Initializing Convolutional Filters with Semantic Features for Text Classification

Shen Li[1,2]
shen@mail.bnu.edu.cn

Zhe Zhao[4,5]
helloworld@ruc.edu.cn

Tao Liu[4,5]
tliu@ruc.edu.cn

Renfen Hu[3,†]
irishere@mail.bnu.edu.cn

Xiaoyong Du[4,5]
duyong@ruc.edu.cn

[1] Institute of Chinese Information Processing, Beijing Normal University
[2] UltraPower-BNU Joint Laboratory for Artificial Intelligence, Beijing Normal University
[3] College of Chinese Language and Culture, Beijing Normal University
[4] School of Information, Renmin University of China
[5] Key Laboratory of Data Engineering and Knowledge Engineering, MOE

Abstract

Convolutional Neural Networks (CNNs) are widely used in NLP tasks. This paper presents a novel weight initialization method to improve the CNNs for text classification. Instead of randomly initializing the convolutional filters, we encode semantic features into them, which helps the model focus on learning useful features at the beginning of the training. Experiments demonstrate the effectiveness of the initialization technique on seven text classification tasks, including sentiment analysis and topic classification.

1 Introduction

Recently, neural networks (NNs) dominate the state-of-the-art results on a wide range of natural language processing (NLP) tasks. The commonly used neural networks in NLP include Recurrent NNs, Convolutional NNs, Recursive NNs and their combinations. NNs are known for their strong abilities to learn features automatically. However, the lack of data or inappropriate parameter settings might greatly limit the generalization abilities of the models (Bengio et al., 2009; LeCun et al., 2015; Krizhevsky et al., 2012; Srivastava et al., 2014). To enhance the performance, a lot of improved methods have been proposed, e.g. developing advanced structures (Zhao et al., 2015; Zhang et al., 2016a), introducing prior knowledge (Hu et al., 2016) and utilizing external resources (Xie et al., 2016; Qian et al., 2016).

It is also noteworthy that the neural networks' performance is sensitive to weight initialization because their objectives are non-convex (Glorot and Bengio, 2010; Saxe et al., 2013; Mishkin and Matas, 2015). In fact, initialization techniques even play a role of catalyst for the revival of neural networks (Hinton et al., 2006; LeCun et al., 2015). Most improvements on initializing weights are based on mathematical methods, e.g. xavier initialization (Glorot and Bengio, 2010) and orthogonal initialization (Saxe et al., 2013). For NLP tasks, an influential technique is to use pre-trained word vectors to initialize embedding layers (Kim, 2014; Chen and Manning, 2014). Consider the embedding layers could be initialized by pre-trained word vectors, how about weights in other layers that are still randomly initialized?

Inspired by this question, we propose a simple yet effective method to improve CNNs by initializing convolutional layers (filters). Unlike the previous weight initialization based on mathematical methods, we encode semantic features into the filters instead of initializing them randomly. As CNNs exploit 1-D convolutional filters to extract n-gram features, our method aims at helping the filters focus on learning useful n-grams, e.g. "not bad" which is more useful than "watch a movie" for determining reviews' polarities. Specifically, we select n-grams from training data via a novel Naive Bayes (NB) weighting technique, and then cluster the n-gram embeddings with K-means algorithm. After that, we use the centroid vectors of the clusters to initialize the filters.

With this initialization method, CNN filters tend to extract important n-gram features at the beginning of the training process. By integrating our method into a classic CNN model for text classification (Kim, 2014), we observe significant im-

† Corresponding author.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1884–1889
Copenhagen, Denmark, September 7–11, 2017.

provements in sentiment analysis and topic classification tasks. The advantages of our approach are as follows:

- Features are directly extracted from training data without involving any external resources;
- The computation brought by our method is relatively small, resulting in small additional training costs;
- The filter initialization is task independent. It could be easily applied to other NLP tasks.

Also, we further analyze the filters, shedding some light on the mechanism how our method influences the training process. The source code is released at https://github.com/shenshen-hungry/Semantic-CNN.

2 Related Work

Most recently, CNNs are becoming increasingly popular in a variety of NLP tasks. An influential one is the work of (Kim, 2014), where a simple CNN with a single layer of convolution is used for feature extraction. Despite its simple structure, the model achieves strong baselines on many sentence classification datasets. Following this work, several improved models are proposed. Zhang and Wallace (2015) improve the model by optimizing hyper-parameters and provide a detailed analysis of the CNN (Kim, 2014). Yin and Schütze (2016) and Zhang et al. (2016b) exploit different pre-trained word embeddings (e.g. word2vec and GloVe) to enhance the model.

In addition to initializing embedding layers with pre-trained word vectors, other pre-designed features also prove to be very effective in assisting the training of neural models. For example, in (Hu et al., 2016), neural models are harnessed by logic rules. Li et al. (2016) propose to use pre-calculated words' weights to guide Paragraph Vector model. Dai and Le (2015) combine the hidden layers of RNNs with linearly increasing weights. Xie et al. (2016) use entity descriptions from knowledge bases (e.g. Freebase) to learn knowledge representations for entity classification and knowledge graph completion. Qian et al. (2016) propose linguistically regularized LSTMs for sentiment analysis with sentiment lexicons, negation words, and intensity words. In this work, we encode semantic features into convolutional layers by initializing them with important n-grams. Being aware of which n-grams are important, CNN is able to extract more discriminative features for text classification.

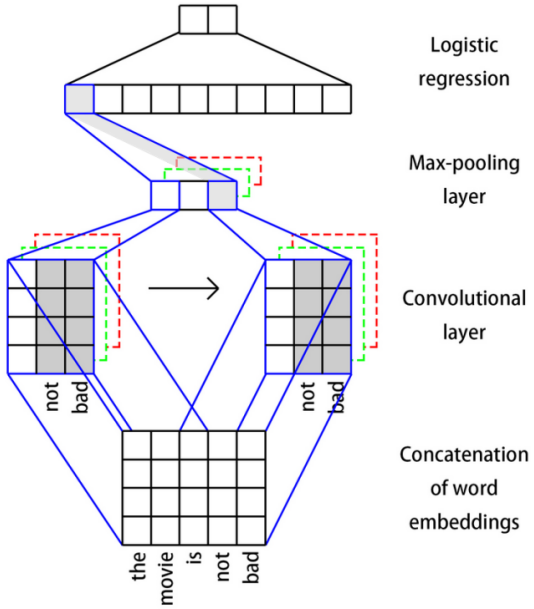

Figure 1: The framework of CNN with one layer of convolution and pooling.

3 Our Method

The intuition behind our method is simple: Since CNNs essentially capture semantic features of n-grams, we can use important n-grams to initialize the filters. As a result, the filters are able to focus on extracting those important n-gram features at the beginning of the training. As shown in Figure 1, we use embeddings of "not" and "bad" to initialize the filter. A larger score will be obtained when the "not bad" filter matches the bigram "not bad" in the text, otherwise a relatively smaller score will be returned.

3.1 N-gram Selection

Firstly, we extract important n-grams from the training data. Intuitively, n-gram "not bad" is much more important than "watch a movie" for determining reviews' polarities. Naive Bayes (NB) weighting is an effective technique for determining the words' importance (Martineau and Finin, 2009; Wang and Manning, 2012). NB weight $\mathbf{r}$ of a n-gram w in class c is calculated as follows:

$$\mathbf{r} = \frac{(\mathbf{p}_c^w + \alpha)/||\mathbf{p}_c||_1}{(\mathbf{p}_{\tilde{c}}^w + \alpha)/||\mathbf{p}_{\tilde{c}}||_1}$$

where $\mathbf{p}_c^w$ is the number of texts that contain

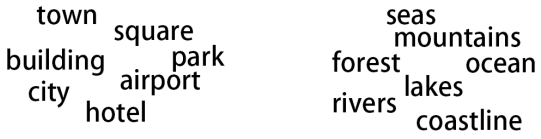

Figure 2: Uni-gram cluster examples.

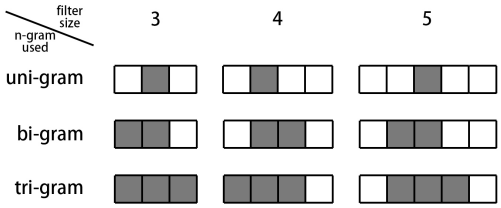

Figure 3: Filter initialization.

n-gram w in class c, $\mathbf{p}_{\tilde{c}}^{w}$ is the number of texts that contain n-gram w in other classes, $||\mathbf{p}_c||_1$is the number of texts in class c, $||\mathbf{p}_{\tilde{c}}||_1$ is the number of texts in other classes, α is a smoothing parameter. For *positive* class in movie review dataset, the ratios of n-grams like "amazing" and "not bad" should be large since they appear much more frequently in positive texts than in negative texts. For neutral n-grams like "of the" and "movie", their ratios should be around 1. For each class, we select the n-grams whose ratios are much higher than 1 for filter initialization. We give examples of n-grams selected by our method in Appendix.

3.2 Filter Initialization

We concatenate word embeddings to construct n-gram embeddings. For example, a tri-gram embedding has 3*100 dimensions when word embedding has 100 dimensions. This concatenation follows the mechanism of convolutional filters,where a filter with n*d dimensions is able to capture n-gram features (d is the dimension of word embedding). Because the number of filters in CNNs is much smaller than the number of n-grams, a filter tends to extract the features of a class of n-grams rather than an individual n-gram. Based on this observation, we don't use n-gram embeddings to initialize the filters directly. Instead, we firstly use K-means to cluster features of the selected n-grams, and then use the clusters' centroid vectors to initialize the filters. In this work, we consider clustering uni-gram (word), bi-gram and tri-gram features. Figure 2 shows two uni-gram cluster examples extracted from the location questions in TREC dataset (Li and Roth, 2002).

After obtaining the n-gram clusters, we feed their centroid vectors into the center of the filters. The remaining positions are still initialized randomly. Taking filters with size 3, 4, 5 as examples, Figure 3 shows how we fill uni, bi, and tri-gram features into the filters. By doing this, we encode semantic features into the filters. For example, in the TREC question classification task, the initialization will result in six types of filters which are sensitive to abbreviation, entity, description, human, location and number questions respectively.

4 Experiments

4.1 Datasets and Hyper-parameter Settings

CNN-non-static[1] (short for CNN) proposed by Kim (2014) is used as our baseline, which consists of one embedding layer, one convolutional layer, one max pooling layer, and one fully connected layer. The model proposed by Kim (2014) is a strong baseline in sentence classification. For details of the model, one can see (Kim, 2014; Zhang and Wallace, 2015). Pre-trained word embeddings on Google News via word2vec toolkit[2] are used for initializing the convolutional filters, besides initializing the embedding layer of CNN as in (Kim, 2014). For a fair comparison, we use the same seven datasets [3] and hyper-parameter setting with Kim (2014)'s work for training and testing. Uni, bi, and tri-gram features are used to initialize the filters. For a K-way classification problem, we select top *10%* n-grams in each class according to NB weighting. Since *300* filters are used in Kim (2014)'s work, we follow this setting and aggregate n-grams into *300/K* clusters for each class. Centroid vectors are used for filling the filters. Taking binary classification dataset MR as an example, *150* "positive" filters and *150* "negative" filters are obtained after initialization.

4.2 Effectiveness of Filter Initialization

In this section, we demonstrate the effectiveness of our initialization technique. We respectively use uni, bi and tri-gram centroid vectors to fill the filters. Table 1 lists the results. The CNN has provided very strong baselines. Our method

[1]The embedding layer in CNN-non-static is initialized with pre-trained vectors from word2vec toolkit and fine-tuned for each task.

[2]https://code.google.com/p/word2vec/

[3](MR (Pang and Lee, 2005), SST-1/2 (Socher et al., 2013), Subj (Pang and Lee, 2004), TREC (Li and Roth, 2002), CR (Hu and Liu, 2004), and MPQA (Wiebe et al., 2005))

Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-non-static	81.5	48.0	87.2	93.4	93.6	84.3	**89.5**
+UNI	82.1	**50.8**	**89.0**	93.7	94.4	**86.0**	89.3
+BI	**82.2**	50.7	88.3	93.7	**94.6**	85.8	**89.5**
+TRI	82.1	49.8	88.2	**93.8**	94.2	85.9	89.2

Table 1: Effectiveness of filter initialization.

Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-non-static (Kim, 2014)	81.5	48.0	87.2	93.4	93.6	84.3	**89.5**
MV-CNN (Yin and Schütze, 2016)	-	49.6	**89.4**	93.9	-	-	-
MGNC-CNN (Zhang et al., 2016b)	-	48.7	88.3	**94.1**	**95.5**	-	-
CNN-Rule (Hu et al., 2016)	81.7	-	89.3	-	-	85.3	-
Our Model (CNN-non-static+UNI)	**82.1**	**50.8**	89.0	93.7	94.4	**86.0**	89.3
combine-skip (Kiros et al., 2015)	76.5	-	-	93.6	92.2	80.1	87.1
Adasent (Zhao et al., 2015)	**83.1**	-	-	**95.5**	92.4	**86.3**	**93.3**
DSCNN (Zhang et al., 2016a)	82.2	50.6	**88.7**	93.9	**95.6**	-	-
PV (Le and Mikolov, 2014)	74.8	48.7	87.8	90.5	91.8	78.1	74.2
NBSVM (Wang and Manning, 2012)	79.4	-	-	93.2	-	81.8	86.3
Tree LSTM (Tai et al., 2015)	-	**51.0**	88.0	-	-	-	-

Table 2: Comparisons of state-of-the-arts.

further improves the accuracies significantly on all datasets except MPQA. The results are consistent with (Wang and Manning, 2012), where NB weighting produces little improvement over MPQA. We can also observe that the performance of uni, bi and tri-grams are comparable. None of them outperforms the others on all datasets.

4.3 Comparisons with State-of-the-arts

Table 2 lists the results of our model and other state-of-the-arts. Models in the first group are improved CNNs based on (Kim, 2014). Among them, MV-CNN and MGNC-CNN utilize multiple pre-trained embeddings as inputs, and CNN-Rule integrates logic rules. Our model achieves the best performance on three tasks without requiring any extra training costs and resources. With this simple initialization method, our model also gives competitive results against more sophisticated deep learning models in the second group, e.g. Adasent (Zhao et al., 2015) and DSCNN (Zhang et al., 2016a) that have complex structures or use the combinations of NNs.

Experiments show that our n-gram features make great contribution to both two-class and multi-class classification. Essentially, our method enables CNNs to obtain better generalization abilities. Furthermore, as the initialization does not rely on any external prior knowledge or resources, it could be easily applied to other NLP tasks or other languages.

	positive filters		negative filters	
	+	-	+	-
UNI	29.5	20.5	21.3	28.7
BI	31.3	18.7	18.1	31.9
TRI	31.4	18.6	17.2	32.8

Table 3: "+" and "-" are used to denote the number of positive and negative weights respectively. The data in the table are obtained from MR by the average of 10 times training. Every time we select 100 filters. 50 of them are initialized with positive n-grams and the rest are with negative n-grams.

4.4 Further Analysis of Filters

We further analyze the filter initialization with an example of binary sentiment classification. Through the initialization we have determined which filters are positive or negative in advance. The corresponding neurons of positive filters upon max-pooling layer are supposed to be activated by positive samples. Since positive (negative) samples have labels of 1 (0), the corresponding weights (in logistic regression) of those "positive" neurons tend to be positive. For the same reason, the negative filters tend to have negative weights. The results shown in Table 3 confirm our hypothesis: Positive/negative filters respectively tend to have positive(+)/negative(-) weights. The difference between positive and negative filters are more obvious in bi-gram and tri-gram cases. It is because bi and tri-gram centroid vectors could initialize more parameters of filters than uni-gram.

In Table 1, experiments show that different

choices of uni, bi, and tri-grams have little influence on the results. The following is our assumption: Compared to uni-grams (words), bi and tri-grams can cover more spaces of filters and introduce more NB information to filters. Filters initialized by them thus pay more attention to NB information than filters initialized by uni-grams according to Table 3. However, bi and tri-grams are also sparser in data than uni-grams. Their NB weights are not as accurate as those of uni-grams, even applied smoothing. As NB weight of a n-gram denotes its contribution to the classification, model initialized with tri-grams does not always perform the best.

5 Conclusion

This paper proposes a novel weight initialization technique for CNNs. We discover that convolutional filters that encode semantic features at the beginning of the training tend to produce better results than being randomly initialized. This has a similar effect with embedding layer initialization via pre-trained word vectors. Experimental results demonstrate the effectiveness of the initialization technique on multiple text classification tasks. In addition, our method requires few external resources and relatively small calculation, making it attractive for scenarios where training costs may be an issue.

In textual data, the features extracted by CNNs are n-grams. However, in fields like computer vision, features extracted by filters are more difficult to interpret. It still requires further exploration to apply our method to fields beyond NLP.

Acknowledgments

This work is supported by the National High Technology Research and Development Program of China (No. 2012AA011104), National Language Committee Research Program of China (No. YB125-124), and National Natural Science Foundation of China (No. 61472428).

References

Yoshua Bengio et al. 2009. Learning deep architectures for ai. *Foundations and trends® in Machine Learning* 2(1):1–127.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *EMNLP*. pages 740–750.

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised sequence learning. In *Proceedings of NIPS 2015*. pages 3079–3087.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*. volume 9, pages 249–256.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural Computation* 18(7):1527–1554.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of ACM SIGKDD 2004*. ACM, pages 168–177.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard H. Hovy, and Eric P. Xing. 2016. Harnessing deep neural networks with logic rules. In *Proceedings of ACL 2016*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of EMNLP 2014*. pages 1746–1751.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Proceedings of NIPS 2015*. pages 3294–3302.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. pages 1097–1105.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of ICML 2014*. pages 1188–1196.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521(7553):436–444.

Bofang Li, Zhe Zhao, Tao Liu, Puwei Wang, and Xiaoyong Du. 2016. Weighted neural bag-of-n-grams model: New baselines for text classification. In *Proceedings of COLING 2016*. pages 1591–1600.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of ACL 2002*. Association for Computational Linguistics, pages 1–7.

Justin Martineau and Tim Finin. 2009. Delta TFIDF: an improved feature space for sentiment analysis. In *Proceedings of ICWSM 2009*.

Dmytro Mishkin and Jiri Matas. 2015. All you need is a good init. *CoRR* abs/1511.06422.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of ACL 2004*. Association for Computational Linguistics, page 271.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL 2005*. Association for Computational Linguistics, pages 115–124.

Qiao Qian, Minlie Huang, and Xiaoyan Zhu. 2016. Linguistically regularized lstms for sentiment classification. *arXiv preprint arXiv:1611.03949* .

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. 2013. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *CoRR* abs/1312.6120.

Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, Christopher Potts, et al. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP 2013*. Citeseer, volume 1631, page 1642.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1):1929–1958.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* .

Sida I. Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of ACL 2012, Volume 2: Short Papers*. pages 90–94.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation* 39(2):165–210.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *AAAI*. pages 2659–2665.

Wenpeng Yin and Hinrich Schütze. 2016. Multichannel variable-size convolution for sentence classification. *CoRR* abs/1603.04513.

Rui Zhang, Honglak Lee, and Dragomir R. Radev. 2016a. Dependency sensitive convolutional neural networks for modeling sentences and documents. In *Proceedings of NAACL 2016*. pages 1512–1521.

Ye Zhang, Stephen Roller, and Byron C. Wallace. 2016b. MGNC-CNN: A simple approach to exploiting multiple word embeddings for sentence classification. *CoRR* abs/1603.00968.

Ye Zhang and Byron C. Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *CoRR* abs/1510.03820.

Han Zhao, Zhengdong Lu, and Pascal Poupart. 2015. Self-adaptive hierarchical sentence model. In *Proceedings of IJCAI 2015*. pages 4069–4076.