Weighted Neural Bag-of-n-grams Model: New Baselines for Text Classification

Bofang Li[*] **, Zhe Zhao**[*]**, Tao Liu, Puwei Wang**[†]**, Xiaoyong Du**
School of Information, Renmin University of China
Key laboratory of Data Engineering and Knowledge Engineering, MOE
{libofang,helloworld,tliu,wangpuwei,duyong}@ruc.edu.cn

Abstract

NBSVM is one of the most popular methods for text classification and has been widely used as baselines for various text representation approaches. It uses Naive Bayes (NB) feature to weight sparse bag-of-n-grams representation. N-gram captures word order in short context and NB feature assigns more weights to those important words. However, NBSVM suffers from sparsity problem and is reported to be exceeded by newly proposed distributed (dense) text representations learned by neural networks. In this paper, we transfer the n-grams and NB weighting to neural models. We train n-gram embeddings and use NB weighting to guide the neural models to focus on important words. In fact, our methods can be viewed as distributed (dense) counterparts of sparse bag-of-n-grams in NBSVM. We discover that n-grams and NB weighting are also effective in distributed representations. As a result, our models achieve new strong baselines on 9 text classification datasets, e.g. on IMDB dataset, we reach performance of 93.5% accuracy, which exceeds previous state-of-the-art results obtained by deep neural models. All source codes are publicly available at https://github.com/zhezhaoa/neural_BOW_toolkit.

1 Introduction

Text representation is a core technology for many NLP tasks. Most text representation approaches fall into one of the two classes: sparse and distributed (dense) representations. One of the most popular sparse representations is bag-of-words (BOW), where each dimension represents the number of occurrences of a word in a text. Though simple, BOW enjoys the advantages of being efficient and surprisingly effective. Until now, BOW representation still serves as baselines on a range of NLP tasks.

In distributed representation, texts are represented by low-dimensional real vectors. Recently, there has been a surge of work proposing to learn distributed text representation through neural networks. There exists two lines of researches in neural models. The first is order/syntax-aware models, such as Convolutional Neural Networks (CNNs) (Kim, 2014; Kalchbrenner et al., 2014; Zhang et al., 2016), Recurrent Neural Networks (RNNs) (Dai and Le, 2015) and Recursive Neural Networks (RecNNs) (Socher et al., 2011; Socher et al., 2012). In these models, words are firstly embedded into low-dimensional real vectors (word embeddings) as the input, and then order/syntax-aware compositions upon words are learned by neural networks (Goldberg, 2015). Another line is neural bag-of-words models, where unordered compositions are learned upon word embeddings (Iyyer et al., 2015). The simplest neural bag-of-words model is word embeddings average. Compared to order/syntax-aware models, they are much more efficient in training (Iyyer et al., 2015) but lose the accuracies due to the ignorance of the order/syntax information.

To the best of our knowledge, most of the neural models map isolated words (uni-grams) to embeddings. Sometimes, consecutive words (n-grams), such as ‘not like’ and ‘as good as’ can convey semantics that are difficult to be obtained by simple compositions of individual words (Mikolov et al., 2013b). A natural extension is to embed n-grams into low-dimensional real vectors, e.g. the embedding of bi-gram

[*]Equal contribution. [†] Corresponding author.

'not like' should be close to the embedding of word 'dislike'. The introduction of the n-gram embeddings takes the word order in short context into consideration, and can further enrich the semantics of text representations.

A distinct characteristic of neural models is their automatic feature extraction ability. Most neural models directly learn compositions upon word embeddings, and are reported to be powerful enough to learn high-quality distributed representations without human intervention. However, in sparse case, heuristic weighting techniques designed by humans are shown to be able to bring significant improvements over raw BOW representation (Wang and Manning, 2012; Martineau and Finin, 2009). For example, in sentiment classification, word 'amazing' is much more important than words like 'movie' and 'of', and should be given more weight in sparse BOW representation. Weighting technique has been successfully applied to sparse representation, but is still seldom used in neural models. Intuitively, neural models can also benefit a lot from knowing the importance of words in advance to guide the training processes.

In this paper, both n-grams and weighting techniques are introduced into the neural bag-of-words models. In fact, our models can be regarded as a neural or distributed counterparts of NBSVM (Wang and Manning, 2012). In NBSVM, these two techniques are shown to be very useful for sparse representations and strong baselines are achieved on a range of text classification tasks. In our work, we show how to transfer these two techniques to distributed representations, and discover that they are also very effective in distributed case.

We evaluate our models on 9 text classification tasks. Significant improvements are witnessed when n-gram and weighting techniques are introduced into neural bag-of-words models. As a result, new strong baselines are achieved on 5 document-level datasets and 4 sentence-level datasets. Most recently, state-of-the-art results on NLP tasks are dominated by deep neural models: CNNs exploit convolutional filters to extract n-grams features from texts; RNNs are reported to be able to capture long-distance patterns from natural languages. RecNNs even take syntactic information into consideration. In theory, these models are very powerful since complex compositionalies are learned upon word embeddings. However, experimental results in this paper give us further insights: though n-gram features have been studied in the NLP literature for decades and are usually viewed as baselines, they can still outperform features learned by the newly proposed deep neural models in many datasets if we can make use of them correctly. At least in text classification tasks, complex deep neural models do not show obvious superiority over our n-grams models on accuracies. What is more, deep neural models always require much more computational resources compared to bag-of-words (n-grams) models.

2 Related Work

Bag-of-words (n-grams)
Bag-of-words (n-grams) models treat a text as a set of words (n-grams), which ignore the fact that texts are essentially sequential data (Pang et al., 2002). Though the order information contained in word sequences is discarded, bag-of-words (n-grams) models are surprisingly effective, and also enjoy the advantages of being efficient and robust. They have been widely used in various kinds of NLP tasks such as information retrieval, question answering and text classification. Usually, sparse BOW features require weighting techniques to achieve better performance, where important words are given more weights while unimportant words are given less weights. For example, NBSVM (Wang and Manning, 2012) uses the ratio of the number of words in positive texts and negative texts to weight words, and achieves competitive results on a range of text classification tasks. However, traditional sparse BOW representations take each word or n-gram as a unit and ignore the internal semantics of them. As a result, they tend to generalize poorly compared with the newly proposed distributed representations.

Recently, distributed text representations are widely used in NLP tasks. The most fundamental work in the distributed representation literature is word embedding. In word embedding algorithms, syntactic and semantic information of words is encoded into low-dimensional real vectors and similar words tend to have close vectors (Bengio et al., 2003; Collobert et al., 2011; Mikolov et al., 2013a; Mikolov et al., 2013b). To obtain text embedding from word embedding, the simplest way is word embeddings (vectors) average (VecAvg). This method belongs to neural bag-of-words models based on the fact that VecAvg also treats text as a set of words and discards order information totally. Other popular neural bag-of-words models include Deep Average Network (DAN) (Iyyer et al., 2015) and Paragraph Vector (PV) (Le and Mikolov, 2014). DAN constructs multiple neural layers upon the average of word vectors in the text. They show that more discriminative features can be extracted by deepening the layers of the neural networks. In PV, text embedding are trained to be useful to predict the words in the text. These neural bag-of-words models are reported to perform better than sparse BOW representations, and inherit the advantages of BOW models of being efficient and robust.

Complex Compositionality

Recently, many deep neural models are proposed on the basis of compositionality: the meanings of the expressions depend on their constituents. These models can learn complex compositionality upon words and achieve state-of-the-art results on a range of NLP tasks. Kim (2014) proposes to use convolutional neural networks (CNNs) to extract text features. N-grams information is extracted by convolutional layers and the most distinct features are selected by max pooling layers. Recurrent Neural Networks (RNNs) are sequential models and are suitable for texts data in nature. Hidden layers of RNNs can preserve the historical sequential information, and can be used as the representations of the texts (Dai and Le, 2015). However, both CNNs and RNNs are essentially 'flat' models, where structural information (e.g. syntactic parse tree) from texts are generally ignored. Socher et al. (2011) propose to use Recursive Neural Networks (RecNNs) to learn syntactic-aware compositionality upon words. They recursively combine neighbor nodes on parse trees in a bottom-up fashion until the root is reached.

Most recently, many researchers have focused on using the combinations of neural networks to achieve better text representations. Sutskever et al. (2014) propose to use multi-layer LSTM networks for text representation and significant improvement is achieved on machine translation when more layers of neural networks are added. Lai et al. (2015) use RNN-CNN for text representations, where RNN layer is used for extracting word sequences information and the most distinct features are selected by max-pooling layer. Tai et al. (2015) model the texts through tree-structured LSTM, which can be viewed as the combination of RecNN and RNN. To take the relationships among sentences in a document into consideration, some works have been done to learn document representation hierarchically (Kiros et al., 2015). For example, in (Li, 2014), RecNN is used for learning sentence embedding from word embedding, and RNN is used for learning document embedding from sentence embedding. Denil et al. (2014), Lin et al. (2015), Li et al. (2015b) and Tang et al. (2015) respectively propose to use CNN-CNN, RNN-RNN, LSTM-LSTM and CNN-LSTM to represent documents hierarchically.

These models are very powerful in theory since they exploit extensive information of texts such as word order, syntax and even relations among sentences. However, for text classification, a relatively simple task in the NLP community, deep neural models do not show their superiority over our n-grams models according to our experimental results. It still requires further explorations to demostrate if deep neural models can really exploit complex information beyond n-grams, or if complex information is really useful for text classification. What is more, most deep neural models require intensive training resources and usually need careful hyper-parameter tuning for specific datasets.

3 Models

In the following subsections, we present the framework of exploiting n-grams and weighting techniques for neural bag-of-words models. We use Paragraph Vector (PV) model (Le and Mikolov, 2014)[1] as a concrete case to illustrate the way of introducing these two techniques. Our method can be applied to other neural bag-of-words models straightforwardly.

Paragraph Vector (PV) is a popular method for text representation. In PV, text embedding is trained to be useful to predict words in the text. Formally, the objective is to maximize the conditional probabilities of words given their texts:

[1]Concretely, PV-DBOW (a variant of PV) is used in our paper.

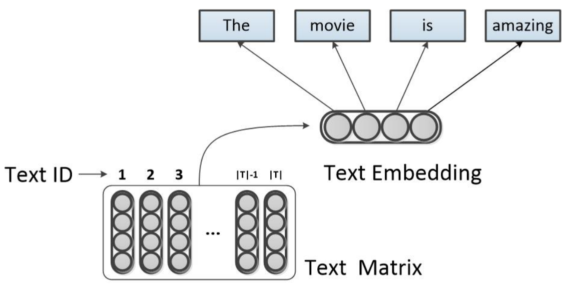

Figure 1: Illustration of the original Paragraph Vector model. The model only considers the uni-grams and they are equally treated in the model.

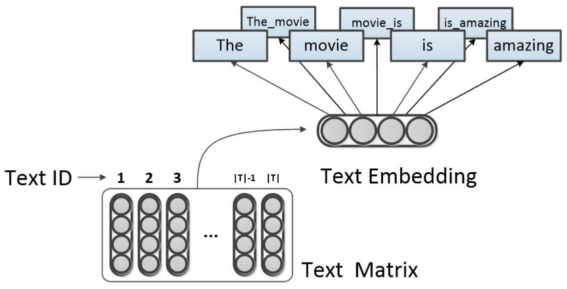

Figure 2: Illustration of n-gram PV model, where n-grams are predicted by the text embedding.

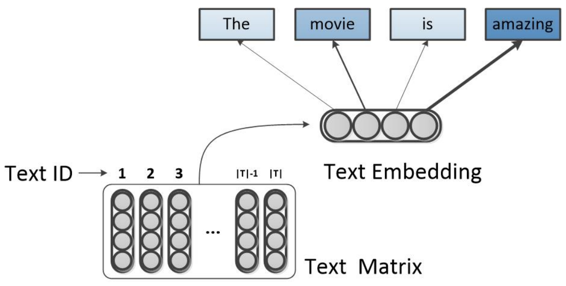

Figure 3: Illustration of weighted PV model, where important words are given more attention during the training process.

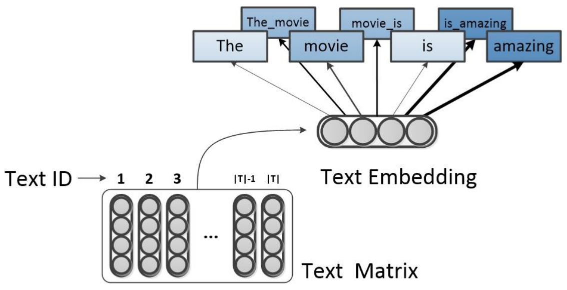

Figure 4: Combination of n-gram and weighting techniques. Text embeddings are trained to be useful to predict important n-grams during the training process.

$$\sum_{i=1}^{|T|}\sum_{j=1}^{|t_i|} logP(w_{i,j}|t_i) \tag{1}$$

where t_i=$\{w_{i,1}$,$w_{i,2}$,......,$w_{i,|t_i|}\}$ denotes the i_{th} text and T=$\{t_1$,t_2,......,$t_{|T|}\}$ denotes the whole dataset. In this paper, the conditional probability is defined by negative sampling (Mikolov et al., 2013b), which speeds up the training process significantly. Figure 1 clearly shows that, in PV, each word is predicted separately and the words in a text are equally treated.

3.1 N-gram Embedding

As we have mentioned above, n-grams can reflect semantics that can not be captured by looking at words individually. To introduce n-grams information into the neural models, a natural extension is to train n-gram embeddings (Li et al., 2015a). Each text is regarded as a set of n-grams (e.g. n = 1, 2, 3) and each n-gram is assigned a randomly initialized vector. The length of the text *t* is denoted by $|t|$. Obviously, text *t* consists of $|t|$ uni-grams, $|t|$-1 bi-grams and $|t|$-2 tri-grams.

During the training process, text embeddings are trained to be useful to predict n-grams in the text (as shown in figure 2). The following objective is optimized:

$$\sum_{i=1}^{|T|}\sum_{n=0}^{N-1}\sum_{j=1}^{|t_i|-n} logP(w_{i,j\sim j+n}|t_i) \tag{2}$$

where *N* denotes the number of consecutive words considered. When *N* equals to 3, uni-grams, bi-grams and tri-grams are predicted by the text. $w_{i,j\sim j+n}$ denotes *n+1* consecutive words $w_{i,j}, w_{i,j+1}, ..., w_{i,j+n}$. Word order in short context is captured by including n-grams information.

3.2 Naive Bayes Weighting

Intuitively, some n-grams are more important and should be paid more attention during the training process. In this paper, we only consider calculating weights of n-grams for binary classification. Without loss of generality, we use 'positive' and 'negative' to denote the name of two classes. Following works done by Wang and Manning (2012) and Martineau and Finin (2009), we use Naive Bayes feature to assign weight for each word. The weight of word *w* is calculated as follows:

$$(\frac{(\#POS(w)+\alpha)/|POS|}{(\#NEG(w)+\alpha)/|NEG|})^{\beta * \lfloor \#POS(w)/|POS| > \#NEG(w)/|NEG| \rfloor} \tag{3}$$

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

where *#POS(w)* denotes the number of positive texts that contain n-gram *w*, and *#NEG(w)* denotes the number of negative texts that contain n-gram *w*. α and β are two hyper-parameters for smoothing ratios (both of them are set to be 0.5 in this paper). $|POS|$ and $|NEG|$ denote the number of positive and negative texts respectively (including pseudo count α). To this end, words that have uneven distributions over classes are given more weights. Naive Bayes weighting is shown to be effective in sparse representation. To introduce weighting techniques into distributed case, weighted objective function is optimized to train text embeddings:

$$\sum_{i=1}^{|T|} \sum_{n=0}^{N-1} \sum_{j=1}^{|t_i|-n} Weight(w_{i,j \sim j+n}) logP(w_{i,j \sim j+n} | t_i) \tag{4}$$

where *Weight(•)* denotes the weight of n-gram •. As a result, text embeddings are trained to predict those important n-grams in larger probabilities rather than those words that have little discriminative information for classification. Figure 3 and 4 respectively demonstrate the overview of introducing weighting techniques into uni-gram and n-gram Paragraph Vector. The way we exploit n-grams and weighting information can be easily used in other neural bag-of-words models. For example, we can use the weighted average of n-gram embeddings as the input of the DAN neural networks (Iyyer et al., 2015).

In summary, this section introduces n-grams and NB weighting into neural bag-of-words models. These two techniques have shown their effectiveness on sparse representation in NBSVM. In the following Experiment Section, we demonstrate the effectiveness of n-grams and NB weighting for distributed representations.

4 Experiments

4.1 Datasets and Training Protocols

We evaluate our models on five document-level and four sentence-level text classification datasets. The details of the datasets are shown in table 1. The pre-processing of texts and the train/test, cross-validation splits strictly follow the implementation in NBSVM[2].

Our models are trained by stochastic gradient descent (SGD). The hyper-parameter setting follows the implementation in (Mesnil et al., 2014) [3] except that the training iterations are determined by validation set. When the training process is finished, the text representations are fed into a logistic regression classifier. Following the work done by Mesnil et al. (2014), we also combine the outputs of logistic classifiers (for sparse and dense representations) via linear interpolation. The weights are determined by validation set.

It is common to use pretrained vectors (Kim, 2014) and additional unlabeled texts (Mesnil et al., 2014; Li et al., 2015a)[4] to assist the training when size of the dataset is small. IMDB is a relatively large-scale

[2]https://github.com/sidaw/nbsvm

[3]https://github.com/mesnilgr/iclr15

[4]Mesnil et al. use the unlabeled data in their published implementation.

dataset and does not require pretrained vectors. For the rest of eight tasks, we use word2vec vectors to initialize the uni-grams. The additional unlabeled texts are added to RT-2k and RTs datasets, the details of which will be discussed in the following subsections.

Unless otherwise noted, we don't perform any data specific hyper-parameter tuning.

Type	Dataset	$\#(train+, train-, test+, test-)$	CV	$\lvert\bar{t}\rvert$	|V|
document-level	IMDB	(12500,12500,12500,12500)	N	231	392K
	RT-2k	(1000,1000)	10	787	51K
	AthR	(399,315,400,313)	N	345	22K
	Xgraph	(491,486,489,487)	N	261	32K
	BbCrypt	(497,496,497,495)	N	269	25K
sentence-level	RTs	(5331,5331)	10	21	21K
	CR	(2406,1366)	10	20	5713
	MPQA	(3316,7308)	10	3	6229
	Subj.	(5000,5000)	10	24	24K

Table 1: Datasets statistics. #(train+,train-,test+,test-): the number of positive and negative samples in train, test set respectively. For datasets that use cross-validation to evaluate models, column 3 only lists the number of positive and negative samples. CV: the number of cross-validation splits. N denotes train/test split. $|\bar{t}|$ denotes the average length of text samples. $|V|$ denotes the vocabulary size.

4.2 Comparison of Models on IMDB Dataset

IMDB dataset is one of the most popular benchmarks in text classification. A large amount of models are evaluated and compared on this dataset. In table 2, we compare our neural bag-of-words models with NBSVM. We can observe that dense representations consistently outperform their sparse counterparts. The n-grams and NB weighting are effective for both sparse and dense representations. Four percent improvement in accuracies is witnessed when n-grams and NB weighting are considered in dense representation. The best result is achieved by the ensemble of dense and sparse represntations (Mesnil et al., 2014).

In table 3, we compare our models with state-of-the-art methods which are dominated by deep neural models. To better compare different methods, we divide the existing models into three groups according to how they exploit information in the texts. Models in the first group treat a text as a bag of words or n-grams. Models in the second group treat texts as sequential data. In the third group, structural information such as syntactic parse tree and relationships among sentences is taken into consideration. In theory, accuracy should benefit from the sequential and structural information of texts. However, we surprisingly find that our models outperform other approaches, even though only n-gram information is exploited in our models.

Models	Sparse	Dense
Unigram	86.95	88.97
Unigram+NB	88.29	**90.10**
Bigram	89.16	91.27
Bigram+NB	91.22	**92.20**
Trigram	91.4	92.14
Trigram+NB	91.87	**92.95**
Ensemble	**93.51**	

Table 2: Comparison of sparse and dense representations. Dense representations consistently outperform sparse representations. N-grams and NB weighting are effective in both sparse and dense cases

Group	Model	Accuracy
bag-of-words	NBSVM-tri(Mesnil et al., 2014)	91.87
	Paragraph Vector(Mesnil et al., 2014)	88.73
	DAN(Iyyer et al., 2015)	89.40
	DV-tri(Li et al., 2015a)	92.14
	our model	92.95
	our ensemble	**93.51**
sequential	word2vec-LSTM(Dai and Le, 2015)	90.00
	SA-LSTM(Dai and Le, 2015)	92.76
	seq2-bown-CNN(Johnson and Zhang, 2015a)	92.33
	CNN+unsup3-tv(Johnson and Zhang, 2015b)	**93.49**
	Ensemble(Mesnil et al., 2014)	92.57
structural	DCNN(Denil et al., 2014)	89.40
	BENN(Li, 2014)	**91.00**
	RecRNN-RNN(Li, 2014)	87.00

Table 3: Comparison of state-of-the-art approaches, which are grouped according to how they exploit texts informtaion

4.3 Comparison of Models on Document-level Datasets

In this subsection, we continue evaluating our methods on document-level datasets. RT-2k dataset contains 2000 movie reviews. Texts in RT-2k and IMDB are both movie reviews and are classified according to whether they are positive or negative. Consequently, texts in IMDB are suitable alternative as additional unlabeled texts for RT-2k. AthR, XGraph and BbCrypt are classification pairs from 20-newsgroups. In these datasets, pretrained word2vec vectors are used to initialize uni-gram embeddings. N-gram embeddings are initialized randomly. To the best of our knowledge, state-of-the-art results on these four document-level datasets are still achieved by NBSVM. In table 4, we compare our models with their sparse counterparts NBSVM. We discover that dense representations benefit a lot from n-gram and weighting techniques. However, dense representations do not perform consistently better than sparse representations.

Not surprisingly, ensemble of the dense and sparse models achieves the best results.

Models	**RT2k**		**AthR**		**XGraph**		**BcCrypt**	
	Sparse	**Dense**	**Sparse**	**Dense**	**Sparse**	**Dense**	**Sparse**	**Dense**
Unigram	86.3	87.6	82.6	79.0	85.1	89.2	98.3	98.9
Unigram+NB	87.8	**88.7**	**87.9**	84.0	**91.2**	90.2	**99.7**	99.2
Bigram	87.4	89.2	83.7	81.1	86.2	90.5	97.7	99.1
Bigram+NB	89.5	**90.5**	**87.7**	86.1	90.7	**90.8**	99.5	**99.7**
Ensemble	**91.4**		**88.0**		**92.0**		**99.7**	

Table 4: Comparison of sparse and dense representations on document-level datasets.

4.4 Comparison of Models on Sentence-level Datasets

In this subsection, we demonstrate the effectiveness of our models on four sentence-level datasets: RTs, MPQA, CR and Subj. Pretrained word2vec vectors are used to initialize uni-gram embeddings. For RTs, data in IMDB dataset are used as additional unlabeled texts. From table 5, it can be observed that comparable results are achieved by dense and sparse representations. Similar with the conclusions obtained in above subsections: both n-grams and NB weighting improve the accuracies significantly. Ensemble of dense and sparse representations gives the best results.

In table 6, we make comparisons of state-of-the-art models, which are grouped according to their ways of exploiting text information. From the first row of table 6, we can observe that traditional PV performs poorly on sentence-level datasets compared to other state-of-the-art models. When n-grams, NB weighting and pretrained word embeddings are introduced, decent accuracies are achieved by PV. We can also observe that deep neural models show their superiority over bag-of-words models. Since n-grams information contained in sentence-level texts is limited, sequential and structural information is important for achieving better accuracies on these datasets. Nevertheless, most deep neural models in table 6 can not be extended to document-level texts. In contrast, our models can be used for texts of variable length.

Models	**RTs**		**MPQA**		**CR**		**Subj.**	
	Sparse	**Dense**	**Sparse**	**Dense**	**Sparse**	**Dense**	**Sparse**	**Dense**
Unigram	76.2	77.3	**86.1**	81.7	79.0	79.1	90.8	90.5
Unigram+NB	78.1	**78.7**	85.3	81.1	**80.5**	80.3	**92.4**	92.0
Bigram	77.7	78.5	**86.7**	82.0	80.8	80.1	91.7	91.2
Bigram+NB	79.4	**79.5**	86.3	82.1	**81.8**	81.1	**93.2**	92.8
Ensemble	**80.8**		**86.8**		**82.5**		**93.6**	

Table 5: Comparison of sparse and dense representations on sentence-level datasets.

Group	Model	RTs	MPQA	CR	Subj.
bag-of-words	Paragraph Vector(Kiros et al., 2015)	74.8	74.2	78.1	90.5
	NBSVM-bi(Wang and Manning, 2012)	79.4	86.3	81.8	93.2
	DAN(Iyyer et al., 2015)	80.3	-	-	-
	cBoW(Zhao et al., 2015)	77.2	86.4	79.9	91.3
	our model	79.5	82.1	81.1	92.8
	our ensemble	**80.8**	**86.8**	**82.5**	**93.6**
sequential	CNN(Kim, 2014)	81.5	89.6	**85.0**	93.4
	RNN(Zhao et al., 2015)	77.2	90.1	82.3	93.7
	BRNN(Zhao et al., 2015)	**82.3**	**90.3**	82.6	**94.2**
structural	combine-skip(Kiros et al., 2015)	76.5	87.1	80.1	93.6
	GrConv(Zhao et al., 2015)	76.3	84.5	81.3	89.5
	AdaSent(Zhao et al., 2015)	**83.1**	**93.3**	**86.3**	**95.5**

Table 6: Comparison of state-of-the-art approaches on sentence-level datasets.

4.5 Further Discussions

From tables in the above subsections, we can observe that n-gram features are still competitive if we can take full advantages of them. For document-level datasets, the ensemble of dense and sparse representation even outperforms the complex deep neural models which take complex compositions into consideration. For sentence-level datasets, competitive results are achieved by our models when pre-trained vectors or additional unlabelled data is added.

When taking efficiency and robustness into consideration, our n-gram models are better choices. Since our models are essentially bag-of-n-grams models, they only require a fraction of time compared to deep neural models. Our models are also robust for both sentence and document level datasets. In contrast, many deep neural models can not be extended to document-level datasets. Besides that, they usually require careful dataset-specific hyper-parameter tuning for better performance. While in our models, experimental setting is universal to all datasets except that the number of iterations are determined by validation set.

5 Conclusion

In this paper, we propose a framework of introducing n-grams and Naive Bayes weighting into neural bag-of-words models. These two techniques are effective for neural models and new strong baselines are achieved when they are used together. Though many state-of-the-art results in NLP tasks are achieved by deep neural models, we discover that for text classification, n-grams information is sufficient to achieve state-of-the-art accuracies. Moreover, our models inherit efficiency and robustness from bag-of-words representations: they only require a fraction of computational resources compared to deep neural models, and at the same time perform consistently well on a range of datasets without specific hyper-parameter tunings. Our source codes are organized as a text classification toolkit at `https://github.com/zhezhaoa/neural_BOW_toolkit`. We recommend to use the ensemble of dense and sparse representations implemented in our toolkit in real-world challenges.

Acknowledgements

This work is supported by National Natural Science Foundation of China with grant No. 61472428, the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China No. 14XNLQ06. This work is partially supported by ECNU-RUC-InfoSys Joint Data Science Lab and a gift from Tencent.

References

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

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

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, pages 3079–3087.

Misha Denil, Alban Demiraj, Nal Kalchbrenner, Phil Blunsom, and Nando de Freitas. 2014. Modelling, visualising and summarising documents with a single convolutional neural network. *CoRR*, abs/1406.3830.

Yoav Goldberg. 2015. A primer on neural network models for natural language processing. *CoRR*, abs/1510.00726.

Mohit Iyyer, Varun Manjunatha, Jordan L. Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Volume 1: Long Papers*, pages 1681–1691.

Rie Johnson and Tong Zhang. 2015a. Effective use of word order for text categorization with convolutional neural networks. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 103–112.

Rie Johnson and Tong Zhang. 2015b. Semi-supervised convolutional neural networks for text categorization via region embedding. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, pages 919–927.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers*, pages 655–665.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, pages 3294–3302.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2267–2273.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning*, pages 1188–1196.

Bofang Li, Tao Liu, Xiaoyong Du, Deyuan Zhang, and Zhe Zhao. 2015a. Learning document embeddings by predicting n-grams for sentiment classification of long movie reviews. *CoRR*, abs/1512.08183.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015b. A hierarchical neural autoencoder for paragraphs and documents. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Volume 1: Long Papers*, pages 1106–1115.

Jiwei Li. 2014. Feature weight tuning for recursive neural networks. *CoRR*, abs/1412.3714.

Rui Lin, Shujie Liu, Muyun Yang, Mu Li, Ming Zhou, and Sheng Li. 2015. Hierarchical recurrent neural network for document modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 899–907.

Justin Martineau and Tim Finin. 2009. Delta TFIDF: an improved feature space for sentiment analysis. In *Proceedings of the Third International Conference on Weblogs and Social Media*.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems*, pages 3111–3119.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. *CoRR*, cs.CL/0205070.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 151–161.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. 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*, pages 1201–1211.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems*, pages 3104–3112.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Volume 1: Long Papers*, pages 1556–1566.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432.

Sida I. Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *The 50th Annual Meeting of the Association for Computational Linguistics, Volume 2: Short Papers*, pages 90–94.

Rui Zhang, Honglak Lee, and Dragomir R. Radev. 2016. Dependency sensitive convolutional neural networks for modeling sentences and documents. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1512–1521.

Han Zhao, Zhengdong Lu, and Pascal Poupart. 2015. Self-adaptive hierarchical sentence model. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI*, pages 4069–4076.