Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling

Kun Xu[1], **Yansong Feng**[1], **Songfang Huang**[2], **Dongyan Zhao**[1]
[1]Institute of Computer Science & Technology, Peking University, Beijing, China
[2]IBM China Research Lab, Beijing, China
{xukun, fengyansong, zhaodongyan}@pku.edu.cn, huangsf@cn.ibm.com

Abstract

Syntactic features play an essential role in identifying relationship in a sentence. Previous neural network models often suffer from irrelevant information introduced when subjects and objects are in a long distance. In this paper, we propose to learn more robust relation representations from the shortest dependency path through a convolution neural network. We further propose a straightforward negative sampling strategy to improve the assignment of subjects and objects. Experimental results show that our method outperforms the state-of-the-art methods on the SemEval-2010 Task 8 dataset.

1 Introduction

The relation extraction (RE) task can be defined as follows: given a sentence S with a pair of nominals e_1 and e_2, we aim to identify the relationship between e_1 and e_2. RE is typically investigated in a classification style, where many features have been proposed, e.g., Hendrickx et al. (2010) designed 16 types of features including POS, WordNet, FrameNet, dependency parse features, etc. Among them, syntactic features are considered to bring significant improvements in extraction accuracy (Bunescu and Mooney, 2005a). Earlier attempts to encode syntactic information are mainly kernel-based methods, such as the convolution tree kernel (Qian et al., 2008), subsequence kernel (Bunescu and Mooney, 2005b), and dependency tree kernel (Bunescu and Mooney, 2005a).

With the recent success of neural networks in NLP, different neural network models are proposed to learn syntactic features from raw sequences of words or constituent parse trees(Zeng et al., 2014; Socher et al., 2012), which have been proved effective, but, often suffer from irrelevant subsequences or clauses, especially when subjects and objects are in a longer distance. For example, in the sentence, “The [singer]$_{e1}$, who performed three of the nominated songs, also caused a [commotion]$_{e2}$ on the red carpet”, the *who* clause is used to modify subject e_1, but is unrelated to the *Cause-Effect* relationship between *singer* and *commotion*. Incorporating such information into the model will hurt the extraction performance. We therefore propose to learn a more robust relation representation from a convolution neural network model that works on the simple dependency path between subjects and objects, which naturally characterizes the relationship between two nominals and avoids negative effects from other irrelevant chunks or clauses.

Our second contribution is the introduction of a negative sampling strategy into the CNN models to address the relation directionality, i.e., properly assigning the subject and object within a relationship. In the above *singer* example, (*singer*, *commotion*) hold the *Cause-Effect* relation, while (*commotion*, *singer*) not. Previous works do not fully investigate the differences between subjects and objects in the utterance, and simply transform a (K+1)-relation task into a (2×K+1) classification task, where 1 is the *other* relation. Interestingly, we find that dependency paths naturally offer the relative positions of subjects and objects through the path directions. In this paper, we propose to model the relation directionality by exploiting the dependency path to learn the assignments of subjects and objects using a straightforward negative sampling method, which adopts the shortest dependency path from the object to the subject as a negative sample. Experimental results show that the negative sampling method significantly improves the performance, and our model outperforms the-state-of-the-art methods on the SemEval-2010 Task 8 dataset.

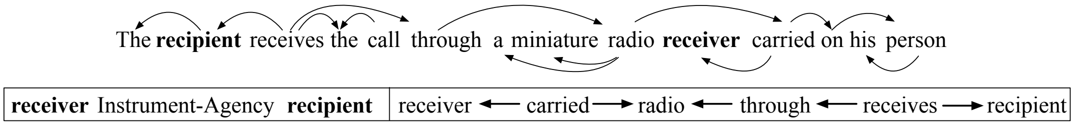
The recipient receives the call through a miniature radio receiver carried on his person
receiver Instrument-Agency recipient
receiver ← carried → radio ← through ← receives → recipient

Figure 1: The shortest dependency path representation for an example sentence from SemEval-08.

2 The Shortest Path Hypothesis

If e_1 and e_2 are two nominals mentioned in the same sentence, we assume that the shortest path between e_1 and e_2 describes their relationship. This is because (1) if e_1 and e_2 are arguments of the same predicate, then their shortest path should pass through that predicate; (2) if e_1 and e_2 belong to different predicate-argument structures, their shortest path will pass through a sequence of predicates, and any consecutive predicates will share a common argument. Note that, the order of the predicates on the path indicates the proper assignments of subjects and objects for that relation. For example, in Figure 1, the dependency path consecutively passes through *carried* and *receives*, which together implies that in the *Instrument-Agency* relation, the subject and object play a sender and receiver role, respectively.

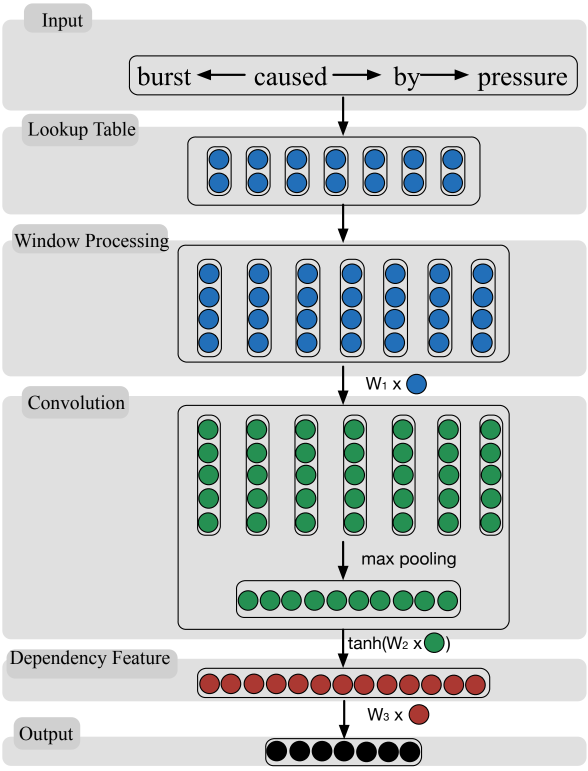

Figure 2: Architecture of the convolution neural network.

3 A Convolutional Neural Network Model

Our model successively takes the shortest dependency path (i.e, the words, dependency edge directions, and dependency labels) from the subject to the object as input, passes it through the lookup table layer, produces local features around each node on the dependency path, and combines these features into a global feature vector that are then fed to a softmax classifier. Each dimension of the output vector indicates the confidence score of the corresponding relation.

In the *lookup table* step, each node (i.e. word, label or arrow) in the dependency path is transformed into a vector by looking up the embedding matrix $W_e \in \mathbb{R}^{d \times |\mathbb{V}|}$, where d is the dimension of a vector and $\mathbb{V}$ is a set of all nodes we consider.

Convolution To capture the local features around each node of the dependency path, we consider a fixed size window of nodes around each node in the *window processing* component, producing a matrix of node features of fixed size $d_w \times 1$, where $d_w = d \times w$ and w is the window size. This matrix can be built by concatenating the vectors of nodes within the window.

In the convolutional layer, we use a linear transformation $W_1 \in \mathbb{R}^{n_1 \times d_w}$ to extract local features around each window of the given sequence, where n_1 is the size of hidden layer 1. The resulting matrix Z has size of $n_1 \times t$, where t is the number of nodes in the input dependency path.

We can see that Z captures local contextual information in the dependency path. Therefore, we perform a max pooling over Z to produce a global feature vector in order to capture the most useful local features produced by the convolutional layer (Collobert et al., 2011), which has a fixed size of n_1, independent of the dependency path length.

Dependency based Relation Representation To extract more meaningful features, we choose hyperbolic tanh as the non-linearity function in the second hidden layer, which has the advantage of being slightly cheaper to compute, while leaving

Train Strategy	Test Strategy	F1(%)
Blind	Blind	79.3
Sighted	Blind	81.3
Sighted	Sighted	89.2

Table 1: Performances on the development set with different train and testing strategies.

the generalization performance unchanged. $W_2 \in \mathbb{R}^{n_2 \times n_1}$ is the linear transformation matrix, where n_2 is the size of hidden layer 2. The output vector can be considered as higher level syntactic features, which is then fed to a softmax classifier.

Objective Function and Learning The softmax classifier is used to predict a K-class distribution $d(x)$, where K is the size of all possible relation types, and the transformation matrix is $W_3 \in \mathbb{R}^{K \times n_2}$. We denote $t(x) \in \mathbb{R}^{K \times 1}$ as the target distribution vector[1]: the entry $t_k(x)$ is the probability that the dependency path describes the k-th relation. We compute the cross entropy error between $t(x)$ and $d(x)$, and further define the objective function over all training data:

$$J(\theta) = -\sum_{x}\sum_{k=1}^{K} t_k(x) \log d_k(x) + \lambda||\theta||^2$$

where $\theta = (W_e, W_1, W_2, W_3)$ is the set of model parameters to be learned, and λ is a vector of regularization parameters. The model parameters θ can be efficiently computed via backpropagation through network structures. To minimize $J(\theta)$, we apply stochastic gradient descent (SGD) with AdaGrad (Duchi et al., 2011) in our experiments[2].

4 Negative Sampling

We start by presenting three pilot experiments on the development set. In the first one, we assume that the assignment of the subject and object for a relation is not given (blind), we simply extract features from e_1 to e_2, and test it in a blind setting as well. In the second one, we assume that the assignment is given (sighted) during training, but still blind in the test phase. The last one is assumed to give the assignment during both training and test steps. The results are listed in Table 1.

The third experiment can be seen as an upper bound, where we do not need to worry about the assignments of subjects and objects. By comparing the first and the second one, we can see that when adding assignment information during training, our model can be significantly improved, indicating that our dependency based representation can be used to learn the assignments of subjects/objects, and injecting better understandings of such assignments during training is crucial to the performance. We admit that models with more complex structures can better handle these considerations. However, we find that this can be achieved by simply feeding typical negative samples to the model and let the model learn from such negative examples to correctly choose the right assignments of subjects and objects. In practice, we can treat the opposite assignments of subjects and the objects as negative examples. Note that, the dependency path of the wrong assignment is different from that of the correct assignment, which essentially offers the information for the model to learn to distinguish the subject and the object.

5 Experimental Evaluation

We evaluate our model on the SemEval-2010 Task 8 (Hendrickx et al., 2010), which contains 10,717 annotated examples, including 8,000 instances for training and 2,717 for test. We randomly sampled 2,182 samples from the training data for validation.

Given a sentence, we first find the shortest dependency path connecting two marked nominals, resulting in two dependency paths corresponding to two opposite subject/object directions, and then make predictions for the two paths, respectively. We choose the relation *other* **if and only if** both predictions are *other*. And for the rest cases, we choose the non-*other* relation with highest confidence as the output, since ideally, for a non-*other* instance, our model will output the correct label for the right subject/object direction and an *other* label for the wrong direction. We evaluate our models by macro-averaged F1 using the official evaluation script.

We initialized W_e with 50-dimensional word vectors trained by Turian et al. (2010). We tuned the hyper parameters using the development set for each experimental setting. The hyper parameters include w, n_1, n_2, and regularization parameters for W_e, W_1, W_2 and W_3. The best setting was obtained with the values: 3, 200, 100, 10^{-4}, 10^{-3}, 10^{-4} and 2×10^{-3}, respectively.

[1] Note that, there may be more than one relation existing between two nominals. A dependency path thus may correspond to multiple relations.

[2] We omit detailed formulas for the limitation of space.

Method	Feature Sets	F1
SVM	16 types of features	82.2
RNN	-	74.8
	+POS, NER, WordNet	77.6
MVRNN	-	79.1
	+POS, NER, WordNet	82.4
CNN	-	78.9
(Zeng et al., 2014)	+WordNet,words around nominals	82.7
depCNN	-	81.3
depLCNN	-	81.9
depLCNN	+WordNet,words around nominals	83.7
depLCNN+NS	-	**84.0**
	+WordNet,words around nominals	**85.6**

Table 2: Comparisons of our models with other methods on the SemEval 2010 task 8.

Negative sampling schemes	F1
No negative examples	81.3
Randomly sampled negative examples from NYT	83.5
Dependency paths from the object to subject	**85.4**

Table 3: Comparisons of different negtive sampling methods on the development set.

Results and Discussion Table 2 summarizes the performances of our model, depLCNN+NS(+), and state-of-the-art models, SVM(Hendrickx et al., 2010), RNN, MV-RNN(Socher et al., 2012), and CNN(Zeng et al., 2014). For fair comparisons, we also add two types of lexical features, WordNet hypernyms and words around nominals, as part of input vector to the final `softmax` layer.

We can see that our vanilla depLCNN+NS, without extra lexical features, still outperforms, by a large margin, previously reported best systems, MVRNN+ and CNN+, both of which have taken extra lexical features into account, showing that our treatment to dependency path can learn a robust and effective relation representation. When augmented with similar lexical features, our depLCNN+NS further improves by 1.6%, significantly better than any other systems.

Let us first see the comparisons among plain versions of depLCNN (taking both dependency directions and labels into account), depCNN (considering the directions of dependency edges only), MVRNN and CNN, which all work in a $2\times K+1$ fashion. We can see that the both of our depCNN and depLCNN outperforms MVRNN and CNN by at least 2.2%, indicating that our treatment is better than previous conventions in capturing syntactic structures for relation extraction. And note that depLCNN, with extra considerations for dependency labels, performs even better than depCNN, showing that dependency labels offer more discriminative information that benefits the relation extraction task.

And when we compare plain depLCNN and depLCNN+NS (without lexical features), we can see that our Negative Sampling strategy brings an improvement of 2.1% in F1. When both of the two models are augmented with extra lexical features, our NS strategy still gives an improvement of 1.9%. These comparisons further show that our NS strategy can drive our model to learn proper assignments of subjects/objects for a relation.

Next, we will have a close look at the effect of our Negative Sampling method. We conduct additional experiments on the development set to compare two different negative sampling methods. As a baseline, we randomly sampled 8,000 negative examples from the NYT dataset (Chen et al., 2014). For our proposed NS, we create a negative example from each *non-other* instance in the training set, 6,586 in total. As shown in Table 2, it is no doubt that introducing more negative examples improves the performances. We can see that our model still benefits from the randomly sampled negative examples, which may help our model learn to refine the margin between the positive and negative examples. However, with similar amount of negative examples, treating the reversed dependency paths from objects to subjects as negative examples can achieve a better performance (85.4% F1), improving random samples by 1.9%. This again proves that dependency paths provide useful clues to reveal the assignments of subjects and objects, and a model can learn from such reversed paths as negative examples to make correct assignments. Beyond the relation extraction task, we believed the proposed Negative Sampling method has the potential to benefit other NLP tasks, which we leave for future work.

6 Conclusion

In this paper, we exploit a convolution neural network to learn more robust and effective relation representations from shortest dependency paths for relation extraction. We further propose a simple negative sampling method to help make correct assignments for subjects and objects within a relationship. Experimental results show that our model significantly outperforms state-of-the-art systems and our treatment to dependency paths can well capture the syntactic features for relation extraction.

References

[Bunescu and Mooney2005a] Razvan C. Bunescu and Raymond J. Mooney. 2005a. A shortest path dependency kernel for relation extraction. In *HLT/EMNLP 2005, Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 6-8 October 2005, Vancouver, British Columbia, Canada*.

[Bunescu and Mooney2005b] Razvan C. Bunescu and Raymond J. Mooney. 2005b. Subsequence kernels for relation extraction. In *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*, pages 171–178.

[Chen et al.2014] Liwei Chen, Yansong Feng, Songfang Huang, Yong Qin, and Dongyan Zhao. 2014. Encoding relation requirements for relation extraction via joint inference. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 818–827.

[Collobert et al.2011] 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.

[Duchi et al.2011] John C. Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

[Hendrickx et al.2010] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of SemEval-2*, Uppsala, Sweden.

[Qian et al.2008] Longhua Qian, Guodong Zhou, Fang Kong, Qiaoming Zhu, and Peide Qian. 2008. Exploiting constituent dependencies for tree kernel-based semantic relation extraction. In *COLING 2008, 22nd International Conference on Computational Linguistics, Proceedings of the Conference, 18-22 August 2008, Manchester, UK*, pages 697–704.

[Socher et al.2012] 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 2012, July 12-14, 2012, Jeju Island, Korea*, pages 1201–1211.

[Turian et al.2010] Joseph P. Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 384–394.

[Zeng et al.2014] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.