LEARNING DOCUMENT EMBEDDINGS BY PREDICTING N-GRAMS FOR SENTIMENT CLASSIFICATION OF LONG MOVIE REVIEWS

Bofang Li , Tao Liu & Xiaoyong Du
School of Information
Renmin University of China
Beijing, P.R. China
{libofang, tliu, duyong}@ruc.edu.cn

Deyuan Zhang
School of Computer
Shenyang Aerospace University
Shenyang, Liaoning, P.R. China
dyzhang@sau.edu.cn

Zhe Zhao
Department of Computer Science
Renmin University of China
helloworld@ruc.edu.cn

ABSTRACT

Despite the loss of semantic information, bag-of-ngram based methods still achieve state-of-the-art results for tasks such as sentiment classification of long movie reviews. Many document embeddings methods have been proposed to capture semantics, but they still can't outperform bag-of-ngram based methods on this task. In this paper, we modify the architecture of the recently proposed Paragraph Vector, allowing it to learn document vectors by predicting not only words, but n-gram features as well. Our model is able to capture both semantics and word order in documents while keeping the expressive power of learned vectors. Experimental results on IMDB movie review dataset shows that our model outperforms previous deep learning models and bag-of-ngram based models due to the above advantages. More robust results are also obtained when our model is combined with other models. The source code of our model will be also published together with this paper.

1 INTRODUCTION

Sentiment analysis is one of the most useful and well-studied task in natural language processing. For example, the aim of movie review sentiment analysis is to determine the sentiment polarity of a review that an audience posted, which can be used in applications such as automatically movie rating. This type of sentiment analysis can often be considered as a classification task. Normally, training and test documents are first represented as vectors. A classifier is trained using training document vectors and their sentiment labels. Test document labels can be predicted using test document vectors and this classifier.

The quality of document vectors will directly affect the performance of sentiment analysis tasks. Bag-of-words or bag-of-ngram based methods have been widely used to represent documents. However, in these methods, each word or n-gram is taken as a unique symbol, which is different to other words or n-grams absolutely, and semantic information is lost.

For modeling semantics of words, word embeddings (Williams & Hinton, 1986; Bengio et al., 2003) is proposed, which has been successfully applied to many tasks such as chunking, tagging (Collobert & Weston, 2008; Collobert et al., 2011), parsing (Socher et al., 2011) and speech recognition (Schwenk, 2007). Following the success of word embeddings, sentence and document embeddings have been proposed for sentiment analysis. For sentence level sentiment analysis, models like recurrent neural network (Socher et al., 2013), convolutional neural network (Kalchbrenner et al., 2014; Kim, 2014), and skip thought vectors (Kiros et al., 2015) all achieved state-of-the-art results. But for document level sentiment analysis, different document embeddings models like convolutional neural network, weighted concatenation of word vectors(Maas et al., 2011), recurrent neural network (Mikolov, 2012), deep Boltzmann machine (Srivastava et al., 2013), and deep averaging network (Iyyer et al., 2015) still can't outperform bag-of-ngram based models such as NBSVM (Wang & Manning, 2012). Thus, more powerful document embeddings learning methods are needed for sentiment analysis.

Recently, Le & Mikolov (2014) proposed a model of learning distributed representation for both sentences and documents, named as Paragraph Vector (PV). PV represents pieces of texts as compact low dimension continuous-value vectors. The process of learning PV is shown in Figure 1-b, which is similar with the typical word embeddings learning methods such as CBOW (Mikolov et al., 2013) shown in Figure 1-a. PV basically treat each document as a special word and learn both document vectors and word vectors simultaneously by predicting the target word.

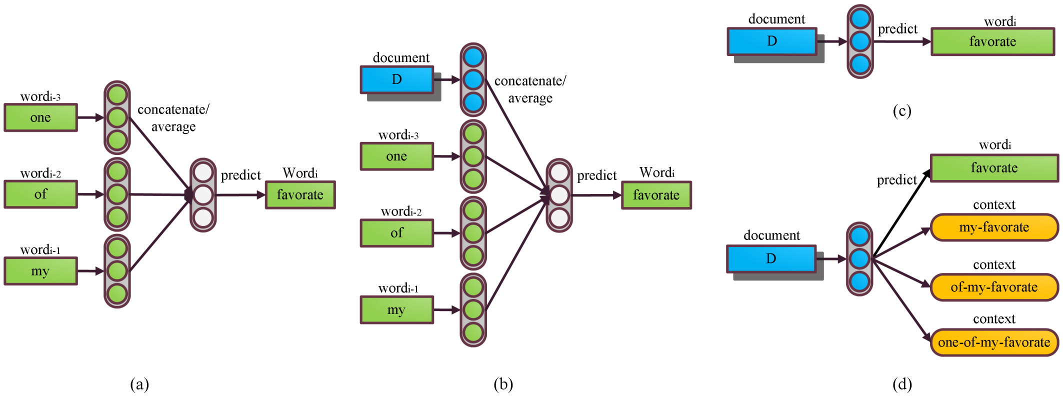

Figure 1: (a) CBOW. (b) PV. (c) simplified version of PV. (d) DV-ngram.

Vectors learned by PV are not sufficient for modeling documents. For example, when the learned information of word vectors of "one", "of", "my" is already sufficient for predicting the next word "favorite" (when the model in Figure 1-a is able to perform the prediction well enough), the document vector can't be sufficiently learned by the model of Figure 1-b. That is, the document vector predicts the word with the help of context, so it do not have to contains all the information. The expressive power of document vectors may be lost in this condition.

Due to this reason, we discover that a simplified version of PV shown in Figure 1-c is more effective for learning document vectors than PV in Figure 1-b [1]. This simplified version of PV learns document vectors alone by predicting its belonging words, thus all the information can only be learned by document vectors to keep the expressive power. But this simplified version of PV does not take contextual words into consideration and thus word order information is lost. [2]

In order to preserve the word order information, our model learns document vectors by predicting not only its belonging words, but n-gram features as well, as shown in Figure 1-d. Note that PV in figure 1-b may not be able to use n-gram features since there are no n-grams that can be specified given certain context. Similar to Paragraph Vector, we name our model as Document Vector by predicting ngrams (DV-ngram). More powerful document vectors can be learned using this model.

2 MODEL

2.1 BASIC MODEL FOR MODELING SEMANTICS

Traditional bag-of-words methods use one-hot representation for documents. Each word is taken as a unique symbol and is different to other words absolutely. This representation often ignores the

[1] In contrast to our experimental results, Le & Mikolov (2014) reported that the simplified PV (referred to as PV-DBOW in their paper) is consistently worse than PV (referred to as PV-DM). But as pointed out by Mesnil et al. (2014), results reported by Le & Mikolov (2014) can only be reproduced when the data is not shuffled, which are considered invalid.

[2] As shown in our experiment section, simplified version of PV (DV-uni) outperforms PV 0.87 percent in terms of accuracy on IMDB dataset.

Table 1: Illustration of documents for comparing document distance

D_1	I saw Captain American yesterday with my friends, its awesome.
D_2	I saw Captain American yesterday with my friends, its inspiring.
D_3	I saw Captain American yesterday with my friends, its meaningless.
D_4	I saw Captain American yesterday with my friends, its awesome and inspiring.

impact of similar words to documents. For example, the distances among the first three documents in Table 1 are same in one-hot vector space, since there is only one different word. But from semantic point of view, D_1 is more similar to D_2 than to D_3. In order to solve this problem, the semantics of documents should be modeled. Distributed representation is a quite effective method for addressing this problem.

Specifically, documents are represented by compact low dimension continuous-value vectors with randomly initialized values. Document vectors are learned by predicting which words belonging to them and which are not. Semantics such as synonyms can be modeled by document embeddings. For example, D_1 tends to be closer to D_4 in the new vector space, since they both need to predict the same word awesome. D_2 tends to be closer to D_4 due to the same reason. This will make D_1 to be much closer to D_2 than to D_3 since both D_1 and D_2 have the same neighbor D_4.

More formally, the objective of the document embeddings model is to maximize the following log probability

$$\sum_i \sum_j \log p\left(w_{i,j}|d_i\right) \tag{1}$$

where d_i denotes the i^{th} document from document set D and $w_{i,j}$ represents the j^{th} word of d_i.

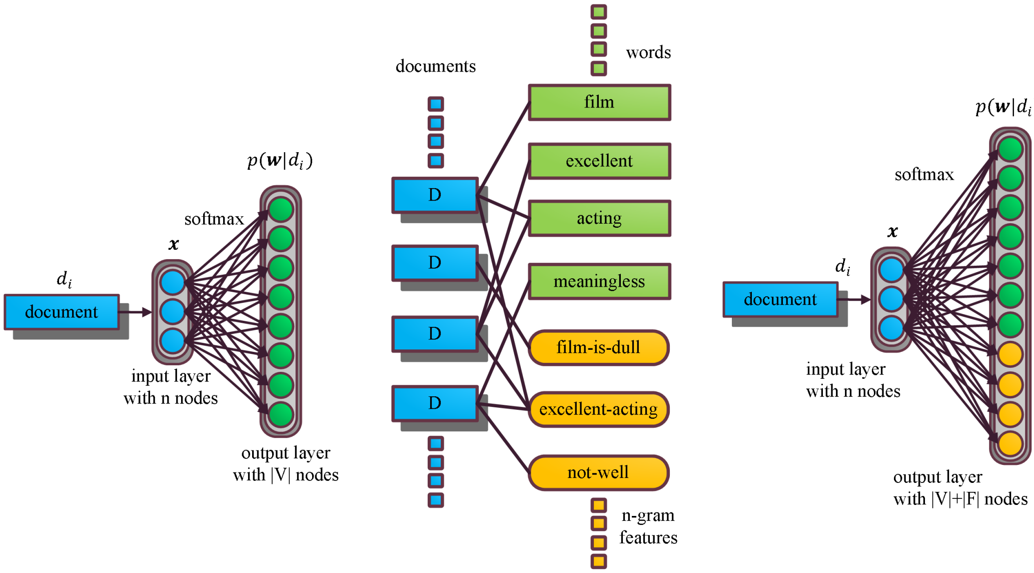

Figure 2: (a) basic DV-ngram model. (b) illustration of n-gram features. (c) DV-ngram model.

In order to compute this probability, a simple neural network is built with a softmax output layer(as depicted in Figure 2.1-a). The input layer of this network has n nodes which represent the document vector, denoted by x. The output layer has $|V|$ (vocabulary size) nodes and the k^{th} node represents the probability that the k^{th} word belongs to this document. This probability can be written as

$$\log p\left(w_{i,j}|d_i\right) = \frac{e^{y_{w_{i,j}}}}{Z} \tag{2}$$

where $y_{w_{i,j}}$ is the unnormalized log-probability for each target word $w_{i,j}$, which can be calculated using $\mathbf{y} = \mathbf{b} + W\mathbf{x}$. W and $\mathbf{b}$ are the networks weights and biases. Z in equation 2 denotes the normalized factor which basically sums up all possible $e^{y_{w_{i,j}}}$

In our model, Stochastic Gradient Descent (SGD) (Williams & Hinton, 1986) is used in all of our experiments for learning.

2.2 Improved Model for Modeling Word Order

Word order is often essential for understanding documents. For example, the following two texts have exact the same words but express totally different meanings due to their different word order: "Despite dull acting, this film is excellent", "Despite excellent acting, this film is dull". In order to model word order, distributed representation of documents is learned by predicting not only its belonging words but also word sequences. For simplicity, n-gram is directly used as word sequence features, which is illustrated by "film-is-dull", "excellent-acting" and "not-well" as shown in Figure 2.1-b. More sophisticated word sequences selecting methods may be investigated in the future.

In practice, each word sequence is treated as a special token and is directly appended to each document. The output layer of the above neural network is also expanded as shown in Figure 2.1-c. Thus, documents that contain semantically similar word sequences also tend to be closer to each other in vector space.

As shown later in our experiments, much better performance can be obtained by this improved model.

2.3 Learning Acceleration

In practice, since the size of vocabulary V and feature set F can be very large, our model needs to compute the output values of $|V| + |F|$ nodes in output layer, which results in computation inefficiency. Negative sampling technique (Mikolov et al., 2013) is used to accelerate the training process. Negative sampling is especially efficient and simple, it only calculates the values of K nodes (K is a small constant) compared to standard softmax which calculates $|V| + |F|$ nodes in each training step. More precisely, negative sampling basically calculates equation 1 as

$$\sum_i \sum_j \left[f\left(x_{w_{i,j}}^\top x_{d_i} \right) + \sum_{k=1}^{K} f\left(-x_{w_{\text{random}}}^\top x_{d_i} \right) \right] \tag{3}$$

where $x_{w_{i,j}}$ represents the vector of j^{th} word/feature from i^{th} document. x_{d_i} represents the vector of i^{th} document. w_{random} represents the vector of word randomly sampled from the vocabulary based on words frequency. K is the negative sampling size and f is sigmoid function.

In summary, in order to get desired document vector, DV-ngram first randomly initialize each document vectors. Then stochastic gradient descent is used to maximize equation 3 to get desired document vectors. The document vectors are eventually sent to a logistic regression classifier for sentiment classification. Note that DV-ngram use no labeled information thus is unsupervised. As shown in our experiments, additional unlabeled data can be use to improve model's performance.

3 Experiments

3.1 Dataset and Experimental Setup

Our model is benchmarked on well-studied IMDB sentiment classification dataset (Maas et al., 2011). This dataset contains 100,000 movie reviews, of which 25,000 are positives, 25,000 are negatives and the rest 50,000 are unlabeled. Average document length of this dataset is 231 words. Accuracy is used to measure the performance of sentiment classification.

For comparison with other published results, we use the default train/test split for IMDB dataset. Since development data are not provided by two datasets, we refer the previous method of Mesnil et al. (2014), i.e. 20% of training data are selected as development data to validate hyper-parameters and experiment settings, optimal results are shown in Table 2.

Table 2: Optimal hyper-parameters and experiment settings

Vector size	Learning rate	Mini-batch	Iteration	Negative sampling size
500	0.25	100	10	5

Document vectors and parameters of neural network are randomly initialized with values uniformly distributed in the range of [-0.001, +0.001]. We use logistic regression classifier in LIBLINEAR package (Fan et al., 2008) [3] as the sentiment classifier.

In order to reduce the effect of random factors, training and testing were done for five times and the average of all the runs was obtained.

The experiments can be reproduced using our DV-ngram package, which can be found at `https://github.com/libofang/DV-ngram`.

3.2 Comparison with Bag-of-ngram Baselines

Our model is first evaluated by comparing with traditional bag-of-ngram baselines since they both use n-gram as feature. The biggest difference of these two kinds of methods is the way of representing documents. Bag-of-ngram methods use one-hot representation which loses semantics in some extent. DV-ngram is superior for modeling semantics since it represents documents by compact low dimension continuous-value vectors.

Table 3: Comparison of DV-ngram with bag-of-ngram baseline.

Model	**Unigram**	**Bigram**	**Trigram**
bag-of-ngram	86.95	89.16	89.00
DV-ngram (our model)	89.12	90.63	91.75
DV-ngram+Unlabd (our model)	**89.60**	**91.27**	**92.14**

As shown in Table 3, DV-ngram with different n-grams consistently outperforms corresponding bag-of-ngram methods. This results also suggests that the performance of DV-ngram can be further improved by adding more unlabeled sentiment related documents. Note that some other models are inherently unable to make use of this additional data such as the bag-of-ngram methods in this table. The best performance is achieved by DV-tri, for simplicity, we will report only the result of DV-tri in following experiments.

3.3 Comparison with Other Models

DV-ngram is compared with both traditional bag-of-ngram based models and deep learning models. Any type of model or feature combination is not considered here for comparison fairness, combination will be discussed later. Additional unlabeled documents are used by Maas, PV and DV-ngram when learning document vectors but not used by other methods since they are task specified.

As shown in Table 4, DV-ngram greatly outperforms most of other deep learning models. Especially, DV-tri outperforms PV 3.41 percent in terms of accuracy. This result shows that the prediction of word sequences is important for document embeddings. Note that even the simplest DV-uni (use words alone with no n-gram feature) outperforms PV 0.87 percent in terms of accuracy. This result supports our claim in Section 1 that the way PV handles context information may not suitable for sentiment analysis of movie reviews.

Among all other models, NBSVM is the most robust model for this dataset. NBSVM basically use labeled information to weight each words. Even though DV-ngram use no labeled information, it still outperforms NBSVM and achieves the new single model state-of-the-art results on IMDB dataset.

[3] Available at `http://www.csie.ntu.edu.tw/~cjlin/liblinear/`

Table 4: Comparison of DV-ngram with other models. [4]

Bag-of-ngram based models	**Accuracy**
LDA (Maas et al., 2011)	67.42
LSA (Maas et al., 2011)	83.96
MNB-bi (Wang & Manning, 2012)	86.59
NBSVM-bi (Wang & Manning, 2012)	91.22
NBSVM-tri (Mesnil et al., 2014)	**91.87**

Deep learning models	**Accuracy**
RNN-LM (Mikolov, 2012)	86.60
WRRBM (Dahl et al., 2012)	87.42
DCNN (Kalchbrenner et al., 2014)	89.4
DAN (Iyyer et al., 2015)	89.4
seq-CNN (Johnson & Zhang, 2015)	91.61
DV-tri (our model)	**91.75**
Maas (Maas et al., 2011)	87.99
PV (Le & Mikolov, 2014)	88.73
DV-tri+Unlab'd (our model)	**92.14**

3.4 Feature Combination

In practice, more sophisticated supervised features such as Naive Bayes weigted bag-of-ngram vectors (NB-BO-ngram) (Wang & Manning, 2012) can be used to improve performance of classification. Previous state-of-the-art results obtained by feature combination is achieved by an ensemble model named seq2-CNN (Johnson & Zhang, 2015). The seq2-CNN model integrates three kind of vectors including NB-BO-ngram in a parallel convolutional neural network. For our model, we directly concatenate the learned document vectors with NB-BO-ngram for classification. As shown in table 5, when integrated with NB-BO-ngram, our model achieves new state-of-the-art result among feature combination models.

Table 5: Different feature combination results.

Model	**Alone**	**+NB-BO-tri**
seq2-CNN (Johnson & Zhang, 2015)	91.96	92.33
DV-tri (our model)	91.75	92.74
DV-tri+Unlab'd (our model)	**92.14**	**92.91**

3.5 Model Ensemble

Recently, a new ensemble model (Mesnil et al., 2014) is proposed, which achieves the new state-of-the-art result for ensemble models on IMDB dataset. Optimal weights are obtained by grid search for each sub-model. In our experiment, we find the weights for different models are almost the same. For simplicity, we directly combine our model with others without weighting.

As shown in Table 6, the previous best performance is obtained by combining PV, RNN-LM and NBSVM (NBSVM with trigram). Without much surprise, a new state-of-the-art result is obtained by replacing PV to our model. Note that combining with or without RNN-LM do not affect results much. One reason for this may be that RNN-LM becomes burdensome when combined with more robust model since RNN-LM alone only achieves 86.6 percent in terms of accuracy.

[4] Result of DCNN is reported by Iyyer et al. (2015). Results of RNN-LM and PV are reported by Mesnil et al. (2014)

Table 6: Different model ensemble results. R: RNN-LM. N: NBSVM.

Model	Alone	+R	+N	+R+N
PV (Mesnil et al., 2014)	88.73	90.40	92.39	92.57
DV-tri (our model)	91.75	92.10	92.81	92.89
DV-tri+Unlab'd (our model)	**92.14**	**92.31**	**93.00**	**93.05**

4 Conclusion

A new method for learning document embeddings has been proposed for sentiment analysis of movie reviews, which is based on recently proposed Paragraph Vector. Instead of learning both document vectors and word vectors simultaneously by predicting the target word, our model learns document vectors alone by predicting both their belonging words and n-gram features. In this way, the expressive power of document vectors is kept. Experimental results show that the proposed model outperforms PV due to this reason.

Furthermore, comparing with traditional bag-of-ngram models, our model can represent the semantics which is important for sentiment analysis. Our model is also compared with other deep learning and bag-of-ngram based models and achieves the state-of-the-art results on IMDB dataset. We also show that the performance of our model can be further improved by adding unlabeled data.

Finally, when combined with NBSVM and RNN-LM, our model achieves state-of-the-art result among all other ensemble models.

The source code of our model will be published together with this paper. We hope this could allow researchers to reproduce our experiments easily for further improvements and applications to other tasks.

5 Acknowledgements

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

References

Bengio, Yoshua, Ducharme, Réjean, Vincent, Pascal, and Janvin, Christian. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

Collobert, Ronan and Weston, Jason. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 160–167. ACM, 2008.

Collobert, Ronan, Weston, Jason, Bottou, Léon, Karlen, Michael, Kavukcuoglu, Koray, and Kuksa, Pavel. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

Dahl, George E., Adams, Ryan Prescott, and Larochelle, Hugo. Training restricted boltzmann machines on word observations. In *ICML*, 2012.

Fan, Rong-En, Chang, Kai-Wei, Hsieh, Cho-Jui, Wang, Xiang-Rui, and Lin, Chih-Jen. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

Iyyer, Mohit, Manjunatha, Varun, Boyd-Graber, Jordan L., and III, Hal Daum. Deep unordered composition rivals syntactic methods for text classification. In *ACL*, 2015.

Johnson, Rie and Zhang, Tong. Effective use of word order for text categorization with convolutional neural networks. In *NAACL*, 2015.

Kalchbrenner, Nal, Grefenstette, Edward, and Blunsom, Phil. A convolutional neural network for modelling sentences. In *ACL*, 2014.

Kim, Yoon. Convolutional neural networks for sentence classification. In *EMNLP*, 2014.

Kiros, Ryan, Zhu, Yukun, Salakhutdinov, Ruslan, Zemel, Richard S, Torralba, Antonio, Urtasun, Raquel, and Fidler, Sanja. Skip-thought vectors. *arXiv preprint arXiv:1506.06726*, 2015.

Le, Quoc V. and Mikolov, Tomas. Distributed representations of sentences and documents. In *ICML*, 2014.

Maas, Andrew L., Daly, Raymond E., Pham, Peter T., Huang, Dan, Ng, Andrew Y., and Potts, Christopher. Learning word vectors for sentiment analysis. In *ACL*, 2011.

Mesnil, Grégoire, Ranzato, Marc'Aurelio, Mikolov, Tomas, and Bengio, Yoshua. Ensemble of generative and discriminative techniques for sentiment analysis of movie reviews. *arXiv preprint arXiv:1412.5335*, 2014.

Mikolov, Tomáš. Statistical language models based on neural networks. *Presentation at Google, Mountain View, 2nd April*, 2012.

Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Gregory S., and Dean, Jeffrey. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

Schwenk, Holger. Continuous space language models. *Computer Speech & Language*, 21(3):492–518, 2007.

Socher, Richard, Lin, Cliff Chiung-Yu, Ng, Andrew Y., and Manning, Christopher D. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, 2011.

Socher, Richard, Perelygin, Alex, Wu, Jean Y, Chuang, Jason, Manning, Christopher D, Ng, Andrew Y, and Potts, Christopher. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, pp. 1642. Citeseer, 2013.

Srivastava, Nitish, Salakhutdinov, Ruslan, and Hinton, Geoffrey E. Modeling documents with deep boltzmann machines. In *UAI*, 2013.

Wang, Sida I. and Manning, Christopher D. Baselines and bigrams: Simple, good sentiment and topic classification. In *ACL*, 2012.

Williams, DE Rumelhart GE Hinton RJ and Hinton, GE. Learning representations by back-propagating errors. *Nature*, pp. 323–533, 1986.