PoD: Positional Dependency-Based Word Embedding for Aspect Term Extraction

Yichun Yin[1], **Chenguang Wang**[2], **Ming Zhang**[3*]
[1] Noah's Ark Lab, Huawei
[2] Computer Science Division, UC Berkeley
[3] Department of Computer Science, Peking University
yinyichun@huawei.com, chenguangwang@berkeley.edu
mzhang_cs@pku.edu.cn

Abstract

Dependency context-based word embedding jointly learns the representations of word and dependency context, and has been proved effective in aspect term extraction. In this paper, we design the positional dependency-based word embedding (PoD) which considers both dependency context and positional context for aspect term extraction. Specifically, the positional context is modeled via relative position encoding. Besides, we enhance the dependency context by integrating more lexical information (e.g., POS tags) along dependency paths. Experiments on SemEval 2014/2015/2016 datasets show that our approach outperforms other embedding methods in aspect term extraction.

1 Introduction

Aspect term extraction aims to extract expressions that represent properties of products or services from online reviews (Hu and Liu, 2004a; Hu and Liu, 2004b; Popescu and Etzioni, 2007; Liu, 2010). Understanding the context between words in reviews, such as through conditional random fields (Pontiki et al., 2014; Pontiki et al., 2015; Pontiki et al., 2016), is the key to superior results in aspect term extraction. Word embeddings are effective to capture the contextual information across a wide range of NLP tasks (Tai et al., 2015; Lei et al., 2015; Bojanowski et al., 2017; Devlin et al., 2019). However, they only produce moderate results in aspect term extraction. Recent studies (e.g., Yin et al. (2016)) indicate that this is due to the distributed nature of the word embedding (Mikolov et al., 2013b), which ignores the rich context between the words, such as syntactic information.

In this paper, we propose positional dependency-based word embedding (PoD) to enhance the context modeling capability for aspect term extraction. PoD explicitly captures two types of contexts, *dependency context* and *positional context*. Inspired by the simple-yet-effective position encoding in Transformer (Vaswani et al., 2017), PoD models the positional context via relative position encoding (Shaw et al., 2018) between words within a fixed window. Besides, the dependency context is defined as the dependency path as well as the attached lexical information (e.g., POS tags and words) along the path. Moreover, PoD is able to incorporate more lexical information into the semantic compositional model via the dependency context, making representations of dependency paths more informative than the ones that only consider grammatical information (Yin et al., 2016). We then linearly combine the dependency and positional context to produce the positional dependencies among words. We also define a margin-based ranking loss to efficiently optimize PoD.

Our contributions are two-fold, (i) we propose positional dependency-based word embedding PoD, which incorporates both positional context and dependency context, (ii) we compare PoD with existing aspect term extraction methods and demonstrate that PoD yields improved results on aspect term extraction datasets.

[*] Corresponding author.

Proceedings of the 28th International Conference on Computational Linguistics, pages 1714–1719
Barcelona, Spain (Online), December 8-13, 2020

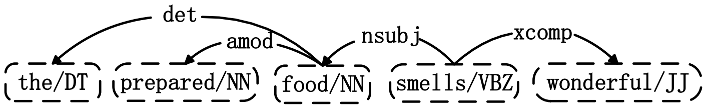

Figure 1: An example sentence, parsed by Stanford CoreNLP.

Target	Context	DC	PC
food	the	$* \xrightarrow{det} *$	-2
	prepared	$* \xrightarrow{amod} *$	-1
	smells	$* \xleftarrow{nsubj} *$	1
	wonderful	$* \xleftarrow{nsubj}$ smells/VBZ $\xrightarrow{xcomp} *$	2

Table 1: Target word, context words and their corresponding contexts: DC refers to dependency context and PC refers to positional context.

2 Positional Dependency-Based Word Embedding

2.1 Model Description

PoD aims to maximize likelihoods of triples ($\mathrm{w_t}$, c, $\mathrm{w_c}$), where $\mathrm{w_t}$ and $\mathrm{w_c}$ represent target word and context word respectively, c refers to positional dependency-based context (an example is in Table 1), which consists of two types of contexts: the dependency context (dependency paths between target and context word) and positional context (relative position encoding between target and context word). Figure 1 illustrates the sentence example according to the triples in Table 1.

We introduce two score functions for triples ($\mathrm{w_t}$, c, $\mathrm{w_c}$) which are as follows.

$$S_{add} = (\mathbf{w}_c + \mathbf{c}) \cdot \mathbf{w}_t^\intercal; S_{puct} = (\mathbf{w}_c \circ \mathbf{c}) \cdot \mathbf{w}_t^\intercal, \tag{1}$$

where S_{add} uses the element-wise addition for the context word and its context c, while S_{puct} uses the element-wise product. We use two embedding matrices $\mathbf{M}_t \in R^{|V| \times d}$ and $\mathbf{M}_c \in R^{|V| \times d}$ to represent target words and context words respectively, where $|V|$ is the size of vocabulary and d is the dimension of embeddings. The $\mathbf{w}_c \in R^{1 \times d}$ and $\mathbf{w}_t \in R^{1 \times d}$ are obtained through lookup operations. Note that we describe how to derive **c** in Section 2.2.

2.2 Positional Dependency

We construct the positional dependency-based context **c** by linearly combining the dependency context vector $\mathbf{c}_{dep}$ derived from semantic composition of lexical dependency paths and the positional context vector $\mathbf{c}_{pos}$ computed based on relative position encoding (Shaw et al., 2018). The representation of positional dependency-based context is defined in Eq. (2).

$$\mathbf{c} = \alpha \cdot \mathbf{c}_{pos} + (1 - \alpha) \cdot \mathbf{c}_{dep}, \tag{2}$$

where α is used to trade-off the effects between dependency and positional contexts in the model.

The basic idea of using relative position encoding is based on the assumption that context words with different relative positions have different impacts on learning the representations of target words. The use of relative position encoding has been proved to be useful in supervised relation classification (Zeng et al., 2014) and machine translation (Vaswani et al., 2017; Shaw et al., 2018). Similar to using embeddings to represent words, we also introduce $\mathbf{M}_l \in R^{(s-1) \times d}$ to represent the relative position encoding and derive $\mathbf{c}_{pos}$ from it, where s is the window size.

We also consider the lexical information along dependency paths when learning the representations of the dependency context. For example, for the pair (food, wonderful) in Figure 1, the corresponding dependency path is $* \xleftarrow{nsubj}$ smells/VBZ $\xrightarrow{xcomp} *$. We denote the words, POS tags as the lexical information, and use $dep = \{g_1, g_2, ..., g_{|c|}\}$ to denote the composite lexical dependency path. The embedding matrix $\mathbf{M}_{dep} \in R^{n \times d}$ is utilized to derive the distributed representations of lexical dependency path $\{\mathbf{g}_1, \mathbf{g}_2, ..., \mathbf{g}_{|c|}\}$, where n is the size of dictionary including words, POS tags and dependency paths.

To obtain $\mathbf{c}_{dep}$, we use RNN model which learns the dependency path representations along the sequence dep in a recurrent manner.

2.3 Model Optimization

We use a margin-based ranking objective to learn model parameters in Eq. (1), which encourages scores of positive triples $(\mathrm{w_t}, \mathrm{c}, \mathrm{w_c}) \in \mathcal{T}$ to be higher than scores of sampled triples $(\mathrm{w'_t}, \mathrm{c}, \mathrm{w_c}) \in \mathcal{T}'$. The ranking loss is as follows.

$$L = \sum_{(\mathrm{w_t},\mathrm{c},\mathrm{w_c})\in\mathcal{T}} \sum_{(\mathrm{w'_t},\mathrm{c},\mathrm{w_c})\in\mathcal{T}'} max\{S(\mathrm{w_t}, \mathrm{c}, \mathrm{w_c}) - S(\mathrm{w'_t}, \mathrm{c}, \mathrm{w_c}) + \delta, 0\}, \tag{3}$$

where δ is the margin value, $S(*)$ is the score function defined in Eq. (1), in which $\mathbf{c}$ is introduced in Eq. (2).

Note that, the proposed Eq. (3) conducts negative sampling on target words rather than dependency paths, which proposes two advantages, (i) it can exploit arbitrary hop dependency paths. Besides, the words and POS tags along the path can be utilized; (ii) it avoids to memorize dependency path frequencies which grow exponentially with the number of hops.

The negative sampling method is employed to train the embedding model (Eq. (1)). These randomly chosen words in $\mathcal{T}'$ are sampled based on the marginal distribution $p(w)$ and $p(w)$ is estimated from the word frequency raised to the $\frac{3}{4}$ power (Mikolov et al., 2013a) in the corpus. We set the negative number to 15 which is a trade-off between the training time and performance. The δ is empirically set to 1 according to (Collobert and Weston, 2008; Bollegala et al., 2015). To avoid the overfitting in RNN, we employ dropout on the input vectors and set the dropout rate to 0.5. The asynchronous gradient descent is used for parallel training. Moreover, Adagrad (Duchi et al., 2011) is used to adaptively change learning rate and the initial learning rate is set to 0.1.

3 Experiment

3.1 Dataset

We evaluate POD on aspect term extraction benchmark datasets: SemEval 2014/2015/2016. The SemEval 2014 datasets include two domains: laptop and restaurant, and we use the D1 and D2 to denote these two datasets respectively. The SemEval 2015/2016 datasets only include restaurant domain. D3 and D4 are utilized to represent them. We use the corpora introduced in (Yin et al., 2016) to learn the distributed representations of words and lexical dependency paths.

3.2 Baseline and Setting

We compare POD with top systems in SemEval with method class *Top system* as shown in Table 1. We also compare our method with notable embedding-based methods with method class *Embedding method* illustrated in Table 1.

In order to choose l, d (Section 2.1) and α (Eq. (2)), 80% sentences in training data are used as training set, and the rest 20% are used as development set. The dimensions of word and dependency path embeddings are set as 100. Larger dimensions get similar results in the development set but cost more time. l is set as 10 which performs best in the development set. Similarly, the αs are set as 0.7, 0.5, 0.5 and 0.5 for datasets D1, D2, D3 and D4 respectively.

To make fair comparisons, we choose parameters l and d on the development set for embedding baselines. All the dimensions of embedding methods are set as 100. The dimensions l in Skip-gram, CBOW and WDEmb models are set as 15, the dimensions in Glove and DepEmb are set as 10. The windows of Skip-gram, CBOW and Glove are set as 5, which are the same as our model. As derived embeddings are not necessarily in a bounded range (Turian et al., 2010), this might lead to moderate results. We apply a simple function of discretization following (Yin et al., 2016) to make embedding features more effective.

3.3 Result and Analysis

The results are described in Table 2 and the t-test is also conducted by random initialization. From the table, we find that POD with both S_{puct} and S_{add} consistently outperform WDEmb which is one of the

Method	Method Class	D1	D2	D3	D4
IHS_RD (Chernyshevich, 2014)	Top system in D1	**74.55**	79.62	-	-
DLIREC (Zhiqiang and Wenting, 2014)	Top system in D2	73.78	84.01	-	-
EliXa (San Vicente et al., 2015)	Top system in D3	-	-	70.04	-
Nlangp (Toh and Su, 2016)	Top system in D4	-	-	-	**72.34**
DRNLM (Mirowski and Vlachos, 2015)	Embedding method	66.91	78.59	64.75	63.89
Skip-gram (Mikolov et al., 2013b)	Embedding method	70.52	82.20	66.98	68.57
CBOW (Mikolov et al., 2013a)	Embedding method	69.80	81.98	67.09	67.43
Glove (Pennington et al., 2014)	Embedding method	67.23	80.69	64.12	64.39
DepEmb (Levy and Goldberg, 2014)	Embedding method	71.02	82.78	67.55	69.23
WDEmb (Yin et al., 2016)	Embedding method	73.72	83.52	68.27	70.20
Ours-POD (S_{add})	Embedding method	73.54^*	$84.21^\dagger$	69.14^*	$70.90^\dagger$
Ours-POD (S_{puct})	Embedding method	74.07^*	$\mathbf{84.82}^*$	$\mathbf{70.18}^\dagger$	71.70^*

Table 2: Comparison of F1 scores on the SemEval 2014/2015/2016 datasets. In t-tests, the marker * refers to p-value < 0.05, the marker † refers to p-value < 0.01, and WDEmb (Yin et al., 2016) is the compared method.

Information	D1	D2	D3	D4
Dependency path	72.13	83.52	68.39	70.90
+ POS tags (only)	72.48	83.87	69.03	71.02
+ Words (only)	73.79	84.31	69.98	71.24
+ POS tags + Words	**74.07**	**84.82**	**70.18**	**71.70**

Table 3: Effects of information in dependency context.

best embedding methods. The reasons are that (i) our model incorporates positional context as relative position encoding to help enhance word embeddings; (ii) the dependency context leverages the lexical dependency path capturing more specific lexical information such as words and POS tags (extracted using Stanford CoreNLP (Manning et al., 2014)) than WDEmb. POD also achieves comparable results with top systems which are based on hand-crafted features in all datasets, which shows that our learned embeddings are effective for aspect term extraction. The S_{puct} performs better than S_{add}, which indicates that the product-based composition method is more capable in capturing the useful features in aspect term extraction. In terms of embedding-based baselines, DepEmb and WDEmb perform better than other baselines, which indicates that encoding syntactic knowledge into word embeddings is desirable for aspect term extraction.

We also analyze the effects of POS tags and words along dependency paths in the dependency context on final results. The results are presented in Table 3. From the table, we observe that both POS tags and words along dependency paths boost aspect term extraction, which indicates that lexical information can encode discriminative information for representations of dependency paths. Meanwhile, POD obtains better results by adding both POS tags and words.

4 Related Work

Association rule mining is used in (Hu and Liu, 2004b) to mine aspect terms. Opinion words are used to extract infrequent aspect terms. The relationship between opinion words and aspect words is crucial to extract aspect terms, which are deployed in many follow-up studies. In (Qiu et al., 2011), the predefined dependency paths are utilized to iteratively extract aspect terms and opinion words. POD instead learns the representation of the dependency context.

Dependency-based word embedding (Levy and Goldberg, 2014; Komninos and Manandhar, 2016) encodes dependencies into word embeddings, and has been shown effective in aspect term extraction as well (Yin et al., 2016). However, only grammatical information is considered among the dependency paths. We instead introduce a positional dependency-based embedding method which considers both dependency context and positional context. End-to-end aspect term extraction (Wang et al., 2016; Wang et al., 2017; Li et al., 2018; Xu et al., 2018) based on neural networks and attention mechanism, have been recently developed. Compare to these methods, POD is an embedding method, can thus be applied to more applications. Compare to deep word representations (Peters et al., 2018; Devlin et al., 2019), POD is more efficient, which is crucial to aspect term extraction.

5 Conclusion

In this paper, we develop a specific word embedding method for aspect term extraction. Our method considers both positional and dependency context when learning the word embedding. Meanwhile, the lexical information along dependency path is encoded into representations of dependency context. Compared with other embedding methods, our method achieves better results in aspect term extraction.

6 Acknowledgement

This paper is supported by National Key Research and Development Program of China with Grant No. 2018AAA0101900 / 2018AAA0101902 as well as the National Natural Science Foundation of China (NSFC Grant No. 61772039 and No. 91646202). Chenguang Wang is supported by Berkeley DeepDrive and Berkeley Artificial Intelligence Research.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL*, 5:135–146.

Danushka Bollegala, Takanori Maehara, and Ken-ichi Kawarabayashi. 2015. Unsupervised cross-domain word representation learning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 730–740, July.

Maryna Chernyshevich. 2014. Ihs r&d belarus: Cross-domain extraction of product features using conditional random fields. *SemEval 2014*, page 309.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*, pages 160–167.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

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

Minqing Hu and Bing Liu. 2004b. Mining opinion features in customer reviews. In *AAAI*, volume 4, pages 755–760.

Alexandros Komninos and Suresh Manandhar. 2016. Dependency based embeddings for sentence classification tasks. In *NAACL*, pages 1490–1500.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2015. Molding cnns for text: non-linear, non-consecutive convolutions. In *EMNLP*, pages 1565–1575, September.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *ACL*, pages 302–308.

Xin Li, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018. Aspect term extraction with history attention and selective transformation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4194–4200. International Joint Conferences on Artificial Intelligence Organization, 7.

Bing Liu. 2010. Sentiment analysis and subjectivity. In *Handbook of Natural Language Processing, Second Edition.*, pages 627–666.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL*, pages 55–60.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119.

Piotr Mirowski and Andreas Vlachos. 2015. Dependency recurrent neural language models for sentence completion. *arXiv preprint arXiv:1507.01193*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543, Doha, Qatar, October.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*, pages 2227–2237.

Maria Pontiki, Haris Papageorgiou, Dimitrios Galanis, Ion Androutsopoulos, John Pavlopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *SemEval 2014*, pages 27–35.

Maria Pontiki, Dimitrios Galanis, Haris Papageogiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Denver, Colorado*.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphee De Clercq, Veronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Núria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, June.

Ana-Maria Popescu and Orena Etzioni. 2007. Extracting product features and opinions from reviews. In *Natural language processing and text mining*, pages 9–28.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational linguistics*, 37(1):9–27.

Iñaki San Vicente, Xabier Saralegi, and Rodrigo Agerri. 2015. Elixa: A modular and flexible absa platform. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 748–752, June.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *NAACL-HLT*, pages 464–468.

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 (Volume 1: Long Papers)*, pages 1556–1566, July.

Zhiqiang Toh and Jian Su. 2016. Nlangp at semeval-2016 task 5: Improving aspect based sentiment analysis using neural network features. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 287–293, June.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *ACL*, pages 384–394.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. In *EMNLP*, pages 616–626.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *AAAI*, pages 3316–3322.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. In *ACL*.

Yichun Yin, Furu Wei, Li Dong, Kaimeng Xu, Ming Zhang, and Ming Zhou. 2016. Unsupervised word and dependency path embeddings for aspect term extraction. *IJCAI*.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation classification via convolutional deep neural network. In *COLING*, pages 2335–2344.

Toh Zhiqiang and Wang Wenting. 2014. Dlirec: Aspect term extraction and term polarity classification system.