Sentiment Analysis on Chinese Movie Review with Distributed Keyword Vector Representation

Chun-Han Chu[1], Chen-Ann Wang[2], Yung-Chun Chang[1], Ying-Wei Wu[3], Yu-Lun Hsieh[1], Wen-Lian Hsu[1]
[1]Institute of Information Science, Academia Sinica, Taipei City, Taiwan
[2]Department of Computer Science, National Tsinghua University, Hsinchu City, Taiwan
[3]Department of Computer Science, National Taiwan University, Taipei City, Taiwan
{johannchu, openan77, changyc, morphe, hsu}@iis.sinica.edu.tw, b02902043@ntu.edu.tw

***Abstract*—In the area of national language processing, performing machine learning technique on customer or movie review for sentiment analysis has been frequently tried. While methods such as support vector machine (SVM) were much favored in the 2000s, recently there is a steadily rising percentage of implementation with vector representation and artificial neural network. In this article we present an approach to implement word embedding method to conduct sentiment analysis on movie review from a renowned bulletin board system forum in Taiwan. After performing log-likelihood ratio (LLR) on the corpus and selecting the top 10000 most related keywords as representative vectors for different sentiments, we use these vectors as the sentiment classifier for the testing set. We achieved results that are not only comparable to traditional methods like Naïve Bayes and SVM, but also outperform Latent Dirichlet Allocation, TF-IDF and its variant. It also tops the original LLR with a substantial margin.**

Keywords—sentiment analysis, machine learning, TF-IDF, LLR, word embedding

I. INTRODUCTION

Ever since data mining and machine learning technique were popularized, people applied the use on a wide range of topics, ranging from classification of insect sounds [26] to recognition of pen strokes on a handwritten assignment of students [27]. And thanks to the easier access to Internet and computers, more data is being generated every day for engineers and researchers to analyze. In the area of natural language processing, while some use it to perform tasks such as information extraction [5], some use it for classification of sentiment orientation. Turney [1] arguably pioneered the application of machine learning on classifying the emotion of movie review. Blizter et al. [4] investigated the domain adaptation for sentiment classifiers by using online merchandise reviews that span from books to household appliances. Lu et al. [6] presented an approach for joint bilingual sentiment classification at the sentence level, whereas Chang et al. [21] built semantic frames from Chinese news articles to detect the emotion of readers.

In this article, we present an innovative approach to conduct sentiment analysis on Chinese movie review with the use of distributed keyword vectors, or DKVs. The DKV is derived from word embedding, a word representation scheme provided by Bengio et al. in the 2000s [3]. We provide comparison of DKV against methods such as Naïve Bayes, support vector machine (SVM), log-likelihood ratio (LLR), Latent Dirichlet Allocation (LDA) [18], and Delta TF-IDF, a modified TF-IDF method [7]. The result outperforms the comparison in many aspects.

II. RELATED WORK

This work is under the influence of a rich collection of previous publications. Under the sentiment classification category, Turney [1] used phrases in a sentence that contain adverbs or adjectives to calculate their mutual information with the word "excellent" and "poor" to obtain the indicative semantic orientation. Pang et al. [2] applied Naïve Bayes, maximum entropy, and SVM on movie review to show the difficulties of sentiment classification versus traditional topic-based categorization; two years later Pang and Lee [8] devised the opinion mining approach of limiting the text categorization techniques on only the subjective portions of the document with minimum cut.

Many also made use of lexical database for such task, either with existing source such as WordNet [9] or generate an opinion word list in the experiment procedure. Kamps et al. [10] used the synonymy relations in WordNet to measure the distance between adjectives as a metric of word similarity. Hu and Liu [11] extracted certain adjectives as opinion words to extract the opinion sentences in customer reviews. As a more creative concept, Takamura, Inui, and Okumura [8] even modeled the orientation of words as spins of electrons and used mean field approximation to compute the probability function of the system.

In recent years, a rather popular method for modelling the semantic orientation of documents is using vectors for word representation, where neighboring words that frequently co-occur with a phrase are used to represent the phrase itself. Apart from Bengio et al. [3], the term "word vector" appeared as early as 1998 in [12], where Schütze used cosine to measure the similarity of words represented by vectors generated with surrounding text. Patwardhan and Pedersen [13] built "gloss vectors" upon word vectors with the glossary of WordNet to estimate the semantic relatedness of concepts. Chu et al. [14] constructed gloss vectors with opinion words from [11] to establish vectors representing positive and negative sentiments to classify unknown words. Mikolov et al. [15] proposed two models (CBOW and skip-gram) for vector representation of words in large data sets learned by neural networks with faster

training time, whereas Hsieh et al. [17] utilized such models to create distributed keyword vectors (DKV) to represent documents for topic detection. Chang et al. [22] also used DKV to perform sentiment analysis on Chinese microblog to measure the regional prejudice that potentially exist among a group of internet users that are geographically aggregated toward outsiders.

III. Methodology

To proceed with the task, numerous methods from information retrieval and natural language processing have been researched. For the sake of simplicity, here we neglect the description of k-nearest neighbor (kNN), Naïve Bayes, SVM, and LDA; we mention only Delta TF-IDF, LLR, and DKV, for DKV is used upon LLR in our experiment.

A. Delta TF-IDF

In information retrieval, many tend to use words in a document to represent the topic that the article is conveying, whereas the most intuitive approach would be to count the *term frequency*, or TF, for each word that is presented as the relevance of the word with the topic. Nevertheless, while topic-specific keywords will with a high chance show up in their categories, prepositions and conjunctions would frequently appear in all categories and ranked high in the list. Therefore, using TF alone as a ranking score is not a robust standard for finding keywords. To counter the problem, *inverse document frequency* (IDF) is introduced to offset the terms that appear too frequently in all domains. The IDF is a logarithmic fraction of the number of all documents over the number of documents that contain a specific keyword:

$$\text{TF-IDF score} = tf_{t,d} * \log\frac{N}{df_t} \quad (1)$$

where $tf_{t,d}$ is the term frequency of term t in document d, N the total number of documents, and df_t the documents that contain the term t. If the denominator of that fraction is high, i.e. the documents where a word springs up, the overall logarithmic would be smaller. Hence after multiplying IDF with the original TF, we obtain a smaller score that brings down the "domain distinguishability" of a certain word, obtaining a more objective criterion.

In the work of Martineau and Finin [7], they further defined the *delta TF-IDF* as the score difference of TF-IDF between two sentiment polarities:

$$V_{t,d} = tf_{t,d} * \log\frac{P}{P_t} - tf_{t,d} * \log\frac{N}{N_t} \quad (2)$$

where P and N represent number of documents in the positive and negative category, respectively. P_t is the number of positive labelled data containing the term t, and N_t is the counterpart for negative labelled data. If the training sets are balanced, i.e. P=N, then we can obtain:

$$V_{t,d} = tf_{t,d} * \log\frac{N_t}{P_t} \quad (3)$$

The resulting score $V_{t,d}$ can be used as an alternative to the original TF-IDF.

B. Log-Likelihood Ratio (LLR)

Another suitable method for finding representative keywords in a document is the log-likelihood ratio, or LLR [18]. LLR calculate the likelihood of the assumption that the existence of a word in a document is not random, where a higher value indicates that the keyword highly relates to the category. The LLR value for each word is calculated as follows. Given a corpus as the training set, we define four frequency values:

$$k = N(w \wedge T), \qquad l = N(w \wedge \neg T)$$
$$m = N(\neg w \wedge T), \qquad n = N(\neg w \wedge \neg T)$$

where k is the number of documents that both contain the word w and belong to topic T, l the number of documents that contain w but do not belong to T, etc. The equation for LLR then takes in the four parameters in the following form:

$$LLR(w,T) = 2 * \log\left(\frac{p(w|T)^k\left(1-p(w|T)\right)^m p(w|\neg T)^l\left(1-p(w|\neg T)\right)^n}{p(w)^{k+l}(1-p(w))^{m+n}}\right) \quad (4)$$

$p(w)$, $p(w|T)$, and $p(\neg w|T)$ are estimated with maximum likelihood estimation. After obtaining the LLR score for each word in the corpus, we can rank them according to the value and select the top ones to represent their categories.

C. Distributed Keyword Vectors

After finding top-ranked keywords for different categories, we then use them to create distributed keyword vectors, or DKV [17]. The distributed keyword vector traces its root to the concept of word embedding models, namely continuous bag-of-words model (CBOW) and skip-gram model (SG) [15].

Instead of learning a statistical language model, CBOW sets out to model each word with a dense vector representation. (Fig. 1a) In essence, CBOW attempts to find the candidate w_t that maximizes the logarithmic probability of its existence sandwiched between words $w_{t-c},\ldots,w_{t-1},w_{t+1},\ldots,w_{t+c}$ as follows:

$$\sum_{t=1}^{T} log\hat{P}\left(w_t|w_{t-c}^{t+c}\right) = \frac{e^{v_{\bar{w}_t}\cdot v_{w_t}}}{\sum_{i=1}^{V} e^{v_{\bar{w}_t}\cdot v_{w_i}}} \quad (5)$$

where c is the window size of the training context for w_t and T the length of training corpus. V is the size of the vocabulary, while v_{wi} denotes the vector of the word w at position t. $v_{\bar{w}_t}$ denotes the sum of vectors of the context words of w_t.

The structure of CBOW is similar to a feed-forward neural network without the nonlinear hidden layer, by exclusion of which will alleviate the burden of computation time required. On the other hand, SG takes an opposite approach where it tries to predict surrounding words given the word w_t as the input (Fig. 1b) with the following equation:

$$\sum_{t=1}^{T}\sum_{j=-c,j\neq 0}^{c} log\hat{P}(w_{t+j}|w_t) = \frac{e^{v_{w_{t+j}}\cdot v_{w_t}}}{\sum_{i=1}^{V} e^{v_{w_i}\cdot v_{w_t}}} \quad (6)$$

where $v_{w_{t+j}}$ and v_{w_t} denote the vectors of words at position *t+j* and *t*, respectively.

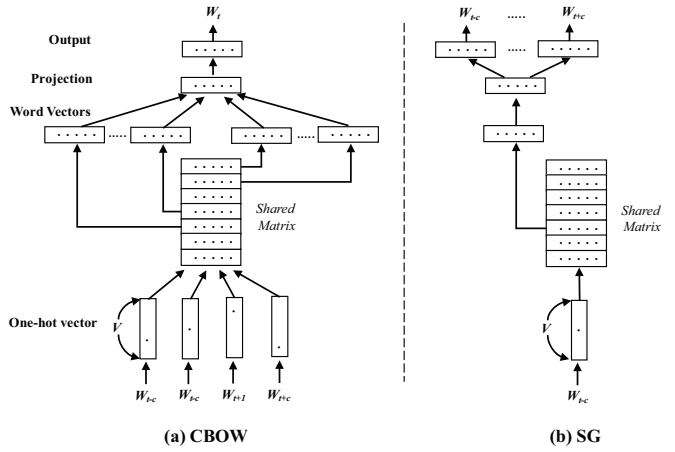

Fig. 1. (a) The CBOW model uses W_{t-c}, …, W_{t+c} as inputs to predict W_t, while in (b) SG uses W_t to predict W_{t-c}, …, W_{t+c} in the ocntext.

These models were used to learn longer text segments [16], such as paragraphs and documents. [16] proposed distributed bag-of-words (DBOW) and distributed memory (DM) models, which are essentially extension of SG and CBOW model, respectively. To construct such document vectors, a unique ID is given to each document where the word vectors among a collection of documents are first learned, followed by running a sliding window in the document from top to bottom to sample every word. In the end, we obtain the embedding for each document.

DKV is derived from the document vector in the preceding description, with the only difference in that *only* the top keywords are used to generate the word vectors, eliminating possible noise induced by irrelevant conjunctions and prepositions. (Fig. 2)

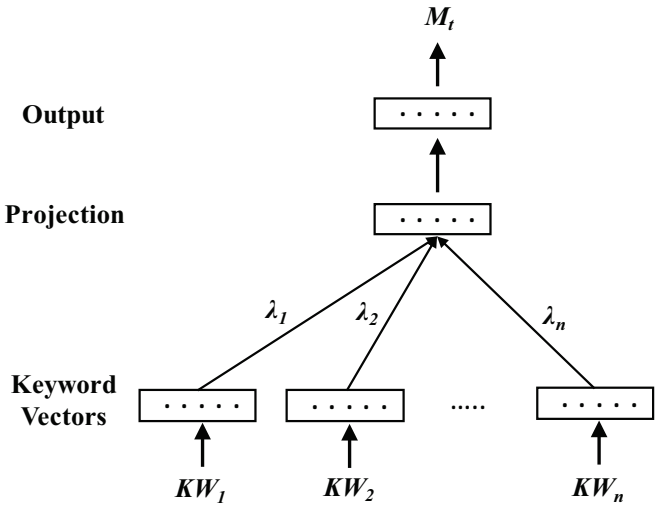

Fig. 2. DKV models the document D_t with keyword vector KW_1, KW_2,……KW_n. λ_1, λ_2,..... λ_n are the LLR score for each keyword.

IV. Experiment

A. Dataset and Procedure

To evaluate the effectiveness of our method, we used python 2.7 with BeautifulSoup module to crawl a total of 4000 Chinese movie reviews (Fig. 4) as our training and testing corpora from PTT, a highly popular BBS forum in Taiwan. In the "movie" discussion board of PTT, most articles are required to provide a category tag in front of the article title ("Good", "Bad", "Really bad", "Disappointingly bad", etc); therefore, they are essentially labelled data suitable for our purpose. Under the current clause of the intellectual property law in Taiwan, using these individual-generated articles for research purpose is allowed as long as proper source are cited. As a result we have stored these reviews on GitHub [20] for anyone who wish to use.

The experiment procedure for our proposed approach is detailed in Fig. 3. We start by splitting the corpus 50/50 into training and testing sets for both positive and negative categories. In each sentiment, 1000 articles are preprocessed with proper segmentation to the unit of sentences. They are then collectively processed through with LLR score to find the top 10000 words; for any term that exists in both categories, it is assigned under the group that produces the higher of the two scores. Afterward, these keywords are used to build the representative vectors for each sentiment, completing the core of the classifier. Finally the testing set is fed into the classifier to test the accuracy, where we use precision, recall, and F-score as metric.

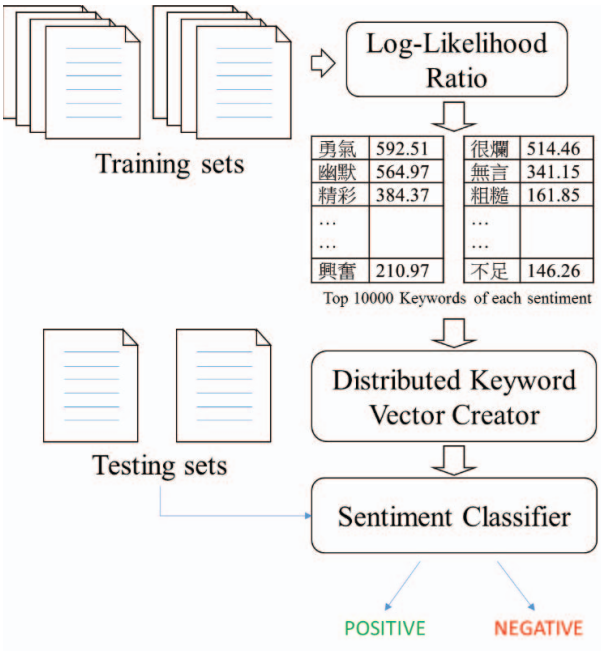

Fig. 3. Experiment procedure

For comparison, we implemented several renowned methods as benchmark. Support vector machine was implemented with libSVM toolkit [23], where the articles are first scanned to find any word that exists in the top-1000 word list (positive and negative) obtained with LLR beforehand; we call these word of

incidence the “hit-words”. These hit-words of a sentiment will be selected out of the word embedding we generated from a Chinese news corpus with the dimension of 300 and summed over with one another after each of them is multiplied by its own LLR value. After averaging the resulting vector with the number of hit-words in that sentiment, we have acquired a 300-dimension sentiment vector. The same procedure can then be replicated for the other sentiment and, after concatenating both, we can send it into libSVM for training.

作者: zate (寧靜的世界)
看板: movie
標題: [失望雷] 期望越高失望越大的蜘蛛人 3
時間: Thu May 10 02:30:21 2007
蜘蛛人 3 看完後有股失落感，我知道是我給予其太大的期望了，想超越第二集的經典並不是這麼容易的事。 我只能說導演山姆雷米的野心太大了，在一集中想講的東西太多，擺明就要當成個人的最後一集拍。許許多多的主軸湊起來，導致影片過長，結構也太冗長鬆散。 不合理的地方我就不說了，這集的份量應該分配個兩集來拍，多許許多多的事情才能描述得更加清楚，而不是大雜匯的什麼都加一點，導致最後很多感覺都沒有到位。動作部分到是沒有太失望，開頭跟哈利的空中追逐，還有最後兩人的聯手，還真是熱血。不過很多動作弄得過快，反而讓人感覺就是十分華麗的打鬥，卻沒有讓人印象非常深刻的動作。

Fig. 4. An example of Chinese movie review from PTT.

TABLE I. PRECISION, RECALL, AND F-1 SCORE OF MOVIE REVIEW

Methods	Positive	Negative	Accuracy
	Precision / Recall /F_1-score (%)		
kNN	37.9 / 41.0 / 39.4	35.8 / 32.8 / 34.2	36.9
LLR	46.7 / 73.1 / 56.9	48.9 / 23.6 / 31.8	48.3
Naïve Bayes	58.3 / 86.8 / 69.7	74.2 / 37.9 / 50.2	62.4
SVM	76.0 / 49.3 / 59.8	62.5 / 84.4 / 71.8	66.9
Delta TFIDF*	63.0 / 97.2 / 76.5	93.8 / 43.0 / 58.9	70.1
LDA-SVM	81.9 / 81.3 / 81.6	81.4 / 82.0 / 81.7	81.7
TFIDF	85.0 / 82.6 / 83.7	83.1 / 85.4 / 84.2	84.0
Delta TFIDF	88.6 / 86.9 / 87.6	87.2 / 88.8 / 87.9	87.9
DKV	89.6 / 87.7 / 88.6	87.9 / 89.8 / 88.8	88.7

Note. Delta TFIDF= Delta TFIDF use LLR value.*

LDA-SVM is implemented by first using LDA toolkit [24] to find top 300 keywords and their LDA values for both sentiment categories among the review articles. Using these 300 words as the dimension of a vector, in each article we again find the hit-words and multiply their occurrences with corresponding LDA values as the value for each “dimension”. The vectors for positive and negative sentiments can be again concatenated and fed into libSVM for classification. As for kNN, we set k as 3 and devise the code ourselves. Naïve Bayes is implemented with a toolkit directly [25].

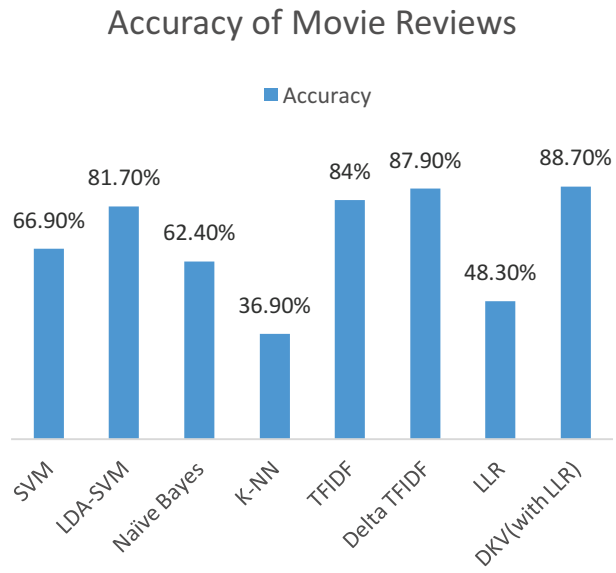

Fig. 5. Comparison of accuracy on classification of movie reviews.

Fig. 6. The word cloud generated from movie review keywords. Colors and their corresponding emotion: Red – negative and Green - positive.

B. *Results and Discussion*

The precision, recall, and F-score of each method for review classification are listed as TABLE I. The accuracy is depicted with bar chart in Fig. 5. DKV achieved the highest accuracy of 88.7%, while Delta TF-IDF came close in second place with 87.9%. The lowest among all methods is 36.9% by kNN. What is worth noting is that implementing DKV on top of the LLR results could obtain remarkable effect (using LLR alone only obtain 48.3%)., while accuracy dropped when Delta TF-IDF piggybacked on LLR instead of working alone. Using LDA to create vectors for SVM surely improved the accuracy as well; still, DKV is proved to be more effective, possibly due to its denser nature of the vectors. To better visualize the keyword

selection method, we present the keywords as word clouds in Fig. 6. Each keyword is color according to its corresponding sentiment, and scaled according to its LLR score. In this way, we can easily identify the features tied to each sentiment. The figure highlights the fact that the extracted sentiment keywords are highly correlated with polarity of movie review, and including them in the distributed vector representations helps DKV achieve more accurate sentiment classification.

推 FSJL: 您解釋前因後果後 我覺得這部真的蠻棒的!推 md1011:感謝你對資料背景提供 很棒的文章　但是這部片好看啦 XD

推 Xyrho: 推好文！不過要把這些全拍出來可能會變三部曲紀錄片吧 XD

推 jerrysu74: 我是版主就給你 M 了.寫的真的很棒.

推 NaStradamus: 全拍出來不是更好睡嗎？XD

推 johnny04a: 推!!!

推 laevil: 推呀，不過 Frank 把那麼多貪污警察婊入獄還沒事還滿誇張的..

推 elmostar: 美國黑幫超好看 我個人覺得情節頗緊湊 不會沉悶 XD

Fig. 7. Some example for comments on movie reviews.

TABLE II. PRECISION, RECALL, AND ACCURACY OF MOVIE REVIEW COMMENTS

Method	Positive	Negative	Accuracy
	Precision / Recall /F_1-score (%)		
LDA-SVM	72.5 \ 18.9 \ 30.0	53.3 \ 92.8 \ 67.7	55.8
SVM	83.5 \ 16.7 \ 27.9	53.2 \ 96.6 \ 68.7	56.3
kNN	59.7 \ 47.3 \ 52.8	56.4 \ 68.1 \ 61.7	57.7
LLR	61.6 \ 73.6 \ 67.0	67.2 \ 54.1 \ 59.9	63.8
Delta TFIDF*	75.3 \ 45.0 \ 56.3	60.8 \ 85.2 \ 70.9	65.1
TFIDF	74.1 \ 67.5 \ 70.6	70.2 \ 76.4 \ 73.1	72.0
Naïve Bayes	70.8 \ 76.5 \ 73.6	74.5 \ 68.5 \ 71.4	72.5
Delta TFIDF	81.7 \ 59.3 \ 68.7	68.1 \ 86.7 \ 76.2	73.0
DKV	77.3 \ 70.3 \ 74.2	74.2 \ 78.4 \ 76.2	75,7

Note. Delta TFIDF= Delta TFIDF use LLR value.*

As an additional test, we also use the extra feature provided at the end of each of these articles. On the discussion board of PTT, the user can provide "like" ("推") or "boo" ("噓") comments to the articles posted by others (Fig. 7). In our assumption, since there are limited number of characters a user can post in these comments each time, the user is often obliged to state their opinions in the most compact sentences, which should provide a cleaner context for vector generation. Consequently, we use the comments in these articles (i.e. the reviews for the review) as the training corpus and conduct the same experiment. The results are shown in Table II and bar charts in Fig. 8.

In the result, we can still see that DKV surpasses other methods; however, its accuracy are not as good as that produced by using the full reviews, contradicting our initial assumption. One of the possible reasons for such phenomenon would be the decrease in diversity among terms used between in the reviews and comments. While the comments might possess a more compact fashion of critical phrases, the spectrum of which the terms used in comments span might not be as wide as that delivered by the corpus of normal reviews.

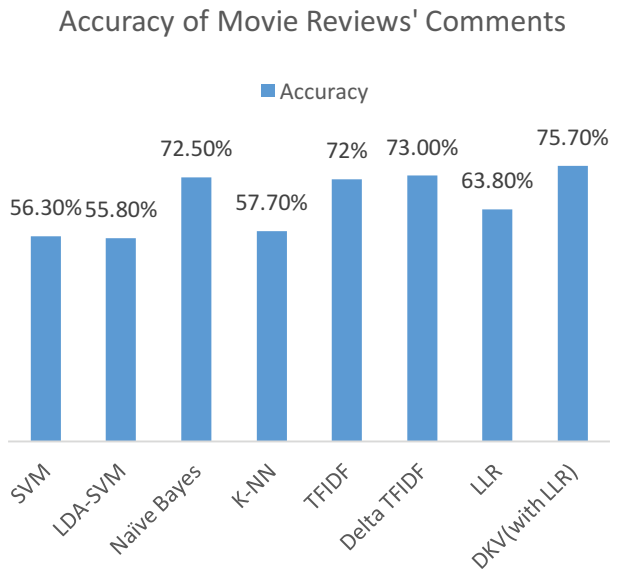

Fig. 8. Comparison of accuracy on classification of movie reviews' comments.

V. CONCLUSION

In this paper we present a modified document embedding method by using keywords extracted from a collection of articles. The approach applied on sentiment analysis task of Chinese movie review is effective; we can achieve results that are easily on par with those obtained from previous methods. It can also surpass the methods like TF-IDF and LLR after using the products of these methods and building representative keyword vectors upon them.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for devoting their time processing through our work and providing insightful comments.

REFERENCES

[1] P.D. Turney, "Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews," Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pp. 417-424, 2002.

[2] B. Pang, L. Lee, and S. Vaithyanathan, “Thumbs up? Sentiment classification using machine learning techniques,” Proceedings of EMNLP, pp. 79-86, 2002.

[3] Y. Bengio, H. Schwenk, J.S. Senécal, F. Morin, and J.L. Gauvain, “Neural Probabilistic Language Models,” Innovations in Machine Learning, vol. 194 of the series Studies in Fuzziness and Soft Computing, pp. 137-186, 2006.

[4] J. Blitzer , M. Dredze , and F. Pereira, “Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification,” In Association of Computational Linguistics, pp. 187-205, 2007.

[5] F. Wu and D.S. Weld, “Open information extraction using Wikipedia,” Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics, pp. 118-127, 2010.

[6] B. Lu, C. Tan, C. Cardie and B.K. Tsou, “Joint bilingual sentiment classification with unlabeled parallel corpora,” Proceedings of the 49th Annual Meeting of the Association of Computational Linguistics, pp. 320-330, 2011.

[7] J. Martineau and T. Finin, “Delta TFIDF: An improved feature space for sentiment analysis,” Proceedings of the Third International ICWSM Conference, 2009.

[8] B. Pang and L. Lee, “A sentimental educatoin: sentiment analysis using subjectivity summarization based on minimum cuts,” Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Article No. 271, 2004.

[9] C. Fellbaum, WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press, 1998.

[10] J. Kamps, M. Marx, R.J. Mokken, and M. Rijke, “Using WordNet to measure semantic orientations of adjectives,” Proceedings of the Fourth International Conference on Language Resources and Evaluation, vol. IV, pp. 1115-1118, 2004.

[11] M. Hu and B. Liu, “Mining and summarizing customer reviews,” Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 168-177, 2004.

[12] H. Schütze, “Automatic word sense disambiguation,” Computational Linguistics - Special issue on word sense disambiguation archive, vol. 24 Issue 1, pp. 97-123. MIT Press Cambridge, MA, USA, March 1998.

[13] S. Patwardhan and T. Pedersen, “Using WordNet-based Context Vectors to Estimate the Semantic Relatedness of Concepts, ” 2006.

[14] C.H. Chu, A.H. Roopa, Y.C. Chang, and W.L. Hsu, “Constructing sentiment sensitive vectors for word polarity classification,” Conference on Technologies and Applications of Artificial Intelligence (TAAI), pp. 252-259, 2015

[15] T. Mikolov, K. Chen, G. Corrado, J. Dean, “Efficient Estimation of Word Representations in Vector Space,” 2013.

[16] Q. Le and T. Mikolov, “Distributed representations of sentences and documents,” in Proceedings of the 31st International Conference on Machine Learning (ICML-14), pp. 1188–1196, 2014.

[17] Y.L. Hsieh, S.H. Liu, Y.C. Chang, and W.L. Hsu, “Distributed keyword vector representation for document categorization,” Conference on Technologies and Applications of Artificial Intelligence (TAAI), pp. 245-251, 2015

[18] D.M. Blei, A.Y. Ng, and M.I. Jordan, “Latent dirichlet allocation,” Journal of Machine Learning Research, vol. 3, 2003.

[19] C. D. Manning and H. Schütze, Foundations of statistical natural language processing. MIT press, 1999.

[20] https://github.com/johannchu/PTT_movie_review

[21] Y.C. Chang, C.C. Chen, and W.L. Hsu, "Semantic Frame-based Approach for Reader-Emotion Detection," In Proceedings of The 19th Pacific Asia Conference on Information Systems, AIS Electronic Library (AISeL), July 2015.

[22] Y.C. Chang, C. Chou, Y. Zhang, X. Wang, and W.L. Hsu, “Sentiment Analysis of Chinese Microblog Message Using Neural Network-based Vector Representation for Measuring Regional Prejudice,” In Proceedings of 20th Pacific Asia Conference on Information Systems, Paper 384, 2016.

[23] C.C. Chang and C.J. Lin., “LIBSVM : a library for support vector machines,” ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011.

[24] https://github.com/a55509432/python-LDA

[25] https://github.com/muatik/naive-bayes-classifier/

[26] Y. Hao, B. Campana, and E. Keogh, “Monitoring and mining insect sounds in visual space,” SIAM International Conference on Data Mining, 2012.

[27] T.F. Stahovich and H. Lin, “Enabling data mining on handwritten homework,” Computers & Graphics, vol. 57, pp. 31-45, June 2016.