Apply Word Vectors for Sentiment Analysis of APP Reviews

Xian Fan, Xiaoge Li, Feihong Du, Xin Li
School of Computer Science and Technology
Xi'an University of Posts & Telecommunications
Xi'an, China

Mian Wei
School of Computer Science and Technology
Tulane University New Orleans
LA 70118, USA

Abstract—**Vector representations for language have been shown to be useful in a number of Natural Language Processing tasks. In this paper, we aim to investigate the effectiveness of word vector representations for the problem of Sentiment Analysis. In particular, we target three sub-tasks namely sentiment words extraction, polarity of sentiment words detection, and text sentiment prediction. We investigate the effectiveness of vector representations over different text data and evaluate the quality of domain-dependent vectors. Vector representations has been used to compute various vector-based features and conduct systematically experiments to demonstrate their effectiveness. Using simple vector based features, we achieve F1 85.77%, the recall 85.20%, and the accuracy 86.35% for text sentiment analysis of APP reviews.**

Keywords-component; vector representations; sentiment words extraction; polarity of sentiment words detection; sentiment analysis

I. INTRODUCTION

Mobile App stores contain millions of Apps which users can download and install on their smart phones. Users can feedback their opinions and sentiment of Apps, which have influence on prospective user decisions and help the developers to evaluate its quality.

Sentiment analysis has been practiced on a variety of topics based on sentiment lexicons [1]–[3]. There are some methods have been proposed for sentiment lexicons building, including thesaurus-based and corpus-based. Thesaurus-based methods construct the sentiment lexicons from common dictionary or HowNet. However, the sentiment words are the domain depended, some sentiment words may have different polarity on different domain. the corpus-based methods include the point mutual information (PMI) method [4], dependency relation analysis [5] and bootstrapping [6]. All these methods need to manual annotate the text, select the seed words and pattern carefully on the specific domain, which are very time costly and sometimes difficult.

Unsupervised vector-based approaches to semantics can model rich lexical meanings, word emending encodes continuous similarities between words as distance or angle between word vectors in a high-dimensional space. In the vector space, as such the degree of similarity between words can be calculated by the distance between vectors. This method is often used in the calculation of semantic similarity [7], and has been successfully used in many NLP tasks [8]–[10].

This work is supported by the National Natural Science Foundation of China (61373116), the Fundamental Research Funds for the key subjects of the universities in Shaanxi Province (112-1602).

In this paper, we present a method using word embedding to build sentiment lexicon, then judge the polarity of sentiment words from dataset of user comments, finally naive Bayes [11] is used to classify the represented features on massive dataset of user reviews from APP store, we obtain the effect of different sentiment lexicon on the text sentiment analysis. As a result, the experiment shows that it is effective to use word vector to build sentiment lexicon for text sentiment analysis.

The remainder of this paper is organized as follow: Section 2 introduces the system architecture and implementation, Sections 3 describes experimental result, Sections 4 presents our conclusion and gives an overview of future work.

II. SYSTEM ARCHITECTURE AND IMPLEMENTATION

A. System Archtecture

This paper aims to build sentiment lexicon and predict text sentiment, the basic framework of the system is shown in Fig. 1. Firstly, the dataset from APP store are tokenized by Chinese segmentation system [12], then word vectors have been learnt to build sentiment lexicon on App reviews domain by Google word2vec [13]. In this case, a few sentiment seeds are extracted from a massive dataset of user comments, subsequently, they are expanded by calculating the distance between all words containing sentiment seeds. Then we predict the polarity of sentiment words since these words don't capture the sentiment information. Finally, we classify the sentiment of users' reviews by the classical naive Bayes.

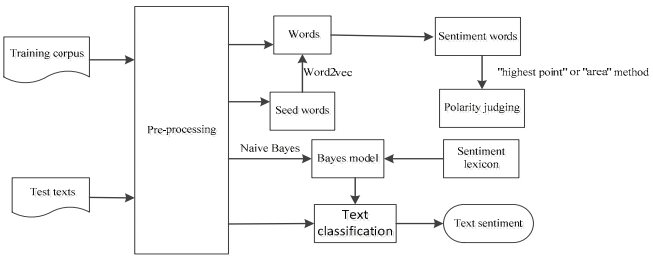

Fig. 1. The framework of the system

B. Construction of Sentiment Lexicon Using Word2vec

Since most of sentiment words are adjective, noun, verb [14], we remove function words from corpus. We select the most frequency （>100）appearing in corpus with 4 or 5-stars

as initial positive seeds, the negative seeds selected from 1 to 3-stars users' comments. The sample of the words is shown in TABLE I.

TABLE I. THE SAMPLE OF THE SENTIMENT SEEDS

Positive words	不错 | great, 力挺 | support, 喜欢 | like, 好 | good, 好好 | fine, 棒 | wonderful, 行 | fantastic
Negative words	卡| lag, 卸载 | uninstall, 坑 | rip-off, 垃圾 | rubbish, 差 | disappointing, 慢 | slow, 死 | die, 退 | drop-out, 闪 | bad

Then we convert all words to vectors by word2vec. The word2vec includes Continuous Bag-of-Words model (CBOW) and Skip-gram model, Hierarchical Softmax and Negative Sampling strategies [15], [16]. For a sentence S composed by words $w_1, w_2, w_3, ..., w_t$, the objective of the CBOW as shown in (1), the Skip-gram is shown in (2).

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\ne 0}\log p\left(w_t \mid w_{t+j}\right), \tag{1}$$

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\ne 0}\log p\left(w_{t+j} \mid w_t\right). \tag{2}$$

where w_t is a word in sentence S, w_{t+j} is the context of w_t. $p\left(W_O \mid W_I\right)$ is defined using the softmax function:

$$P\left(W_O \mid W_I\right) = \frac{\exp\left(V_{w_o}^{'T} V_{w_I}\right)}{\sum_{w=1}^{w}\exp\left(V_w^{'T} V_{w_I}\right)}. \tag{3}$$

where V_w and $V_w^{'}$ are the "input" and "output" vector representations of w, and W is the number of words in the vocabulary. In order to simplify the equation and decrease computation quantity, word2vec introduces Hierarchical Softmax and Negative Sampling for each model.

When generate vectors for words by word2vec, according to the cosine formula, we can get all words that are associated with the given word. For example, the part of words associated with "垃圾 | junk" listed in TABLE II.

TABLE II. THE PART OF WORDS ASSOCIATED WITH "垃圾 | JUNK"

	Word	Similarity
1	狗屎 | shit	0.508662
2	差劲 | bad	0.455745
3	渣渣 | terrible	0.439946
4	真尼玛 | fuck	0.435789
5	饭桶 | good-for-nothing	0.432835
6	完蛋 | dead	0.426218
7	垃圾桶 | rubbish	0.419206
8	太尼玛 | worst	0.413044
9	冒火 | anger	0.411646
10	恶心 | nausea	0.409563

So the building of sentiment lexicon is described as follows:

1) Convert the words to vectors in the vector space using word2vec.

*2) According to the cosine formula, expand sentiment seeds:*Calculate the cosine values between all words and sentiment seeds, choose the top 20 to join the sentiment seeds.

3) To judge whether sentiment seeds change: If there is no change, the sentiment lexicon is complete, that are all sentiment seeds. Otherwise, jump to the second step.

C. Predicting the Polarity of Sentiment Words

According to word2vec, the polarity of sentiment words can't be predicted. Therefore, the polarity of a sentiment word has to be done according to the score of text [17].

There are two methods to be considered for predicting the polarity of a sentiment word. One is the "highest point" method, which select the most frequent words appearing in the positive and negative comments to determine the polarity of sentiment words. According to the frequency of the sentiment word in the text with different score, choose the sentiment of the text in which the frequency of the sentiment word is the most as the sentiment of the word. When the sentiment of the text is positive, it is considered that the word is positive, otherwise the word is negative.

The other is the "area" method. According to the frequency of the sentiment word in the area of the positive texts and the negative texts. When the frequency of the sentiment word in positive texts is greater than in negative texts, it is considered the word is positive, if not, the word is negative.

D. Naive Bayes Classifier

After building the sentiment lexicon, all the sentiment words are used as the features of the text, combined with the naive Bayes we can predict the sentiment of the text. In this paper, assume W is the feature vector of the text, w is the value of W, j is the dimension of W. Y is the classification mark, C is the value of Y. k is the number of the classification. As such the Bayesian classifier can be used as follows:

$$y = f(w) = \arg\max_{c_k}\frac{p\left(Y=c_k\right)\prod_j p\left(W^{(j)}=w^{(j)} \mid Y=c_k\right)}{\sum_k p\left(Y=c_k\right)\prod_j p\left(W^{(j)}=w^{(j)} \mid Y=c_k\right)}. \tag{4}$$

Using the maximum likelihood estimation to calculate the corresponding probability with (5), we can determine the classification y of the test sample w. The priori probabilities is:

$$p(Y=c_k)=\frac{Doc_{c_k}}{Doc}. \tag{5}$$

where Doc is the number of texts in the training corpus, Doc_{c_k} is the number of the texts of corresponding classification c_k. The likelihoods function is:

$$p\left(W^{(j)}=w^{(j)} \mid Y=c_k\right)=\frac{Word_{w^{(j)}}}{Word_{c_k}}. \tag{6}$$

where $Word_{c_k}$ is the sum of the number of all sentiment words in classification c_k. $Word_{w^{(j)}}$ is the number of word $w^{(j)}$ in classification c_k. Assuming that V is the sum of $Word_{c_k}$ in all classification, in order to avoid the posterior probability is 0, we use the Laplace transform for the likelihoods function.

III. RESULT

A. Experimental Data

In this paper, we download 12000 users' comments of video Apps from 360 mobile store as our experimental data. 5000 users' comments are used as the corpus to build sentiment lexicon, 5000 of them are used as training corpus for sentiment analysis, 2000 of them are used as texts for sentiment testing.

B. Experimental Evaluation Criteria

The training corpus is divided into two categories: positive and negative. When testing the performance of Classifier, take macro average precision, macro average recall and macro average F1 value as measures [1]. N represents the number of categories, $true(c_k)$ represents the number of proper classifications under the c_k category, $doc(c_k)$ represents the number of texts under the c_k category, $response(c_k)$ represents the initial number of texts under the c_k category. The calculation equations as follows.

$$MacroP=\frac{1}{N}\sum_{k=1}^{N}\frac{true(c_k)}{doc(c_k)}, \tag{7}$$

$$MacroR=\frac{1}{N}\sum_{k=1}^{N}\frac{true(c_k)}{response(c_k)}, \tag{8}$$

$$MacroF1=\frac{MacroP\cdot MacroR\cdot 2}{MacroP+MacroR}. \tag{9}$$

C. Experimental Results and Analyses

In order to study the influence of the number of sentiment seeds on the building of the sentiment lexicon, we choose the different number of sentiment seeds to build the lexicon with word2vec. The horizontal axis represents the number of iterations, the vertical axis represents the number of sentiment words in the lexicon. The result as Fig. 2 shows.

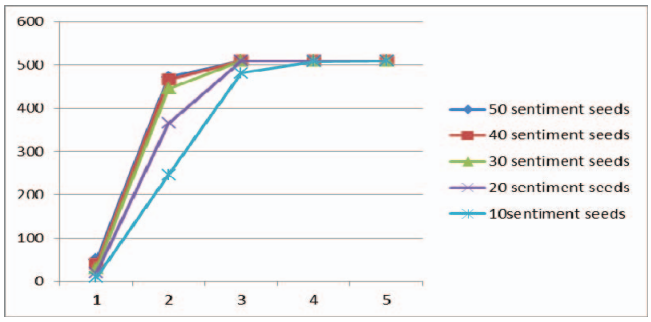

Fig. 2. The influence of the number of sentiment seeds on the sentiment lexicon

It can be seen after the fourth iteration of training by word2vec, the number of the sentiment words in all sentiment lexicons is almost no longer increase. So the number of sentiment seeds has no influence on the building of sentiment lexicon.

In this paper, we study the influence of word2vec's parameters on sentiment analysis for short texts. The CBOW model and Skip-gram model with Hierarchical Softmax (HS) and Negative Sampling (NEG) strategies have been studied, the results are shown in Fig. 3.

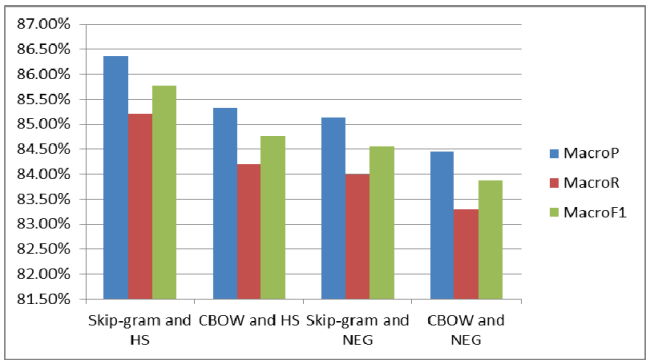

Fig. 3. The influence of word2vec's parameters on text sentiment analysis

As can be seen from the Fig. 3, when using the word2vec to generate the vectors and build sentiment lexicon for short text sentiment analysis, the Skip-gram model and Hierarchical Softmax strategy can improve macro average precision and macro average recall for sentiment analysis. Usually the Skip-gram model and the Hierarchical Softmax strategy have better effects on the low frequency words, CBOW model training

speed is faster and Negative Sampling strategy is better for high frequency words [18].

After getting sentiment words, we predict the sentiment of words by "highest point" and "area" methods respectively. In this paper, the texts are divided into five grades from 1 star to 5 stars. If the score of the text is in 1–3, the sentiment of text is negative. If the score of the text is in 4–5, the sentiment of text is positive [19]. The results of sentiment analysis are shown in Fig. 4.

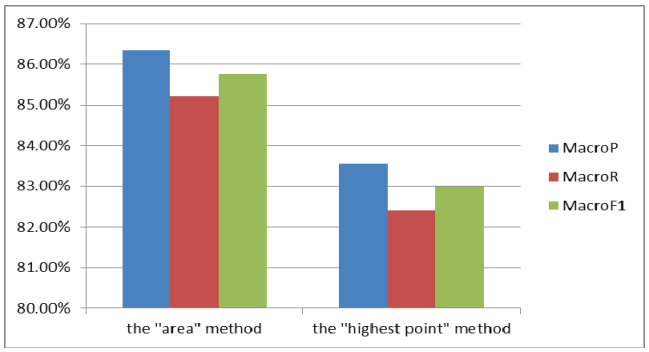

Fig. 4. The influence of different sentiment prediction of words on sentiment analysis

It can be seen when using "area" method, the results of text sentiment analysis are more effective than the "highest point" method. Although the "highest point" method is simple, when the sentiment of word is weak and the frequency of word is almost same, the result of sentiment word is often wrong. "area" method predicts the polarity of word by comparing the frequency of the word in positive and negative texts. When the sentiment of word is weak, it still correctly predicts the sentiment of the word.

Finally, in order to test the influence of different sentiment lexicons on the text sentiment analysis, we compare three sentiment lexicons that include the common lexicon of National Taiwan University, the lexicon built by PMI and the lexicon built by word2vec. The lexicon of National Taiwan University has 2810 positive words and 8276 negative words. The lexicon built by PMI has 2888 positive words and 319 negative words. The lexicon built by word2vec has 268 positive words and 307 negative words. The results are shown in Fig. 5.

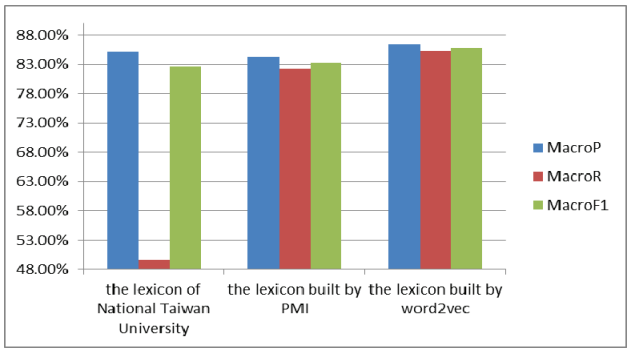

Fig. 5. The influence of different lexicons on sentiment analysis

As seen in the Fig. 5, although National Taiwan University's lexicon can get higher macro average accuracy, many of the sentiment words in the comments are not in the lexicon, so the average recall is low. However, when using the other two lexicons, macro average recall is greatly improved. The precision and recall rate of lexicon built by word2vec is greater than the lexicon built by PMI, and the building of lexicon by word2vec is easier than the building of lexicon by PMI.

IV. Conclusion and Future Work

In this paper, we study the building of sentiment lexicon based on word vectors. Based on the classical naive Bayes, we compare the effect of different sentiment lexicons on the performance of text sentiment analysis. The results show that the lexicon built by word2vec for sentiment analysis gets the higher precision and recall rate. The follow-up work will consider extracting the aspects of the entity described in the text through some methods, for example, LDA [20] or Bootstrapping [6] method. We analyze the sentiment of aspects and make more meaningful researches according to the sentiment of the aspects. For example, the comment "good, user friendly. Blue UI is very beautiful and charming, pictures are very smooth, the speed is also very quick.", we can extract the "UI", "picture" and "speed" as aspects which the sentiment can be analyzed. Using these we can get advantages and disadvantages of the software and improve the quality of software in a targeted fashion.

Acknowledgment

It gives us great pleasure to attend the 2016 3rd International Conference on Systems and Informatics (ICSAI 2016). We would like to thank the anonymous reviewers for their comments. We thank members of the text mining Laboratory for providing their data sets and for their comments.

References

[1] L Zhou, B Li, W Gao, Z Wei, KF Wong. Unsupervised discovery of discourse relations for eliminating intra-sentence polarity ambiguities. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 162–171, Edinburgh, Scotland, UK. Association for Computational Linguistics.

[2] Zhao Y, Qin B, Liu T. Collocation polarity disambiguation using web-based pseudo contexts. Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, 2012.

[3] Baccianella S, Esuli A, Sebastiani F. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. International Conference on Language Resources and Evaluation, Lrec 2010, 17-23 May 2010, Valletta, Malta. 2010:83-90.

[4] Yazhen Li , Xiaoge Li, Gen Yu. Research of Sentiment Classification Based on the Chinese Stock Blog. Journal of Wuhan University (Natural Science Edition), 2015, 61(2):163-168.

[5] Yan Sun, Xueguang Zhou, Wei Fu. Opinion Words Expansion Based on Dependency Parsing. Journal of Beijing University of Posts and Telecommunications, 2012, 35(5):90-93.

[6] Changhou Wang, Fei Wang. Extracting sentiment words using pattern based Bootstrapping method. Computer Engineering and Applications, 2014, 50(1):127-129.

[7] Maas A L, Daly R E, Pham P T, et al. Learning word vectors for sentiment analysis. Meeting of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, 2011:142-150.

[8] Turian J, Ratinov L, Bengio Y. Word representations: a simple and general method for semi-supervised learning. ACL 2010, Proceedings of the, Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden. 2010:780-1.

[9] Mikolov T, Sutskever I, Chen K, et al. Distributed Representations of Words and Phrases and their Compositionality. Advances in Neural Information Processing Systems, 2013, 26:3111-3119.

[10] Bansal M, Gimpel K, Livescu K. Tailoring Continuous Word Representations for Dependency Parsing. Meeting of the Association for Computational Linguistics. 2014:809-815.

[11] Ding Yang, Aimin Yang. Classification approach of Chinese texts sentiment based on semantic lexicon and naive Bayesian. Application Research of Computers, 2010,27(10) : 3739+3743.

[12] Liping Du, Xiaoge Li, Yu Gen, et al. New Word Detection Based on an Improved PMI Algorithm for Enhancing Segmentation System. Acta Scientiarum Naturalium Universitatis Pekinensis, 2016, 52(1):35-40.

[13] word2vec. https://code.google.com/p/word2vec/. unpublished

[14] Chengqing Zong. Statistical Natural Language Processing.Beijing: Tsinghua university press, 2008:395-398.

[15] Mikolov T, Chen K, Corrado G, et al. Efficient Estimation of Word Representations in Vector Space. Computer Science, 2013.

[16] Mikolov T, Sutskever I, Chen K, et al. Distributed Representations of Words and Phrases and their Compositionality. Advances in Neural Information Processing Systems, 2013, 26:3111-3119.

[17] Stelios Karabasakis. Aspect Miner Fine-grained feature level opinion mining from rated review corpora. unpublished

[18] Rong X. word2vec Parameter Learning Explained. Eprint Arxiv, 2014.

[19] Khalid H, Shihab E, Nagappan M, et al. What do mobile app users complain about?. Software, IEEE, 2015, 32(3): 70-77.

[20] Turney PD. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews.Proceedings of the 40th annual meeting on association for computational linguistics. Association for Computational Linguistics, 2002: 417-424.