Distributed Text Representation with Weighting Scheme Guidance for Sentiment Analysis

Zhe Zhao, Tao Liu(✉), Xiaoyun Hou, Bofang Li, and Xiaoyong Du

School of Information, Renmin University of China, Beijing, China
{helloworld,tliu,xiaoyunhou,libofang,duyong}@ruc.edu.cn

Abstract. With rapid growth of social media, sentiment analysis has recently attracted growing attention in both academic and industrial fields. One of the most successful paradigms for sentiment analysis is to feed bag-of-words (BOW) features into classifiers. Usually, weighting schemes are required to weight raw BOW features to obtain better accuracy, where important words are assigned more weights while unimportant ones are given less weights. Another line of researches for sentiment analysis focuses on neural models, where dense features are automatically extracted from texts by neural networks. In this paper, we take advantages of techniques in both lines of researches, where weighting schemes are introduced into the neural models to guide neural networks to focus on those important words. Neural models are known for their automatic feature learning abilities, however, we discover that when suitable guidance such as weighting schemes are applied, better features can be extracted for sentiment analysis. Experimental results show that our models outperform or can compete with state-of-the-art approaches on three commonly used sentiment analysis datasets.

Keywords: Distributed representation · Sentiment analysis · Weighting scheme

1 Introduction

Sentiment analysis aims at extracting users' subjective information from raw texts. It is valuable for companies and institutions to turn user-generated texts into knowledge for decision support. One of the most successful paradigm for sentiment analysis is to feed bag-of-words (BOW) features with weighting schemes into classifiers such as support vector machines [16]. In BOW, each word represents one dimension of features, and weighting schemes give those important words more weights [11]. Though simple, strong baselines are achieved on a range of sentiment analysis datasets.

However, traditional text features suffer from high dimensionality and data sparsity. They take each word as an atomic unit, which totally ignores the internal semantics of the words. Recently, many researchers have turned their attention to extracting dense text representation by neural networks (NNs) [5]. Typical neural networks include Convolutional NNs [7], Recursive NNs [17],

© Springer International Publishing Switzerland 2016

Paragraph Vector (PV) [9], etc. Usually, these models learn dense text representations upon pre-trained dense word representations. One advantage of these models is that they can extract features automatically. In this sense, we can directly feed raw data into neural networks with no requirements of prior knowledge. However, a natural question is raised here: if prior knowledge is available, can we utilize the knowledge to help the training of the neural networks to achieve better results? This is the motivation of our models.

In this paper, a novel model is proposed which takes advantages of both neural models and traditional weighting schemes. Weighting schemes tell us which words are important and which words are not. Taking movie reviews for example, words like 'wonderful' and 'worst' reflect reviewers' sentiment tendencies, while words like 'of' and 'watch' contain little information about users' attitudes to movies. A lot of work has proven the effectiveness of weighting schemes on raw BOW features [2,8,11,15,19]. Inspired by their successes in BOW, we discover that weighting techniques are also very useful for guiding the training of neural networks. By knowing importance of words in advance, we can train neural models to focus on extracting features from words that is critical to the meanings of texts, while ignore those unimportant words. Since previous weighting schemes are originally applied to BOW features, we propose a novel weighting framework specially designed for neural models. As a result, very competitive results are achieved in both sentence-level and document-level datasets.

Section 2 reviews the related work of sentiment analysis. Section 3 discusses how to integrate weighting techniques into neural models. Section 4 proposes a novel weighting framework specifically designed for neural models. Experimental results are given in Sect. 5, followed by the conclusion in Sect. 6. Source code of the paper will be available at http://github.com/zhezhaoa/Weighted-Paragraph-Vector. Other researchers can reproduce our results easily and further improve models upon our new baselines.

2 Related Work

Bag-of-words Features. Text representation is of vital importance for sentiment analysis. One of the most popular features is bag-of-words (BOW). Early work that uses BOW feature on sentiment analysis is done by [16]. They found that binary weighted feature with SVM classifier gives promising results. Following their work, many researchers focus on applying various weighting schemes to original BOW text representations. One of the most successful attemptions is to introduce information of class preference into weighting schemes. [11] calculate the document frequency (DF) of words in positively and negatively labeled texts respectively, and use the ratios of them to weight words. [19] apply class preference weighting on bag-of-ngrams features and achieve very strong baselines on a range of sentiment analysis and text classification tasks. [8] add credibility weighting to determine which words are important for sentiment analysis. [2,15] give detailed discussions over different weighting schemes on BOW features.

Deep Neural Networks. Recently, neural networks (NN) have become increasingly popular in natural language processing (NLP). They can generate dense word and text representations and achieve state-of-the-art results on a range of tasks. One of the most fundamental work in this field is word embedding, where dense word representations are learned by modeling the relationships between target words and their contexts [1]. Almost all neural models for NLP are built upon word embeddings since neural networks require dense representation as input to make models computational. [7] propose to use Convolutional NN (CNN) for sentiment analysis, which can effectively capture n-gram features and achieve state-of-the-art results on many sentence-level datasets. Recursive NN (RecNN), proposed by [17], constructs neural networks on the basis of parse tree and is able to extract fine-grained information of sentences. Another family of neural networks for sentiment analysis is Recurrent NN (RNN). The hidden layers of the RNN store all previous information in theory. The hidden layer of the last word can be used as the representation of the whole texts. Most recently, combinations of neural networks are proposed to capture complex structures of the texts. Their training processes can be categorized in two steps. The first step is to extract sentence features from word features (word embeddings), and the second step is to extract document features from sentence features [3,10]. As a result, these models can not only capture word order and syntactic information, but also take relationships among sentences into considerations. The drawbacks of these deep neural networks are also obvious. These models are expensive in computational resources. Besides that, they are not as robust as traditional BOW approaches. The effectiveness of these models closely relies on careful hyper-parameters tuning and some sub-tasks such as pre-trained word embedding, parsing and sentence segmentation.

Neural Bag-of-words Models. Though deep neural models are very powerful in theory, only limited improvements are achieved in sentiment analysis when compared with traditional BOW features with weighting schemes. It seems that information like word order, sentence and document structure is not very crucial for sentiment analysis. Another line of neural models is neural bag-of-words models. Instead of constructing complex composition upon word embeddings, these models basically ignore order and syntactic information. Representative neural bag-of-words models include Deep Average Networks (DAN) [6] and Paragraph Vector (PV) [9]. DAN at first takes average of the words embeddings as the inputs and then constructs multiple neural layers upon them. PV embeds text by making it useful to predict the words it includes. These models enjoy the advantages of being simple and robust compared with other deep neural networks, and can still achieve competitive results on sentiment analysis tasks.

Existing neural bag-of-words models treat each word equally. To capture better features for sentiment analysis, in this paper we use weighting schemes to guide the training of Paragraph Vector models, thus the new models can pay more attention to words that reflect the polarities of texts. When suitable weighting schemes are used, significant improvements over traditional Paragraph Vector model are witnessed.

3 Weighted Paragraph Vector (WPV)

In this study, we introduce weighting schemes into a neural bag-of-words model, Paragraph Vector (PV). PV is just the direct extension of word2vec. Word2vec is a very popular word embedding training toolkit[1]. It achieves state-of-the-art results on a range of linguistic datasets, and at the same time only requires a fraction of time compared with previous word embedding models [13,14]. The main difference between PV and word2vec is that PV treats each text as a special word and uses this special word to predict the words in the text. PV includes two variants, PV-DBOW and PV-DM, which corresponds to two variants in word2vec, Skip-gram (SG) and continuous bag-of-words (CBOW). In the following subsections, we introduce weighting techniques into PV-DBOW and PV-DM respectively.

3.1 Weighted PV-DBOW

The original objective function of PV-DBOW is as follows:

$$\sum_{i=1}^{|T|}\sum_{j=1}^{|t_i|} logP(w_{ij}|t_i) + \sum_{i=1}^{|WN|}\sum_{-c\leq j\leq c, j\neq 0} logP(w_i|w_{i+j}) \tag{1}$$

$$where \quad P(a|b) = \sigma(\, e(a) \bullet e(b)\,) \prod_{k=1}^{K} E_{w_k \sim P_n(w)}(\sigma(\, -e(b) \bullet e(w_k)\,))$$

where t_i=$\{w_{i1}, w_{i2}, ..., w_{i|t_i|}\}$ denotes i_{th} text and T=$\{t_1, t_2, ..., t_{|T|}\}$ denotes the whole dataset. $|WN|$ is the number of training words in the whole dataset. Negative sampling [14] is used to define the 'conditional probabilities' of target word given its context: $e(\cdot)$ denotes the embedding of word (text) $\cdot$. $\bullet$ denotes inner product. σ is sigmoid function and $P_n(w)$ is uni-gram distribution raised to the n-th power. The first part of the objective uses text that contains the target word as context. The second part is just the objective of ordinary word embedding model, where words in local window are used as context. Traditional Paragraph Vector model treats each word in the text equally. In this sense, it can be viewed as the counterpart of BOW features where each feature is represented by the count of the word in the text.

Obviously, some words are more important for discriminating polarities of the sentiment. The main idea of our model is to make embedding of text pay more attention to those discriminative words, instead of neutral words which have little value for determining polarities of texts. We give each word a weight (a real value), which represents how important a word is for sentiment analysis. How to assign the weights to words will be discussed in Sect. 4. The objective of weighted PV-DBOW is as follows:

$$\sum_{i=1}^{|T|}\sum_{j=1}^{|t_i|} Weight(w_{ij}) logP(w_{ij}|t_i) + \sum_{i=1}^{|WN|}\sum_{-c\leq j\leq c, j\neq 0} logP(w_i|w_{i+j}) \tag{2}$$

[1] http://code.google.com/p/word2vec.

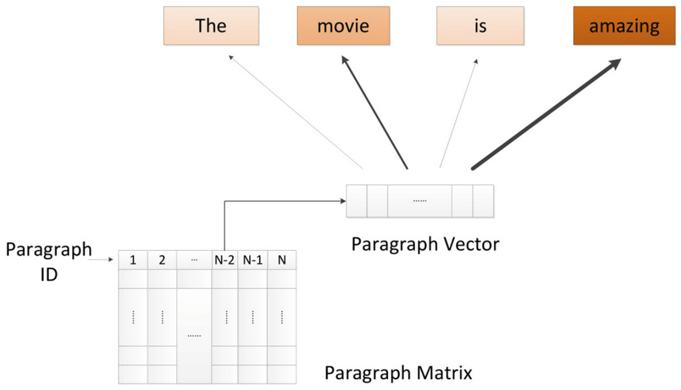

Fig. 1. Illustration of weighted PV-DBOW

where *Weight(w)* reflects the importance of word w. By optimizing the above weighted objective, the trained text embedding is able to predict important word in larger probabilities while ignore those unimportant words. Figure 1 illustrates the framework of weighted PV-DBOW.

3.2 Weighted PV-DM

PV-DM uses average of the text embedding and word embeddings in local window to predict target word. The original objective of PV-DM can be written as follows:

$$\sum_{i=1}^{|WN|} logP(w_i|w_i^{context}) \tag{3}$$

$$where \quad w_i^{context} = e(t_*) + \sum_{-c \leq j \leq c, j \neq 0} e(w_{i+j}) \qquad and \quad w_i \in t_*$$

Like the way we introduce weighting into PV-DBOW. The objective of weighted PV-DM is as follows:

$$\sum_{i=1}^{|WN|} Weight(w_i) logP(w_i|w_i^{context}) \tag{4}$$

An alternative for introducing weighting information into PV-DM is to change the representation of target words' contexts. Since text embeddings should pay more attention to important words, we give text embeddings more weights in constructing contexts when target words are important. In this way, the text embeddings are more affected by those important words while less affected by those unimportant words. The objective function can be written as follows:

$$\sum_{i=1}^{|WN|} logP(w_i|w_i^{context}) \tag{5}$$

$$where \quad w_i^{context} = Weight(w_i)e(t_*) + \sum_{-c \leq j \leq c, j \neq 0} e(w_{i+j}) \qquad and \quad w_i \in t_*$$

We find the latter one performs slightly better in practice. In the rest of the paper, we use weighted PV-DM to denote the latter model.

One may connect our models to the attention based neural models, where attention is also regarded as weighting mechanism. The weights in attention models are obtained during the training process, which comes at a cost [10]. However, the weights in this paper are calculated before the training process. We will discuss how to obtain weights for words in details in the next section.

4 Weighting Schemes

For several decades, intensive researches have been done on designing various weighting schemes on raw BOW features. Most of weighting schemes can not be directly applied to neural case, but the intuition behind them are very valuable, which guides us to design a new weighting framework for Paragraph Vector. As mentioned in above section, traditional PV treats each word equally and can be viewed as the neural counterparts of raw BOW feature. We use PV as baselines and gradually add new weighting techniques upon it.

4.1 IDF Weighting

Direct improvement over BOW baseline is inverse document frequency (IDF) weighting, where words that appear in every texts are given very low weights. Document frequency of word w is denoted by *df(w)*. A substitution for IDF weighting is sub-sampling technique used in [14], where high-frequency words are removed randomly. These two techniques perform almost the same in practice. For sake of space saving, in the following subsection we add new weighting techniques on the basis of IDF weighting. The weighting function of IDF is as follows:

$$Weight_1(w) = log(\frac{|T| + 1}{df(w)}) \tag{6}$$

4.2 Weighting with Class Preference

IDF and sub-sampling weighting do not take class preference into consideration. Intuitively, a word is important if it has uneven distribution over classes. In this paper, we only consider binary sentiment analysis. We use *Pos(w)* and *Neg(w)* to respectively denote the number of positively and negatively labeled texts that contain word *w*. [11,19] show that weighting word w with *log(Pos(w)/Neg(w))* is very effective for BOW features. We adapt this term weighting function for neural case:

$$Weight_2(w) = \begin{cases} (Pos(w)/Neg(w))^k & Pos(w) > Neg(w) \\ (Neg(w)/Pos(w))^k & Pos(w) \leq Neg(w) \end{cases} \tag{7}$$

where k is a hyper-parameter and determined by validation set. Empirically, 0.5 is a good choice for k.

4.3 Credibility

The last block we add on our weighting framework is credibility. In fact, credibility is a supplementary of class preference weighting. The occurrence of a word in positive and negative texts can be modeled by binomial distribution. If a word occurs twice in positive texts and once in negative texts, we can not reject hypothesis that word has even distribution with reasonable significance levels. If a word occurs 200 times and 100 times in positive and negative texts respectively, we can conclude that the word has uneven distribution over classes with greater credibility, even though the ratio is the same with the previous case. Therefore the number of times a word occurs in the dataset is also an important factor for us to determine if a word is 'important' or not. The weighting for credibility used in this paper is as follows:

$$Weight_3(w) = log(log(Count(w) + 1) + 1) \tag{8}$$

where *Count(w)* is used to denote the number of times word w occurs in the dataset.

Finally, we combine three components together and obtain the entire weighting framework for Paragraph Vector models.

$$\begin{aligned} Weight(w) &= Weight_1(w)Weight_2(w)Weight_3(w) \\ &= (Pos(w)/Neg(w))^{k*\lfloor Pos(w)>Neg(w)\rfloor} \\ &\quad *log(log(Count(w)+1)+1)*log(\tfrac{|T|+1}{df(w)}) \end{aligned} \tag{9}$$

$$where \quad \lfloor\cdot\rfloor = \begin{cases} 1 & \cdot\, is\, True \\ -1 & \cdot\, is\, False \end{cases}$$

5 Experiments

5.1 Experimental Setup

Models in this paper are trained by stochastic gradient descent (SGD). Text embeddings obtained by models are regarded as texts features and are fed into logistic regression classifier [4]. Pre-processing of datasets and hyper-parameters setting follow the implementation by [12]. 10 percent of training data is selected as validation data to determine the number of training epochs and hyper-parameters in weighting schemes. The above settings are applied to all datasets unless otherwise noted. Experiments are conducted on two document-level datasets, IMDB[2] and RT-2k[3], and one sentence-level dataset, RT-s[4]. Detailed statistics of these datasets are shown in Table 1.

[2] http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz.
[3] http://www.cs.cornell.edu/people/pabo/movie-review-data/review_polarity.tar.gz.
[4] http://www.cs.cornell.edu/people/pabo/movie-review-data/rtpolaritydata.tar.gz.

Table 1. Detailed statistics of datasets. CV denotes that dataset is evaluated by cross validation. l denotes the average length of texts and |V| denotes the vocabulary size.

Dataset	Train(+,−)	Test(+,−)	l	|V|
IMDB	12500,12500	12500,12500	231	392K
RT-2k	1000,1000	CV	787	51K
RT-s	5331,5331	CV	21	21K

5.2 Effectiveness of Weighting Schemes

We gradually add weighting techniques upon Paragraph Vector baselines and observe the effectiveness of each weighting function over classification accuracies. From Tables 2, 3 and 4, we can observe that weighting schemes, especially weighting with class preference, are very effective for sentiment analysis. Since RT-2k and RT-s only contain limited texts, $Weight_2$ and $Weight_3$ are calculated by statistics of IMDB dataset. Adding additional texts from IMDB dataset to these two datsets is also helpful. We also conduct 'oracle' experiments where $Weight_2$ and $Weight_3$ are calculated by all data (including test data). Though knowing basic statistics of test data in advance is not realistic in many cases, 'oracle' experiments further demonstrate the effectiveness of class preference weighting in Paragraph Vector models.

When all weighting techniques are added, very competitive results are achieved on all three datasets. Though PV models are known for being able to extract features from texts automatically, weighting schemes are still very useful for these models.

Figures 2 and 3 illustrate the accuracies of PV-DM and Weighted PV-DM on IMDB dataset at different embedding dimensions and iterations level. We can observe that WPV achieves decent accuracies even when dimensions of text is very small, while the accuracies of PV declines sharply as we reduce the

Table 2. Illustration of effectiveness of weighting schemes on IMDB. Weighting techniques are gradually applied to PV models and competitive results are achieved finally.

Features	PV-DBOW	PV-DM
Baselines	87.6	88.5
+IDF	88.5(+0.9)	88.9(+0.4)
+Class Preference	89.5**(+1.0)**	90.0**(+1.1)**
+Credibility	90.0(+0.5)	90.2(+0.2)
+Oracle	91.4	91.5

Table 3. Illustration of effectiveness of weighting schemes on RT-2k. PV baselines are weak in this dataset. However, class preference weighting give rise to very significant improvements

Features	PV-DBOW	PV-DM
Baselines	84.9	86.0
+IDF	85.5(+0.6)	86.4(+0.4)
+Class Preference	87.9**(+2.2)**	88.3**(+1.9)**
+Credibility	88.5(+0.6)	88.8(+0.5)
+Additional texts	89.7(+1.2),	89.6(+0.8)
+Oracle	90.4	90.5

Table 4. Illustration of effectiveness of weighting schemes on RT-s. The corpus size of RT-s is very limited, additional text is necessary for this dataset.

Features	PV-DBOW	PV-DM
Baselines	71.5	70.1
+Additional texts	76.8	76.5
+Class Preference	77.8**(+1.0)**	77.7**(+1.2)**
+Credibility	78.4(+0.6)	78.3(+0.6)
+Oracle	79.4	79.2

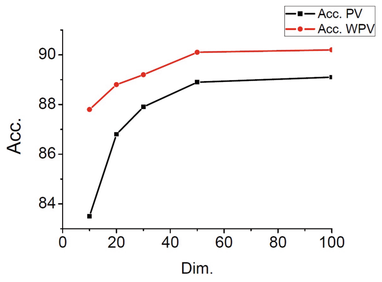

Fig. 2. Accuracies of PV-DM and Weighted PV-DM at different dimensions levels. Iter. is set to 20

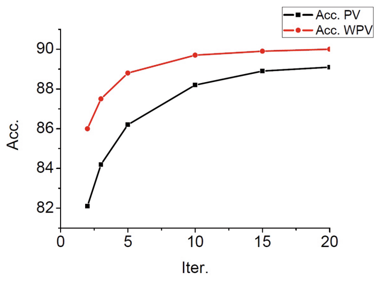

Fig. 3. Accuracies of PV-DM and Weighted PV-DM at different iterations levels. Dim. is set to 50

text features dimensions. WPV also outperforms PV significantly at different iterations levels, and achieves optimal accuracy with less training epochs.

5.3 Comparison of State-of-the-art Models

IMDB is one of the most popular sentiment analysis dataset and has been studied by a large amount of researches. State-of-the-art results on IMDB are obtained by NBSVM and various neural networks. Different state-of-the-art models are compared in Table 5. From IMDB column we can observe that even though our models are essentially bag-of-words models, the results can compete with or even outperform the models that exploit complex information of texts. Since our models basically ignore word order and syntactic information, they are much faster in training time and require less computational resources compared with other state-of-the-art approaches. When our model is combined with NBSVM (Ensemble), state-of-the-art results are achieved on IMDB dataset.

The previous state-of-the-art models on RT-2k is NBSVM. Rare work reports the effectiveness of neural network on this dataset. We speculate the reasons are that the size of RT-2k is limited and the average length is too long. Original Paragraph Vector models only obtain weak baselines on this dataset. However, weighting schemes are very effective on this dataset and we can observe that new state-of-the-art results are achieved by our models.

Table 5. Comparison of models on three commonly used datasets. PV is implemented by source code[a] provided by [12]. Results of Ensemble is obtained by linear combination of WPV and NBSVM [12]. Results in rows 4–7 are from [19]. DCNN uses convolutional-convolutional neural network to extract text features upon word embeddings [3]. Results in row 9–11 are from [10], where recursive-recurrent neural network is used for text representations learning. RecNN: [18]. CNN: [7].

Models	IMDB	RT-2k	RT-s
PV	88.7	86.0	76.5
WPV	90.5	89.6	78.3
Ensemble	**92.6**	**90.5**	79.8
SVM-uni	87.0	86.3	76.2
SVM-bi	89.2	87.4	77.7
NBSVM-uni	88.3	87.8	78.1
NBSVM-bi	91.2	89.5	79.4
DCNN	89.4	-	-
RecNN-RNN	87.0	-	-
WNN	90.2	-	-
BENN	91.0	-	-
RecNN	-	-	77.7
CNN	-	-	**81.5**

[a] http://github.com/mesnilgr/iclr15

RT-s is a sentence-level dataset. Since the corpus size of RT-s is too limited, it requires additional data to achieve decent accuracy. The best result on RT-s is achieved by Convolutional neural network, which performs slightly better than our models. It is probably because that order information is considered in CNN. However, our models require only a small fraction of training time compared with CNN.

6 Conclusion

In this paper, we present a method for introducing weighting schemes into Paragraph Vector models. Weighting schemes tell which words are important, and can be easily used to guide the training of the neural models in our new method. We also propose a novel weighting framework specially designed for neural case. Three weighting techniques are gradually added to our weighting framework to better determine the importance of words for sentiment analysis. From Experimental results, we obtain following conclusions: (1) Though neural models are known for their ability to extract features automatically, prior knowledge like weighting schemes still show obvious effectiveness on neural models, just like weighting schemes work for raw BOW features. (2) Features extracted by our new method are very discriminative. We can obtain high-quality text features with less embedding dimensions and training epochs compared to traditional PV. Even 10-dimensional text feature can achieve good results. (3) Even though our models are essentially bag-of-words models, they can still rival the models that exploit complex compositionality over words and phrases. Very competitive results are achieved by our models on two document-level and one sentence-level sentiment analysis datasets.

Acknowledgements. This work is supported by National Natural Science Foundation of China (61472428, 61003204), the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China No. 14XNLQ06 and Tencent company.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. J. Mach. Learn. Res. **3**, 1137–1155 (2003)
2. Deng, Z., Luo, K., Yu, H.: A study of supervised term weighting scheme for sentiment analysis. Expert Syst. Appl. **41**(7), 3506–3513 (2014)
3. Denil, M., Demiraj, A., Kalchbrenner, N., Blunsom, P., de Freitas, N.: Modelling, visualising and summarising documents with a single convolutional neural network (2014). abs/1406.3830
4. Fan, R., Chang, K., Hsieh, C., Wang, X., Lin, C.: LIBLINEAR: a library for large linear classification. J. Mach. Learn. Res. **9**, 1871–1874 (2008)
5. Goldberg, Y.: A primer on neural network models for natural language processing (2015). CoRR abs/1510.00726
6. Iyyer, M., Manjunatha, V., Boyd-Graber, J., Daumé III, H.: Deep unordered composition rivals syntactic methods for text classification. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics: Long Papers, ACL 2015, vol. 1, pp. 1681–1691 (2015)
7. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, pp. 1746–1751 (2014). A meeting of SIGDAT, a Special Interest Group of the ACL
8. Kim, Y., Zhang, O.: Credibility adjusted term frequency: a supervised term weighting scheme for sentiment analysis and text classification (2014). CoRR abs/1405.3518
9. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings of the 31th International Conference on Machine Learning, ICML 2014, pp. 1188–1196 (2014)
10. Li, J.: Feature weight tuning for recursive neural networks (2014). abs/1412.3714
11. Martineau, J., Finin, T.: Delta TFIDF: an improved feature space for sentiment analysis. In: Proceedings of the Third International Conference on Weblogs and Social Media, ICWSM 2009 (2009)
12. Mesnil, G., Mikolov, T., Ranzato, M., Bengio, Y.: Ensemble of generative and discriminative techniques for sentiment analysis of movie reviews (2014). abs/1412.5335
13. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space (2013). CoRR abs/1301.3781

14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems 26, pp. 3111–3119 (2013). 27th Annual Conference on Neural Information Processing Systems 2013
15. Paltoglou, G., Thelwall, M.: A study of information retrieval weighting schemes for sentiment analysis. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL 2010, pp. 1386–1395 (2010)
16. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using machine learning techniques (2002). cs.CL/0205070
17. Socher, R., Huval, B., Manning, C.D., Ng, A.Y.: Semantic compositionality through recursive matrix-vector spaces. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, pp. 1201–1211 (2012)
18. Socher, R., Pennington, J., Huang, E.H., Ng, A.Y., Manning, C.D.: Semi-supervised recursive autoencoders for predicting sentiment distributions. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, pp. 151–161 (2011). A meeting of SIGDAT, a Special Interest Group of the ACL
19. Wang, S.I., Manning, C.D.: Baselines and bigrams: Simple, good sentiment and topic classification. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers, vol. 2, pp. 90–94 (2012)