Document Sentiment Classification based on the Word Embedding

Yanping Yin[1, a], Zhong Jin[1, b]

[1]School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, 210000, China

[a]email: jan-yyp@163.com, [b]email: zhongjin@njust.edu.cn

Keywords: Word Embedding; Support Vector Machine; Sentiment Classification

Abstract. N-gram feature is commonly used to represent document, however, it often leads to the curse of dimensionality. Sentiment classification based on word embedding and SVM is proposed. The method uses word embedding to represent document, which can make the final representation of the document consistent with the dimension of word embedding. Experiments show that the proposed method can significant reduce the dimension of document representation and improve the accuracy of document sentiment classification.

Introduction

The target of sentiment classification is to classify opinion documents into two folders, positive and negative sentiment, which is very helpful in business intelligence application and recommender systems, etc.[1] Most of the existing techniques for document-level sentiment classification are based on supervised, semi-supervised or unsupervised learning theory [2].

An approach of sentiment classification, named SVM-WE, is proposed in this paper, which uses word embedding to represent document and selects support vector machine as classifier. More specifically, the word embedding trained by skip-model are firstly accumulated to represent document, and then support vector machine is used to classify the represented features as it always outperforms other classifiers in most cases.

In this paper, we will not only compare the influence of CBOW model and Skip-gram model on trained word embedding, but also analyze the effect of different dimension of word embedding on the performance of sentiment classification. In addition, considering the different frequency and sentiment tendentiousness of word, an extension of SVM-WE which uses weighted sum of word embedding to represent document is also presented in this paper.

The remainder of this paper is organized as follows: Some related work are reviewed in the second section. The third section details the SVM-WE method and its extension. Experiment results and analysis are provided in the fourth section. Finally, some conclusions and discussions about the method, SVM-WE, are presented.

Related Work

As is known to all, natural language processing is based on word as word is the smallest linguistic units with independent meaning. There are two ways commonly used to represent word, distributed representation and one-hot representation, of which the latter is more intuitive. In one-hot representation, word is represented as a Boolean vector whose length is equal to the size of vocabulary. For every word, the position corresponding to the word in the representation vector is set to one and the remainder are set to zeros, which make the feature convenient to be stored by a sparse model, such as just assign a number ID to each word. Though the one-hot representation approach is used widely as it is simple and easy to implement, its shortcomings is significantly. For example, in the representation space each vector is independent even the original words are very similar [3].

Distributed representation of words, also known as word embedding, which was originally proposed by Hinton in 1986 is different from one-hot representation [4]. Word embedding

© 2015. The authors - Published by Atlantis Press

represents word as a low dimensional vector trained by language model and makes the related or similar words closer in the vector space. Thus it can overcome the disadvantage of one-hot representation that feature vectors cannot reflect the dependency relationship between words. Bengio [5], Collobert [6], Mikolov [7], Huang [8] had ever proposed different language model to train word embedding. Bengio used a language model built by three-layer neural network to train the feature vector. Collobert gained the word embedding by a method simplified the output layer of neural network and realized the part-of-speech tagging, named entity recognition, phrase recognition, semantic role labeling and other natural language processing tasks based on the vectors. Recurrent neural network was used by Mikolov as a language model which made full use of documental information. Huang improved the model proposed by Collobert and promoted the semantic component in the word embedding. Currently, the most popular models of word embedding are continuous bag-of-words model (hereafter referred to CBOW) and continuous skip-gram model (hereafter referred to Skip-gram) which presented by Mikolov in 2013 [9] [10].

Sentiment classification aims to distinguish whether people like/dislike a product from their reviews. It has emerged as a proper research area. In general, the supervised learning has a better performance than unsupervised learning and semi-supervised learning. Pang et al. (2002) [11] who first used machine learning to do sentiment classification tried to classify movie reviews into positive/negative by using three different classifiers – Naïve Bayes, Maximum Entropy and SVM. They tested different feature combinations such as unigrams, unigrams + bigrams and unigrams + POS (part-of-speech) tags, etc. The experimental results showed that SVM combined with unigrams obtained the best performance. In recent years, there has been a growing interest in enhancing classification accuracy by different techniques [12] [13].

SVM-WE Based on Word Embedding

In this paper, word embedding are trained by a Google toolkit, named word2vec, which realizes CBOW model and Skip-gram model. These two models are shown in figure 1.

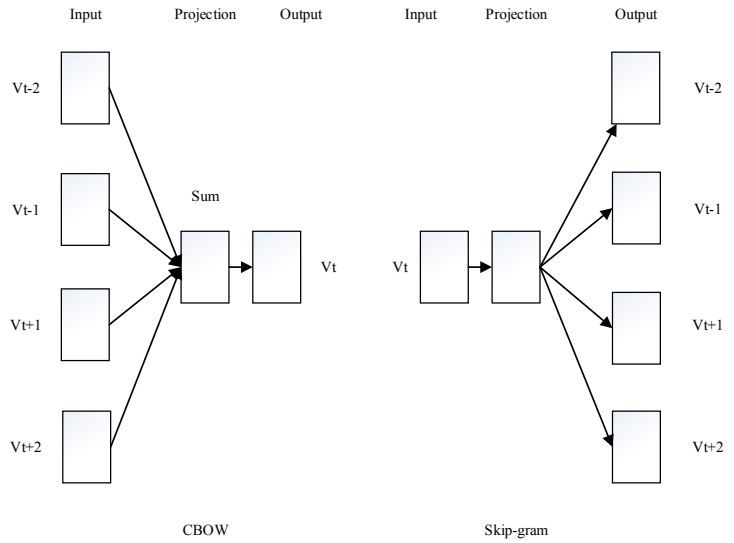


Fig.1. The structure of CBOW and Skip-gram model

As can be seen from the figure, the structure of CBOW model is similar to the structure of feedforward neural network language model. The only difference is that CBOW model removes the nonlinear hidden layer which is most time-consuming and then it just has the input, projection and output layer. The projection of CBOW is shared for all words, thus, all words get projection into the same position. Skip-gram is similar to CBOW, but instead of predicting the current word based on the document, it tries to maximize classification of a word based on another word in the same sentence. More precisely, it uses each current word as an input to a log-linear classifier with continuous projection layer, and predicts words within a certain range before and after the current word.

CBOW and Skip-gram have different advantages. Word embedding trained by CBOW contains more syntax information which can obtain a better result in syntax test while trained by Skip-gram contains more semantic information which can perform better in semantic test. Obviously, the semantic information plays a more important role in sentiment, as several sentiment words which are not grammatical can also express the sentiment of document. Thus, in the model, SVM-WE, proposed in this paper, Skip-gram is used to train the word embedding.

Mikolov found that simple word embedding addition can often produce meaningful results. For example, the sum of word embedding of Germany and capital is close to word embedding of Berlin. So we speculate that the sum of word embedding is meaningful and can represent the document.

The definition of document representation based on word embedding is as follows:

Given a document $D(t_1, v_1; t_2, v_2; \ldots; t_n, v_n)$ where $t_1, t_2, \ldots, t_n$ are words in document and v_i ($(v_{i1}, v_{i2}, \ldots, v_{im})$) is word embedding of word t_i. The representation of document is

$$D = \sum_{i=1}^{n} v_i = (\sum_{i=1}^{n} v_{i1}, \sum_{i=1}^{n} v_{i2}, \ldots \sum_{i=1}^{n} v_{im}) \quad (1)$$

This paper proposed a method called SVM-WE which combines word embedding based document representation and SVM. The flow diagram of SVM-WE is showed in figure 2. This method has three steps.

Step 1: Use word2vec toolkit to train word embedding and to obtain a set of word embedding.

Step 2: Represent train documents and test documents according to formula (1) and in which the word embedding is obtained by step 1.

Step 3: Use SVM classifier trained by the train documents to classify test documents.

Similar to weighted feature in vector space model, we can also give weights to words which measure the importance of the words in the document. Different weighting method [14] is given in Table 1.

Table 1. w_i and tf_i represent the weight and the frequency of t_i in document D respectively.

Type	formula
TF	$w_i = tf_i$
IDF	$w_i = \log(N / n)$
TF-IDF	$w_i = tf_i \times \log(N / n)$
TFC	$w_i = \frac{tf_i \times \log(N / n)}{\sqrt{\sum_{t_i \in D} [tf_i \times \log(N / n)]^2}}$
ITC	$w_i = \frac{\log(tf_i + 1) \times \log(N / n)}{\sqrt{\sum_{t_i \in D} [\log(tf_i + 1) \times \log(N / n)]^2}}$

n is the number of document which has word t_i, N is the number of documents.

The extension definition of document representation is can be formulated as:

$$D = \sum_{i=1}^{n} w_i v_i = (\sum_{i=1}^{n} w_i v_{i1}, \sum_{i=1}^{n} w_i v_{i2}, \ldots \sum_{i=1}^{n} w_i v_{im}) \quad (2)$$

Similar to the method SVM-WE, we use the sum of weighted word embedding to represent document and use SVM to classify. It is not difficult to find that formula (1) is a particular form of formula (2) with Boolean weighting.

Experiment and Analysis of Test Results

The experiment in this section involves three aspects. Firstly, we validate the effective of using Skip-gram rather than CBOW to train word embedding in SVM-WE. Secondly, the performances of SVM-WE and other methods [15], SVM-uni and SVM-bi, which use unigram and bigram as

feature to represent document respectively and then use SVM classify the features, are compared. Last but not least, we also analyze the extension method of SVM-WE.

We compare with published results on the following datasets. Detailed statistics are shown in table 2.

RT-s: Short movie reviews dataset containing one sentence per view (Pang and Lee, 2005).

CR: Customer reviews dataset (Hu and Liu, 2004) processed like in (Nakagawa et al., 2010).

RT-2k: The standard 2000 full-length movie review dataset (Pang and Lee, 2004).

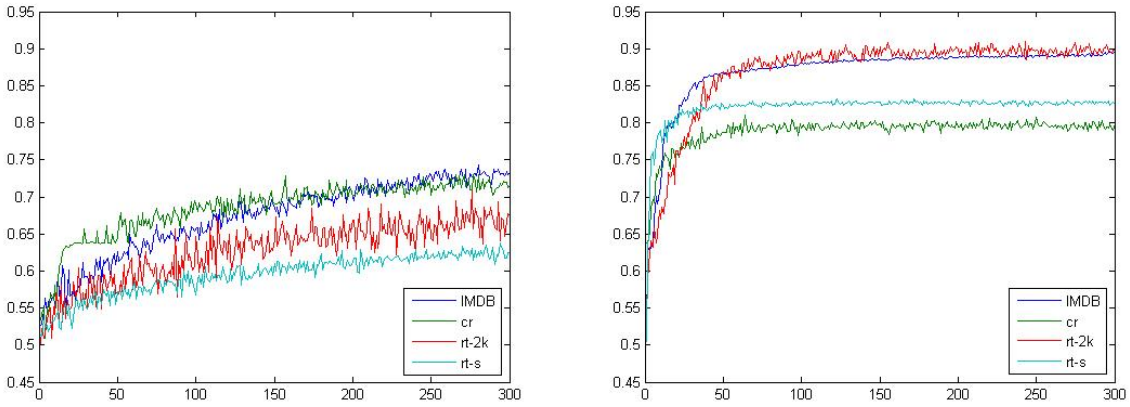
IMDB: A large movie review dataset with 50k full-length reviews (Maas et al., 2011).

Table 2. Dataset statistics. $(N+, N-)$: number of positive and negative examples.

Dataset	$(N+, N-)$	l	$\|V\|$	CV
RT-s	(5331,5331)	21	21000	10
CR	(2406,1367)	20	5713	10
RT-2k	(1000,1000)	787	51000	10
IMDB	(25000,25000)	231	392000	N

l : average number of words per example. $|V|$: the vocabulary size. CV : number of cross-validation splits, or N for train/test split.

Figure 2 shows the result of document sentiment classification based on CBOW and Skip-gram respectively. The horizontal axis is the dimension of word embedding which ranges from 1 to 300 and the vertical axis shows the accuracy of classification.



(a) The result of CBOW (b) The result of Skip-gram

Fig.2. The result of CBOW and Skip-gram

From the figure 2, we can see the accuracy of classification is promoted on the whole with the growth of dimension of word embedding. And before a certain dimension, the accuracy increases obviously while levels off after that dimension. So it can be speculated that a few dimension of word embedding contains enough information to deal with the sentiment classification and higher dimension of word embedding just contains more redundant information which can't improve the accuracy obviously. Of course, as the training of word embedding is an optimized process, fluctuates of accuracy within a reasonable range is acceptable. We can also find that higher accuracy and smaller fluctuation can be obtained when Skip-gram is used to train word embedding rather than CBOW. This result verifies that Skip-gram performs better in semantic test and word embedding trained by Skip-gram has more accurate information as the document representation used by SVM-WE doesn't consider the grammatical information. For the good performance of word embedding trained by Skip-gram, it is used in SVM-WE to train word embedding.

We analyze the influence of dimension of word embedding on sentiment classification as detailed above. According to the results of figure 3, we train word embedding of 50, 50, 100, 200 dimension respectively on dataset RT-s, CR, RT-2k, IDBM. Table 3 shows the results of SVM-WE,

SVM-uni and SVM-bi on dataset RT-s, CR, RT-2k, IDBM.

Table 3. The results for datasets.

Dataset\Method	SVM-uni	SVM-bi	SVM-DR
RT-s	0.7620	0.7770	**0.8260**
CR	0.7900	0.8080	**0.8092**
RT-2k	0.8625	0.8740	**0.9056**
IDBM	0.8695	0.8916	**0.8920**

As can be seen from table 3, SVM-WE acquires better results than SVM-uni and SVM-bi, which demonstrates that the sum of word embedding trained by Skip-gram can represent document effectively and outperform other representation with unigram or bigram as features.

We make an extension to SVM-WE by considering the weight of word. Table 4 shows the results of SVM-WE and results of the extension of SVM-WE that uses different weights provided in table 1. And the word embedding of 50, 50, 100, 200 dimension are trained respectively on dataset RT-s, CR, RT-2k, IDBM according to table 3.

Table 4: The results of SVM-WE and extension of SVM-WE

Dataset	SVM-DR	TF	IDF	TF-IDF	TFC	ITC
RT-s	**0.826**	0.825	0.816	0.819	0.818	0.820
CR	**0.809**	0.793	0.785	0.790	0.786	0.800
RT-2k	**0.905**	0.890	0.890	0.892	0.887	0.890
IDBM	**0.892**	0.887	0.873	0.879	0.877	0.882

As seen in table 4, considering weights of words doesn't improve the accuracy. We speculate this result may be due to that the process of training word has considered the importance of words in document. Thus calculating the weight of words doesn't increase the useful information but increase the redundant information.

Conclusion

Combined with word embedding and SVM classifier, we propose a method called SVM-WE to classify the sentiment document. And word embedding trained by Skip-gram can retain semantic information and avoid the curse of dimensionality which may be generated by. At the same time, SVM can perform a bit better than other classifiers. In addition, as the extension of SVM-WE doesn't improve the performance, we further intend to take sentiment dictionary into account. We plan to give different weights for sentiment words and non-sentiment words. So how to give weights to sentiment words and non-sentiment words is a worthy direction for further study.

Acknowledgement

In this paper, this work is partially supported by National Natural Science Foundation of China under Grant Nos. 61373063, 61233011, 61125305, 61375007, 61220301, and by National Basic Research Program of China under Grant No. 2014CB349303.

References

[1] Zhao Yanyan, Qin Bing, Liu Ting. Sentiment classification [J]. Journal of Software. 2010, 21(8): 1834-1848.

[2] Wang G, Sun J, Ma J, et al. Sentiment classification: The contribution of ensemble learning [J]. Decision support systems, 2014, 57: 77-93.

[3] Zhang jian, Qu dan, Li zhen. Recurrent neural network language model based on word vector features [J]. Pattern recognition and artificial intelligence. 2015, 28(4): 299-305.

[4] Hinton G E. Learning distributed representations of concepts [C]. Proceedings of the eighth annual conference of the cognitive science society, Amherst: 1986.

[5] Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model [J]. The Journal of Machine Learning Research, 2003, 3: 1137-1155.

[6] Collobert R, Weston J, Bottou L, et al. Natural language processing (almost) from scratch [J]. The Journal of Machine Learning Research, 2011, 12: 2493-2537.

[7] Mikolov T, Kombrink S, Burget L, et al. Extensions of recurrent neural network language model [C]. Proceedings of the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing. Prague: IEEE, 2011.

[8] Huang E H, Socher R, Manning C D, et al. Improving word representations via global condocument and multiple word prototypes [C]. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics. Stroudsburg: Association for Computational Linguistics, 2012.

[9] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space [J]. arXiv preprint arXiv:1301.3781, 2013.

[10] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality [C]. Proceedings of Advances in neural information processing systems. Harrahs and Harveys: NIPS, 2013.

[11] Pang B, Lee L, Vaithyanathan S. Thumbs up?: sentiment classification using machine learning techniques [C]. Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10. Stroudsburg: Association for Computational Linguistics, 2002.

[12] Li S, Lee S Y M, Chen Y, et al. Sentiment classification and polarity shifting [C]. Proceedings of the 23rd International Conference on Computational Linguistics. Stroudsburg: Association for Computational Linguistics, 2010.

[13] Xia R, Zong C, Li S. Ensemble of feature sets and classification algorithms for sentiment classification [J]. Information Sciences, 2011, 181(6): 1138-1152.

[14] Zong Chengqin. Statistical natural language processing [second edition]. Tsinghua University Publishing House, 2013: 416-433.

[15] Wang S, Manning C D. Baselines and bigrams: Simple, good sentiment and topic classification [C]. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2. Stroudsburg: Association for Computational Linguistics, 2012.