEEE 11th International Conference on Dependable, Autonomic and Secure Computing, DASC 2013*, 21–22 December, Chengdu, China, pp.438–445.

Rosenthal, S., Ritter, A., Nakov, P. and Stoyanov, V. (2014) 'Semeval-2014 task 9: sentiment analysis in Twitter', *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014*, 23–24 August, Dublin, Ireland, pp.73–80.

Socher, R., Huang, E.H., Pennington, J., Ng, A.Y. and Manning, C.D. (2011a) 'Dynamic pooling and unfolding recursive autoencoders for paraphrase detection', *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a Meeting*, 12–14 December, Granada, Spain, pp.801–809.

Socher, R., Pennington, J., Huang, E.H., Ng, A.Y. and Manning, C.D. (2011b) 'Semi-supervised recursive autoencoders for predicting sentiment distributions', *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, A Meeting of SIGDAT, A Special Interest Group of the ACL*, 27–31 July, John McIntyre Conference Centre, Edinburgh, UK, pp.151–161.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A.Y. and Potts, C. (2013) 'Recursive deep models for semantic compositionality over a sentiment treebank', *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, A Meeting of SIGDAT, A Special Interest Group of the ACL*, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, pp.1631–1642.

Sutskever, I., Martens, J. and Hinton, G.E. (2011) 'Generating text with recurrent neural networks', *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, 28 June–2 July, Bellevue, Washington, USA, pp.1017–1024.

Wu, W., Li, H., Wang, H. and Zhu, K.Q. (2012) 'Probase: a probabilistic taxonomy for text understanding', *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, SIGMOD '12*, pp.481–492.