Domain Information Enhanced Dependency Parser

Nan Yu[1], Zonglin Liu[2], Ranran Zhen[2], Tao Liu[2], Meishan Zhang[3], Guohong Fu[4](✉)

[1] School of Computer Science and Technology, Soochow University, China
`nyu@stu.suda.edu.cn`
[2] School of Computer Science and Technology, Heilongjiang University, China
{`liuzonglin1993,zenrran,taoooo1009`}`@gmail.com`
[3] School of New Media and Communication, Tianjin University, China
`mason.zms@gmail.com`
[4] Institute of Artificial Intelligence, Soochow University, China
`ghfu@suda.edu.cn`

Abstract. Dependency parsing has been an important task in the natural language processing(NLP) community. Supervised methods have achieved great success these years. However, these models can suffer significant performance loss when test domain differs from the training domain. In this paper, we adopt the Bi-Affine parser as our baseline. To explore domain-specific information and domain-independent information for cross-domain dependency parsing, we apply an ensemble-style self-training and and adversarial learning, respectively. We finally combine the two strategies to enhance our baseline model and our final system was ranked the first of at NLPCC2019 shared task on cross-domain dependency parsing.

Keywords: Cross-domain · Dependency Parsing · Self-training · Adversarial learning · Ensemble.

1 Introduction

Dependency parsing is a fundamental task in NLP research, which aims to parse the syntax structure of sentences by establishing the relationships between words. It has received great attention[3,18,10,8,9,26] these years, and most of them focus on the supervised methods for dependency parsing[17,3,10,8].

There are several algorithms for dependency parsing, like graph-based[21,27,8] and transition-based[3,31,10,1]. Given a sentence, the graph-based parsers compute the scores of all arcs and labels, and find the highest scoring dependency tree, while the transition-based parsers establish a dependency tree by a sequence of shift-reduce operations. Recently, the neural features have achieved great success in NLP tasks[5,6]. Many supervised dependency parsers adopt the neural features and achieved high performances in news domain[3,32,10,8](above 90%). Among these neural dependency parsers, the Bi-Affine parser[8] can achieve the top performance.

However, supervised methods have a deficiency on cross-domain dependency parsing, because of the data distribution difference between training data and test data. When these supervised models trained by the news domain training data, and then use them to predict new domain test data(like web fiction, production comments, etc.), the performances of them dropped drastically[23,18,11].

Self-training is an effective semi-supervised method, which can improve the original model performance by utilizing the data distribution information from unlabeled data. But when we apply self-training to dependency parsing, it relies on high-quality additional training data heavily[4,16,13]. Several inappropriate methods for obtaining additional training data even have a negative influence on cross-domain dependency parsing[24,2].

Ensemble has been a simple but effective method to obtain the high-quality automatic trees[28,20,23]. [7] combines the self-training and ensemble, proposing an ensemble-style self-training for cross-domain citation classification. Inspired by [7], we use ensemble models to predict the unlabeled data and obtain 1-best automatic trees. Then we sample the automatic trees randomly to obtain the high-quantity additional training data. Finally, the original parsers can learn the target domain-specific information by several re-training iterations with the additional training data.

Domain-independent information is another effective resource for cross-domain tasks. Domain adversarial training can extract domain-independent information by the adversarial domain classifier, which have been demonstrated helpful in dependency parsing[25]. Inspired by [25], we apply the adversarial learning to enhanced our baseline.

In this paper, we investigate two kinds of domain information for cross-domain dependency parsing. We follow [8], reimplementing the Bi-Affine parser as our baseline. Then, we enhance the Bi-Affine parser with target domain-specific information and the domain-independent information, which is complementary. On the one hand, we apply the ensemble-style self-training method to extract the domain specific information from unlabeled target domain data. On the other hand, we follow [25], extending our baseline with the adversarial learning to extract the domain-independent information. Our codes will be released for public at http://github.com/yunan4nlp/cross_domain_parser.

We conduct the experiments on the NLPCC2019 shared task corpus, and our final models rank the first in all teams[22]. We perform several development experiments to analyze the adversarial training and ensemble-style self-training, and the results show these two methods are both effective methods for cross-domain dependency parsing.

2 Related Work

2.1 Dependency parsing

There are two mainly algorithms for dependency parsing, the graph-based[21,27,8] and the transition-based[3,32,31,10,14,1].

The graph-based parsers compute the scores of all arcs and labels, and search the highest scoring dependency trees as output. And the transition-based parsers convert the dependency tree predictions into a sequence of action predictions. Both two kinds of parsers focus on the statistical models by using manually-designed features in early time[19,29,31,19].

Recently, the neural features have been widely investigated in dependency parsing[3,8,21,10,17]. Among these neural dependency parsers, the Bi-Affine parser[8] has achieved state-of-the-art performance. Thus, we reimplement the Bi-Affine parser as baseline, which is popular in dependency parsing[9,15,30].

2.2 Domain Adaption

Self-training methods have been demonstrated useful for dependency parsing by a number of studies[24,2,4,16,13]. The majority methods focus on complex additional training data sample methods to achieve this goal[4,16,13]. Ensemble is a simple but effective method for dependency parsing to improve the performance[28,20,23]. [7] propose an ensemble-style self-training method for citation classification, which can offer the high-quality additional training data for self-training. Inspired by [7], we extend the Bi-Affine parser with ensemble-style self-training.

Adversarial training has been proposed for cross-domain adaption, bringing significantly better performances for dependency parsing[25]. Our adversarial domain classifier is mainly borrowed by [12].

3 Our Methods

3.1 The Bi-Affine Parser

Our baseline model is borrowed from the Bi-Affine parser[8], which is the state-of-the-art dependency parsing model and has achieved the top performances.

The mainly structure of the Bi-Affine parser is consisted of five parts: (1)a embedding layers; Given a sentence, we input the word forms $\{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n\}$ and part-of-speech(POS) tags $\{\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_n\}$ of each word to get their neural embeddings respectively, and then concatenate them. (2)three bi-directional long short term memory(Bi-LSTM$^{\times 3}$) layers; (3)a multi-layer perceptron(MLP) layer; (4)a Bi-Affine layer, we input the hidden layers of MLP and output the scores of arcs and labels. (5)an arg max decoder. We introduce the Bi-Affine parser briefly, and for more details, one can read their original paper.

3.2 Adversarial Training

We mix two different domain training data and feed into the Bi-Affine parser[8]. The baseline parser can extract the rich information from both two domain training data, including syntax, domain information etc. Intuitively, the target domain-independent information is useful for the cross-domain adaption.

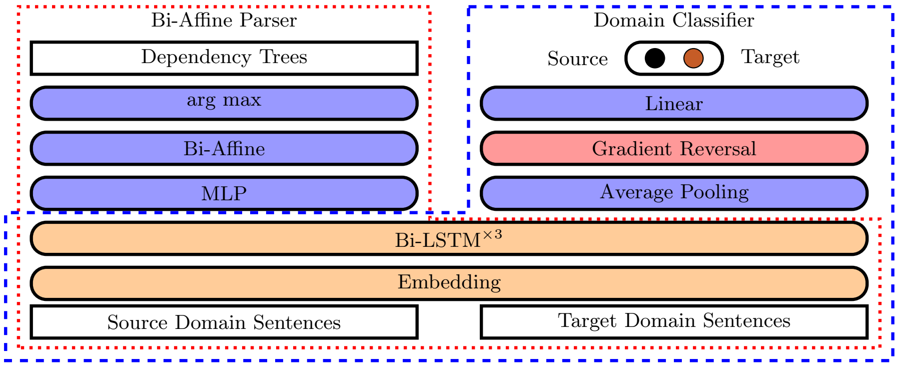


Fig. 1: The structure of adversarial learning based Bi-Affine parser

We follow [12], building an adversarial domain classifier with gradient reversal layer(GRL) to reduce the source domain-specific information and learn target domain-independent information.

Fig. 1 shows the structure of the Bi-Affine parser extended by adversarial learning, including a Bi-Affine parser and an adversarial domain classifier. In the adversarial domain classifier, the model parameters of the embedding layer and the Bi-LSTM layer are shared with the Bi-Affine parser. We use the Bi-LSTM hidden representations $\{\mathbf{h}_r^1, \mathbf{h}_r^2, ..., \mathbf{h}_r^n\}$ to calculate the sentence representation $\mathbf{h}_s$ by average pooling:

$$\mathbf{h}_s = \frac{1}{n}\sum_{1}^{n}\mathbf{h}_r^i \tag{1}$$

The essential part of the adversarial domain classifier is the GRL, and we apply it directly as follow:

$$\mathbf{h}_s = \mathrm{R}_\lambda(\mathbf{h}_s), \varDelta\mathbf{h}_s = -\lambda\frac{d\mathrm{R}_\lambda(\mathbf{h}_s)}{d\mathbf{h}_s} = -\lambda\mathbf{I} \tag{2}$$

where the $\varDelta\mathbf{h}_s$ is the gradient of the h_s, the λ is a constant factor, and the $\mathbf{I}$ is an identity matrix.

Finally, we use linear layer to compute the domain scores:

$$\mathbf{o} = \mathbf{W}\cdot\mathbf{h}_s \tag{3}$$

3.3 Ensemble-style Self-training

In this work, we follow [7], extending the baseline models with the ensemble-style self-training. Then we apply a random sample method to obtain additional training data, which add to original training data to re-train the basic models. Ensemble-style self-training is very similar to tri-training[33], which also trains three basic models and utilizes the unlabeled data to improve basic models. In previous work, the tri-training shows no improvement in dependency parsing[28].

First, we train three Bi-Affine parser models $\{\theta_1, \theta_2, \theta_3\}$ with same original training data $G = \{g_1, g_2, ..., g_x\}$ and different random seeds as our basic models. The performances of those models are almost same. We apply the ensemble method to those models, and predict the unlabeled data $U = \{u_1, u_2, ..., u_x\}$ to obtain the automatic dependency trees $U' = \{u'_1, u'_2, ..., u'_x\}$. We get the scores of arcs and labels, and apply softmax method to get the probalilities of arcs. We average the probalilities of arcs and labels, which is given by equation 4, and we select 1-best dependency trees as candidates.

$$\begin{cases} p^a = \frac{exp(\mathbf{s}^a)}{\sum_{a' \in A} exp(\mathbf{s}^{a'})}, p^l = \frac{exp(\mathbf{s}^l)}{\sum_{l' \in L} exp(\mathbf{s}^{l'})} \\ p^a_{avg} = \sum_1^3 p^a_i, p^l_{avg} = \sum_1^3 p^l_i \end{cases} \tag{4}$$

We sample 100k automatic dependency trees SU' randomly according to development experiment results and add those data into original training data. Then we use the new training data $G' = \{g_1, g_2, ..., g_x, u'_1, u'_2, ..., u'_r\}$ to re-train the three basic models. Our self-training iteration would be repeated for several time, and the additional training data SU' used only once. Finally, we use the ensemble models to predict the dependency trees of test data T to get the dependency trees T'. Algorithm 1 shows the pseudo codes of ensemble-style self-training algorithim.

Algorithm 1 Ensemble-style self-training algorithim

```
1: for $i = 1 \to 3$ do
2:     $\theta_i^1 \leftarrow Train(\theta_i^0, G)$
3: for $j = 1 \to k$ do
4:     $U' \leftarrow Ensemble(\{\theta_1^j, \theta_2^j, \theta_3^j\}, U)$
5:     $SU' \leftarrow Sample(U')$
6:     $G' \leftarrow SU' + G$
7:     for $i = 1 \to 3$ do
8:         $\theta_i^{j+1} \leftarrow Train(\theta_i^j, G')$
9: $T' = Ensemble(\{\theta_1^k, \theta_2^k, \theta_3^k\}, T)$
```

3.4 Training

We reimplement the Bi-Affine parser[8] and apply the cross-entropy plus with an l_2 regularization as the loss function of our models:

$$L(\theta) = \begin{cases} p_g^* = \frac{exp(\mathbf{s}_g^*)}{\sum exp(\mathbf{s}^{*'})} \\ -log(p_g^*) + \frac{\beta}{2}||\theta||^2 \end{cases} \tag{5}$$

where p_g^* means the probabilities of gold arcs, labels and domain, respectively; θ represents the model parameters of parser.

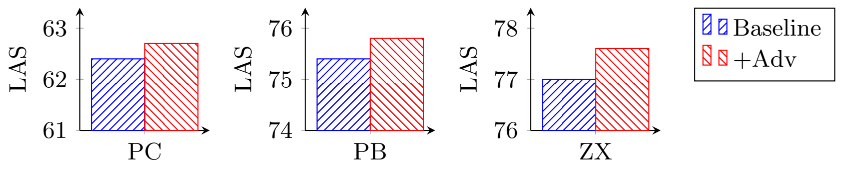


Fig. 2: The influence of adversarial learning.

4 Experiments

4.1 Settings

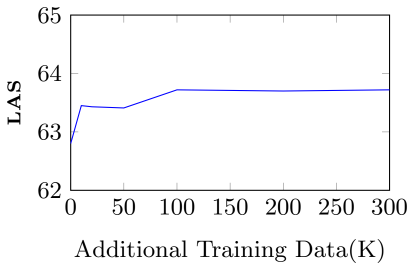
Data We conduct our experiments on NLPCC2019 shared task corpus[22], which has four domains: Balanced Corpus(BC, text selected from HLT-CDT and PennCTB treebanks), Products Comments(PC, from Taobao), Product Blogs(PB, from Taobao headlines), and the web fiction "Zhuxian(ZX)". The BC is the source domain, and others are target domains.

Evaluation We apply two standard evaluation methods to test our models, including unlabeled attachment scores(UAS) and labeled attachment scores(LAS). The UAS is the precision of the arcs without labels, and the LAS is the precision of the arcs with relation labels.

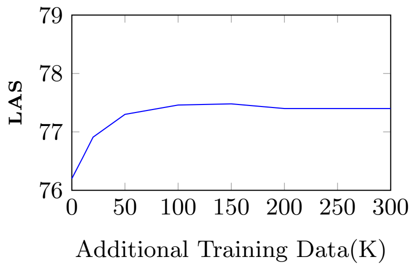
4.2 Results

There are two subtasks in the NLPCC2019 shared task: un-supervised and semi-supervised domain adaptation. The difference of two subtasks is whether can use the target domain training data. For example, when the target domain is PC, un-supervised task can only use the BC training data and PC unlabeled data to train the models. The semi-supervised task can add PC training data to original training data. Both two subtasks can only use PC development to fine-tune the models. In this section, we show three several factors of our models by conducting several sets of experiments in semi-supervised domain adaptation settings only.

Adversarial Training First, we apply the adversarial training to enhance our baseline model according to equation 2. We exploit constant factor $\lambda = 10^{-5}$ according to devlopment experiments. Fig. 2 shows the influence of the adversarial learning. When adversarial learning is exploited, our model achieves at 77.7 on LAS in ZX domains. And resulting overall improvements 77.7- 77.0 = 0.7 LAS. The PC and PB domains have similar tendencies as well.



(a) PC



(b) PB

Fig. 3: The influence of different quantities of additional training data.

Ensemble Then, we train three basic parsers for every domain by same original training data and a different random seed. And we apply the ensemble to average the arc and label probabilities from three basic models according to equation 4. The results are shown in Table 1, we can see the ensemble is the effective method for dependency parsing.

Table 1: The performances of ensemble. The Baseline$_*$ means the performances of baseline models, the Ensemble means the performances of ensemble method.

	PC(UAS/LAS)	PB(UAS/LAS)	ZX(UAS/LAS)
Baseline$_1$	70.0/62.0	80.5/75.4	81.4/77.0
Baseline$_2$	70.3/62.4	80.4/75.4	81.2/77.0
Baseline$_3$	70.4/62.2	80.1/75.2	81.8/77.0
Ensemble	**70.5/62.9**	**80.8/75.9**	**82.6/78.2**

Ensemble-style Self-training Then we examine the strategy of ensemble-style self-training. The amount of additional training data is a key factor for it. In Fig. 3, the performances are relatively stable surrounding 100k additional training data, where all domains under both settings are close to their peak values. Thus we random select 100k additional training data as the final setting. The ZX domain has only 30k additional training, so we use all of them in self-training.

Final Model We combine the adversarial learning and the ensemble-style self-training to enhance the baseline models. The experiment results are shown in Table 2. We find both two methods can improve the baseline models. In ZX domain, the baseline model achieves 81.4 on UAS and 77.0 on LAS. When we apply the adversarial learning, and baseline can improve by 82.2 - 81.4 = 0.8

UAS and 77.7 - 77.0 = 0.7 LAS, when we use the ensemble-style self-training to utilize the data distribution information from unlabeled target domain data. Baseline can improve by 82.8 - 81.4 = 1.4 UAS and 78.5 - 77.0 = 1.5 LAS. Finally, when we combine both two methods to baseline models, we find that the baseline model can be improved by 83.2 - 81.4 = 1.8 UAS and 79.0 - 77.0 = 2 LAS. Both PC and PB domains have similar tendencies as well.

Table 2: The influence of adversarial learning and ensemble-style self-training in semi-supervised adaptation. +A means apply adversarial learning to baseline. +S means apply ensemble-style self-training to baseline. +A+S means combine the adversarial learning and ensemble-style self-training to baseline.

	PC(UAS/LAS)	PB(UAS/LAS)	ZX(UAS/LAS)
Baseline	70.6/62.4	80.1/75.4	81.4/77.0
+Adv	70.6/62.7	80.6/75.8	82.2/77.7
+S	**71.5/63.7**	81.8/77.3	82.8/78.5
+A+S	71.2/63.4	**82.6/78.2**	**83.2/79.0**

5 Conclusion

We described our models submitted in the NLPCC2019 shared task on cross-domain dependency parsing. First, we borrowed the Bi-Affine parser[8] as our baseline, and exploited the ensemble-style self-training method with a simple random data sample method. Second, we enhanced them with adversarial training. Final results show that our models are very competitive and achieve the top performances in all teams[22].

6 Acknowledgments

We thank our anonymous reviewers for their helpful comments. This work was supported by National Natural Science Foundation of China (Grant No.61672211, U1836222).

References

1. Ballesteros, M., Dyer, C., Goldberg, Y., Smith, N.A.: Greedy transition-based dependency parsing with stack lstms. Computational Linguistics **43**(2), 311–347 (2017)
2. Cerisara, C.: Semi-supervised experiments at loria for the spmrl 2014 shared task. In: Proc. of the Shared Task on Statistical Parsing of Morphologically Rich Languages (2014)

3. Chen, D., Manning, C.: A fast and accurate dependency parser using neural networks. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 740–750 (2014)
4. Chen, W., Wu, Y., Isahara, H.: Learning reliable information for dependency parsing adaptation. In: Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1. pp. 113–120. Association for Computational Linguistics (2008)
5. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of machine learning research **12**(Aug), 2493–2537 (2011)
6. Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R., Makhoul, J.: Fast and robust neural network joint models for statistical machine translation. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1370–1380 (2014)
7. Dong, C., Schäfer, U.: Ensemble-style self-training on citation classification. In: Proceedings of 5th international joint conference on natural language processing. pp. 623–631 (2011)
8. Dozat, T., Manning, C.D.: Deep biaffine attention for neural dependency parsing. arXiv preprint arXiv:1611.01734 (2016)
9. Dozat, T., Qi, P., Manning, C.D.: Stanford's graph-based neural dependency parser at the conll 2017 shared task. In: Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. pp. 20–30 (2017)
10. Dyer, C., Ballesteros, M., Ling, W., Matthews, A., Smith, N.A.: Transition-based dependency parsing with stack long short-term memory. arXiv preprint arXiv:1505.08075 (2015)
11. Foster, J., Cetinoglu, O., Wagner, J., Le Roux, J., Hogan, S., Nivre, J., Hogan, D., Van Genabith, J.: # hardtoparse: Pos tagging and parsing the twitterverse. In: Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence (2011)
12. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. arXiv preprint arXiv:1409.7495 (2014)
13. Goutam, R., Ambati, B.R.: Exploring self training for hindi dependency parsing. In: Proceedings of 5th International Joint Conference on Natural Language Processing. pp. 1452–1456 (2011)
14. Honnibal, M., Johnson, M.: An improved non-monotonic transition system for dependency parsing. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1373–1378 (2015)
15. Jiang, X., Li, Z., Zhang, B., Zhang, M., Li, S., Si, L.: Supervised treebank conversion: Data and approaches. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 2706–2716 (2018)
16. Kawahara, D., Uchimoto, K.: Learning reliability of parses for domain adaptation of dependency parsing. In: Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II (2008)
17. Kiperwasser, E., Goldberg, Y.: Simple and accurate dependency parsing using bidirectional lstm feature representations. Transactions of the Association for Computational Linguistics **4**, 313–327 (2016)
18. Kong, L., Schneider, N., Swayamdipta, S., Bhatia, A., Dyer, C., Smith, N.A.: A dependency parser for tweets. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1001–1012 (2014)

19. Koo, T., Collins, M.: Efficient third-order dependency parsers. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. pp. 1–11. Association for Computational Linguistics (2010)
20. Le Roux, J., Foster, J., Wagner, J., Kaljahi, R., Bryl, A.: Dcu-paris13 systems for the sancl 2012 shared task (2012)
21. Pei, W., Ge, T., Chang, B.: An effective neural network model for graph-based dependency parsing. 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). pp. 313–322 (2015)
22. Peng, X., Li, Z., Zhang, M., Wang, R., Zhang, Y., Si, L.: Overview of the nlpcc 2019 shared task:cross-domain dependency parsing. In: Proceedings of The 8th CCF International Conference on Natural Language Processing and Chinese Computing (NLPCC-2019) (2019)
23. Petrov, S., McDonald, R.: Overview of the 2012 shared task on parsing the web (2012)
24. Plank, B., Søgaard, A.: Experiments in newswire-to-law adaptation of graph-based dependency parsers. In: International Workshop on Evaluation of Natural Language and Speech Tool for Italian. pp. 70–76. Springer (2012)
25. Sato, M., Manabe, H., Noji, H., Matsumoto, Y.: Adversarial training for cross-domain universal dependency parsing. In: Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. pp. 71–79 (2017)
26. Shareghi, E., Li, Y., Zhu, Y., Reichart, R., Korhonen, A.: Bayesian learning for neural dependency parsing. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 3509–3519 (2019)
27. Wang, W., Chang, B.: Graph-based dependency parsing with bidirectional lstm. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 2306–2315 (2016)
28. Zhang, M., Che, W., Liu, Y., Li, Z., Liu, T.: Hit dependency parsing: Bootstrap aggregating heterogeneous parsers. In: Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL) (2012)
29. Zhang, Y., Clark, S.: A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. pp. 562–571. Association for Computational Linguistics (2008)
30. Zhang, Y., Li, Z., Lang, J., Xia, Q., Zhang, M.: Dependency parsing with partial annotations: An empirical comparison. In: Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 49–58 (2017)
31. Zhang, Y., Nivre, J.: Transition-based dependency parsing with rich non-local features. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2. pp. 188–193. Association for Computational Linguistics (2011)
32. Zhou, H., Zhang, Y., Huang, S., Chen, J.: A neural probabilistic structured-prediction model for transition-based dependency parsing. 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). pp. 1213–1222 (2015)
33. Zhou, Z.H., Li, M.: Tri-training: Exploiting unlabeled data using three classifiers. IEEE Transactions on Knowledge & Data Engineering (11), 1529–1541 (2005)