Overview of the NLPCC 2019 Shared Task: Cross-Domain Dependency Parsing

Xue Peng[1], Zhenghua Li[1(✉)], Min Zhang[1], Rui Wang[2], Yue Zhang[2], and Luo Si[2]

[1] School of Computer Science and Technology, Soochow University, Suzhou, China
20175227031@stu.suda.edu.cn, {zhli13,minzhang}@suda.edu.cn
[2] Alibaba Group, Hangzhou, China
{masi.wr,shiyu.zy,luo.si}@alibaba-inc.com

Abstract. This paper presents an overview of the NLPCC 2019 shared task on cross-domain dependency parsing, including (1) the data annotation process, (2) task settings, (3) methods, results, and analysis of submitted systems and our recent work (Li+19), (4) discussions on related works and future directions. Considering that unsupervised domain adaptation is very difficult and has made limited progress in the past decades, we for the first time setup semi-supervised subtasks that allow to use a few thousand target-domain labeled sentences for training. We provide about 17 K labeled sentences from a balanced corpus as the source domain (BC), and as three target domains 10 K sentences from product comments (PC), 8 K sentences from product blogs (PB), and 3 K sentences from the web fiction named "Zhuxian" (ZX). All information about this task can be found at http://hlt.suda.edu.cn/index.php/Nlpcc-2019-shared-task, including the data sharing agreement.

1 Introduction

With the surge of web data (or user generated content), cross-domain parsing has become the major challenge for applying syntactic analysis in realistic NLP systems. To meet the challenge of the lack of labeled data, we have manually annotated large-scale high-quality domain-aware datasets with a lot of effort in the past few years.[1] Figure 1 shows an example dependency tree.

For this shared task, we provide about 17 K sentences from a balanced corpus as the source domain (BC), and as three target domains 10 K sentences from product comments (PC), 8 K sentences from product blogs (PB), and 3 K sentences from the web fiction named "Zhuxian" (ZX). We setup four subtasks with two cross-domain scenarios, i.e., unsupervised domain adaptation (no target-domain training data) and semi-supervised (with target-domain training data), and two settings, i.e., closed and open.

[1] Webpage for our treebank annotation: http://hlt.suda.edu.cn/index.php/SUCDT.

Supported by National Natural Science Foundation of China (Grant No. 61876116, 61525205).

J. Tang et al. (Eds.): NLPCC 2019, LNAI 11839, pp. 760–771, 2019.
https://doi.org/10.1007/978-3-030-32236-6_69

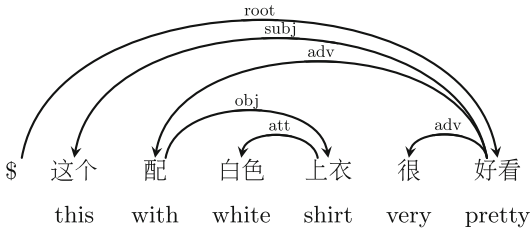

Fig. 1. An example from the PC domain. The English translation is "This looks very pretty with a white shirt."

Please kindly note that *semi-supervised in our domain adaptation scenario means that the model is also provided with labeled training data for the target domain*. In contrast, *traditional semi-supervised learning refers to approaches that make use of large-scale unlabeled data beside labeled data*. Using target-domain unlabeled data is usually considered a must for domain adaptation, and thus we use *semi-supervised* to distinguish whether using target-domain labeled data, not unlabeled data.

There are totally 16 teams that sign in this shared task, and 2 other teams that only want to use the data for study. Finally, 7 teams submit their results, among which one team misunderstands the task settings and recalls their submission.

One team (SJ_SuperLZ) submits results for all four subtasks, and wins the first place in the semi-open subtask 4 among two submissions. Another 3 teams submit results for both closed (un/semi) subtasks, and SyntaxError wins the first place on both tracks. One team focuses on the semi-closed subtask 2 and another team focuses on the semi-open subtask 4. Only two teams participate in the open (un or semi) tracks, and the use of contextualized word representations such as ELMo [15] and BERT [4] is not common. Only SJ_SuperLZ builds their system upon BERT via fine-tuning.

In order to gain more insights on the state-of-the-art progress, we also experiment with the approaches with the same codes in our recent work (Li et al. 2019, abbr. as Li+19) [10] and compare with those submitted systems.

2 Related Shared Tasks

Due to space limitation, we only give a brief and incomplete introduction on previous shared tasks on dependency parsing, especially domain adaptations. Please refer to Li+19 for discussions on methods previously proposed for cross-domain dependency parsing.

As the first CoNLL shared task on dependency parsing, **CoNLL-2006** on multi-Lingual dependency parsing releases benchmark datasets for 13 languages [1].

CoNLL-2007 focuses on both multilingual dependency parsing on 10 languages and domain adaptation on English [13]. For English, they release two

non-WSJ out-of-domain datasets, i.e., PCHEM (biomedical or chemical research abstracts) and CHILDES (the EVE corpus, only unlabeled dependency tress) datasets. They provide a combined development dataset for parameter selection and two separate test datasets for final evaluation. Although the organizers put a lot of effort to make the out-of-domain datasets be the consistent with the WSJ data in annotation schemes, the annotated trees still contains a lot of consistencies between source- and target-domains, which may be the reason behind the failure of many adaptation methods, as discussed in Dredze et al. [5].

CoNLL-2008 proposes the tasks of joint parsing of syntactic and semantic dependencies for English, with the aim of encouraging joint modeling of two related tasks, i.e., dependency parsing and word-based SRL [19]. They convert span-based SRL into word-based semantic dependencies based on head-finding rules.

CoNLL-2009 extends the previous-year task from monolingual English to multilingual, covering 7 languages [7]. For English, German, and Czech, they also prepare out-of-domain test datasets for final evaluation. However, the focus is not on domain adaptation without providing the dev data, though initial results are obtained on the outside domains.

CoNLL-2017/2018 proposes multilingual parsing from raw text to universal dependencies for many languages [23,24], thanks to the development of the University Dependencies project [14]. Participants need to process raw text without gold-standard tokenization (or word segmentation) and morphological features such as part-of-speech tags. CoNLL-2018 covers 82 UD treebanks in 57 languages.

SANCL-2012. Petrov and McDonald [16] organized the "parsing the web" shared task and the first workshop on syntactic analysis of non-canonical language (SANCL), based on their newly-constructed Google English Web Treebank consisting of web texts of five sources, i.e., email, answers, weblogs, newsgroups, and reviews. Each source has 1–2 K test sentences manually annotated with constituent trees. The goal is to adapt WSJ-trained parsers (constituent- or dependency-based) to these new domains.

3 Data Annotation

Due to space limitation, we only introduce our data annotation process very briefly. Please refer to Li et al. [10] for more details.

Annotation Guideline. During the past few years, we have been developing and updating a detailed annotation guideline that on the one hand aims to fully capture Chinese syntax, and on the other hand tries to guarantee inter-annotator consistency and facilitate model learning. The current guideline has about 80 pages and contains 21 dependency labels. Please refer to the SUCDT webpage for the newest version.

Annotation Process. All sentences are labeled by two persons (*annotation phase*), and inconsistent submissions are discussed by senior annotators to

determine the final answer (*review phase*). Annotators are required to re-do the incorrect submissions according to the answer given in the annotation platform, so that they can gradually improve their annotation ability (*study phase*). However, they can also decide to make complaints in the annotation system if they are sure that the provided answer is wrong, so that more senior annotators can re-check the answer (*complaint phase*).

Partial Annotation. In order to reduce annotation cost, we adopt the active learning procedure based on partial annotation, where sentences that are most difficult for the model are first chosen, and then a certain percentage of most difficult words are then selected for manual annotation. Most sentences in our labeled data are partially annotated. However, since training with partial trees is not popular in parsing community, we automatically complete all partial trees into high-quality full trees [26] based on our CRF-enhanced biaffine parser [9] trained on the combination of all our labeled data.[2]

Table 1 shows data statistics for this shared task. The source domain is balanced corpus (BC), composed of 8K sentences from HIT-CDT [2] and 8K from Penn CTB [21], annotated according to our guideline.[3] The first domain is product comments (PC) from Taobao. The product blog (PB) texts are crawled from the Taobao headline website, which contains articles written by users mainly on description and comparison of different commercial products. The third target domain is "Zhuxian" (ZX, also known as "Jade dynasty"), a web fiction previously annotated and used for cross-domain word segmentation [25].

The datasets are annotated by many different annotators, and the averaged sentence lengths and annotated words numbers per sentence also vary for different domains. Therefore, the consistency ratios can only give a rough idea on the annotation difficulty for texts of different domains.

All labeled datasets are shared by Soochow University and the PC/PB unlabeled datasets are provided by Alibaba Group. Participants need to sign corresponding data sharing agreements to obtain the data.

Table 1. Data statistics. K means thousand in sentence number.

	BC	PC	PB	ZX
train/dev/test (K)	16.3/1/2	6.6/1.3/2.6	5.1/1.3/2.6	1.6/0.5/1.1
consensus ratio (sent-wise)	45.88	39.20	35.88	46.20
consensus ratio (token-wise)	82.25	68.23	69.38	79.21
aver. sent len	13.82	13.95	12.12	21.25
aver. num of annotated words	3.80	3.33	5.09	5.07
unlabeled (K)	0	350	300	30

[2] The parser can be tried at http://hlt-la.suda.edu.cn.

[3] Our major purpose for annotating these datasets is to support supervised treebank conversion.

4 Task Settings

- Subtask 1 (un-closed): unsupervised domain adaptation (closed)
- Subtask 2 (semi-closed): semi-supervised domain adaptation (closed)
- Subtask 3 (un-open): unsupervised domain adaptation (open)
- Subtask 4 (semi-open): semi-supervised domain adaptation (open)

Unsupervised domain adaptation assumes that there is no labeled training data for the target domain. For example, when the target domain is PC, in the unsupervised domain adaptation scenario, you cannot use PC-Train. However, PC-Dev/Unlabeled are allowed to use.

Semi-supervised domain adaptation means that there exists a labeled training dataset for the target domain. For example, when the target domain is PC, PC-Train/Dev/Unlabeled all can be used in semi-supervised domain adaptation scenario.

Closed means that (1) you can only use our provided data and information, including the our provided word segmentation, automatic part-of-speech (POS) tags, pre-trained word embeddings;[4] (2) it is not allowed to use other resources such as dictionaries, labeled or unlabeled data for related tasks; (3) it is not allowed to use other tools to produce new information such as ELMo/BERT, POS tagger, etc.

Open has no restriction, and you can use any resource. However, it is strongly recommended that participants of this shared task and future researchers clearly describe in their system reports and papers what external resources are used and how parsing performance is affected.

Multi-source domain adaptation? NO. It is not allowed to use training data from other target domains. For example, when the target domain is PC, PB-Train and ZX-Train cannot be used in all four subtasks. However, after this shared task, researchers may explore this research line with our data.

Train with dev data? NO. It is not allowed to add dev data into training data. Dev data can only be used for parameter and model selection.

Evaluation Metrics. We use the standard unlabeled/labeled attachment score (UAS/LAS, percent of words that receive correct heads [and labels]). We do not complete partial trees into full ones for the dev and test datasets. It is straightforward to perform evaluation against partial trees by simply omitting the words without gold-standard heads. For any subtask, a participant team must submit results on all three target domains, so that we can obtain three LAS values. We average the three LAS directly to determine the final ranking.

5 Methods

5.1 Submitted Systems

We delete the detailed descriptions of the methods of the submitted systems, which are in the order of task registration **SJ_superLZ** (Shanghai Jiao Tong

[4] The word embeddings are obtained by training word2vec on the Chinese Gigaword 3 and all the target-domain unlabeled data.

University, Li et al. [11]), **14yhl9days** (University of South China), **SyntaxError** (Heilongjiang University, Yu et al. [22]), **AntNLP** (East China Normal University), **BLCU_Parser** (Beijing Language and Culture University), **NNU** (Nanjing Normal University, Xia et al. [20]). *Please refer to the long version of this paper at Zhenghua' homepage.*

5.2 Summaries of the Systems from Different Aspects

Basic Parsers. Three teams (i.e., SJ_superLZ, SyntaxError, BLCU_Parser) adopt the biaffine parser, or a similar graph-based framework, due to its simplicity and state-of-the-art performance. AntNLP employs the transition-based L2RPTR parser [6] and their recently proposed graph-based GNN parser [8], as two different views for the co-training procedure. NNU adopts the transition-based STACKPTR parser [12], and 14yhl9days uses the basic transition-based parser [3] extended with a BiLSTM encoder.

Our preliminary experiments with the biaffine parser show that using charLSTM-based word representation besides word embeddings, also mentioned by BLCU_Parser, leads to better performance than using POS tag embeddings.

Handling Partially Annotated Data. To control the influence of noisy arcs in the completed trees, SyntaxError finds that it is helpful to simply removing arcs with marginal probability less than 0.7 for both the auto-completed full trees in the labeled training data and the auto-parsed 1-best trees in the unlabeled data during loss computation.

After reading their system report, we also run experiments based on the biaffine parser (w/ local loss) to verify this issue. We find that compared with using all dependencies, LAS of the CONCAT method (semi-supervised) decreases by about 4 points if using only manually labeled dependencies (probability being 1 in the provided data) on ZX-dev, and LAS increases by about 0.4 points if only using dependencies of probability higher than 0.7.

Our results show that (1) the completed full trees are actually of high-quality, considering we use a CRF-enhanced biaffine parser trained on about 100 thousand labeled sentences for the completion task; (2) filtering low-probability dependencies is helpful, but not so very helpful on the labeled training data (maybe more useful for unlabeled data).

Our previous work demonstrates that using probabilistic CRF-based parser can more effectively learn from partial trees (manually annotated or after filtering) [26].

Combination of Source- and Target-Domain Labeled Data. For the semi subtasks 2 and 4, it is a very important to combine and balance the source- and target-domain labeled training data. Four different strategies are investigated.

(1) The fine-tuning method first trains the model on the source-domain training and then fine-tunes the model with the target-domain training data.

This idea is used by SJ_superLZ, BLCU_Parser and NNU. This method may suffer from the limited use of the source-domain training data.

(2) The MTL method, used by BLCU_Parser, treats the source- and target-domain parsing as two independent tasks under the MTL framework. Since only the encoder parameters are shared, the contribution of the source-domain training data is limited, considering the two domains have the same annotation guideline.

(3) The concatenation (CONCAT) method directly merges the two training datasets together without corpus weighting. This method is intuitively more effective than the above two, and is adopted by SyntaxError and AntNLP.

(4) The domain embedding (DOEMB) method is similar to the CONCAT method but distinguishes whether the input sentence comes from the source or target domains by appending a domain id embedding for each word. This method is investigated in Li+19 and is shown to be more effective than the CONCAT method.

The corpus weighting strategy, as described in Li+19, may be useful for balancing the contributions of the source- and target-domain training data, especially when the scales of the two datasets have large gap.

Utilizing Unlabeled Data. It is very attractive to utilize the large-scale and easy-to-collect unlabeled target-domain data in the research area of domain adaptation. Self/co/tri-training are widely used as typical approaches in traditional semi-supervised learning.[5] The basic idea is first using the current parser(s) to automatically parse the unlabeled data, and selecting high-confidence parse trees as extra training data, and re-training the parser(s), and so on, in a bootstrapping fashion. Both SyntaxError and SJ_superLZ use tri-training, and AntNLP adopts co-training.

Recently, the emergence of contextualized word representations, such as ELMo and Bert, has greatly advanced NLP research. The use of very-large-scale unlabeled data via language model loss, allows very deep neural networks to effectively learn extensively useful word and sentence representations. The improvement from using such representations is surprisingly large for a variety of NLP tasks, sometimes surpassing the total amount of improvement from research during the past decade. In some sense, it seems not enough to classify ELMo/BERT-enhanced approaches as traditional semi-supervised learning approaches. In fact, some researchers even base on ELMo and BERT as a new transferring learning (broader than domain adaptation) methodology [17].

In the open subtasks 3 and 4, the use of ELMo/BERT is rare so far, with the exception of SJ_superLZ. With many different ways to use ELMo/BERT for domain adaptation, we believe that it is very helpful to fine-tune ELMo/BERT on the unlabeled data of both the source and target domains. This simple method can build connections between the open-domain very-large-scale unlabeled texts

[5] Please notice again that semi-supervised in our domain adaptation scenario is about whether using target-domain labeled training data.

for pre-training ELMo/BERT and the data for our tasks in hands, and between the source and target domains.

Model Ensemble. Model ensemble has been used as an extremely useful and popular technique in many shared tasks. Both SyntaxError and 14yhl9days use model ensemble. For traditional discrete-feature based parsing, it is usually needed to use several divergent models trained on the same training data or a homogeneous model trained on random subsets of the whole training data. For neural network based parsing, different models can usually be obtained by using different random seeds for parameter initialization, which is adopted by SyntaxError.

The arc/label probabilities of different models are averaged for finding the optimal tree via Viterbi decoding, as SyntaxError does. Another popular way is to use the vote counts of the 1-best output parses of different models as the arc/label score for Viterbi decoding [18].

6 Results and Analysis

6.1 Re-running Our Li+19 Codes

To understand the effectiveness of approaches investigated in Li+19, we run the same codes[6] on the datasets of this shared task.

The data settings differ in two major aspects between this shared task and Li+19. First, an extra PC domain is used in this shared task. Second, the BC training data in Li+19 contains about 52 K sentences from HIT-CDT, including about 8 K manually labeled sentences and 44 K sentences (noisy full trees) being converted from the HIT-CDT guideline through supervised treebank conversion [9]. Therefore, the BC training data in Li+19 is larger but more noisy.

We delete the results and discussions on the effect of **corpus weighting.** *Please refer to the long version of this paper at Zhenghua's homepage.*

Main results on the dev data. Table 2 shows the results on the dev data for the methods investigated in Li+19, i.e., CONCAT, DOEMB, ELMo, and fine-tuned ELMo. Most observations are consistent with Li+19, though the BC training data differs. In the following, we briefly examine and discuss the results from different perspectives.

Domain differences. Training on BC-train produces best performance on ZX, and worst performance on PC, indicating that BC is most similar with ZX and most dissimilar with PC. Compared with training with BC-train, training on the corresponding target-domain training data greatly boosts parsing performance, especially for PC.

CONCAT vs. DOEMB. The results clearly show that for the semi-supervised scenario, DOEMB is consistently superior to CONCAT for combining the source- and target-domain data, increasing LAS by 1.5-2.2.

[6] https://github.com/SUDA-LA/ACL2019-dp-cross-domain.

Table 2. Final results on the dev data. FT-ELMo means fine-tuned ELMo.

	PC		PB		ZX	
	UAS	LAS	UAS	LAS	UAS	LAS
Trained on the source-domain data (unsupervised)						
BC-train	40.26	25.64	67.40	60.60	71.22	63.73
BC-train + ELMo	44.31	30.46	70.91	64.70	73.95	66.17
BC-train + FT-ELMo	**52.33**	**39.13**	**77.76**	**71.77**	**76.27**	**69.70**
Trained on the target-domain data (semi-supervised)						
PC-train	**68.03**	**59.68**	62.90	55.98	54.67	44.69
PB-train	42.14	30.13	**78.11**	**72.68**	60.24	49.34
ZX-train	34.71	20.40	58.36	50.75	**75.99**	**70.70**
PC-train + FT-ELMo	**72.34**	**63.82**	73.46	67.24	63.33	53.27
PB-train + FT-ELMo	54.89	43.03	**82.18**	**77.10**	71.34	61.20
ZX-train + FT-ELMo	44.94	29.68	70.39	62.95	**80.92**	**75.83**
Trained on source- and target-domain data (semi-supervised)						
CONCAT	68.31	59.42	78.11	72.80	79.44	74.35
DOEMB	69.37	60.94	80.03	75.00	80.76	76.23
DOEMB + ELMo	*69.20*	*60.48*	80.31	75.43	81.24	*75.99*
DOEMB + FT-ELMo	**72.01**	**63.86**	**83.10**	**78.87**	**83.21**	**78.84**

Effect of un-fine-tuned ELMo. We use the same ELMo with Li+19, which is trained on the general-purpose Chinese Gigaword 3 corpus. We have added un-fine-tuned ELMo for unsupervised "BC-train + ELMo" and semi-supervised "DOEMB + ELMo", leading to opposite effect. In the unsupervised scenario, using ELMo representations as extra input boosts parsing accuracy by large margin (2.4–4.8 in LAS). In contrast, in the semi-supervised case, un-fine-tuned ELMo has little effects, maybe due to the high baseline (DOEMB) performance using the target-domain training data. We do not conduct experiments using un-fine-tuned ELMo on other scenarios to save computation resource.

Effect of fine-tuned ELMo. Similar to Li+19, we fine-tune ELMo on the combined corpus of BC-train and (PC/PB/ZX)-(train/unlabeled). It is very clear that fine-tuned ELMo can effectively further boost parsing accuracy over un-fine-tuned cases, i.e., "BC-train + ELMo" by 2.5–8.7, and "DOEMB + ELMo" by 2.9–3.4. When training with only the target-domain data in the second major row (comparing the diagonals), fine-tuned ELMo can improve LAS by 4.1–5.1.

Comparing the performance of "PC-train + FT-ELMo" and "DOEMB + FT-ELMo" on PC-dev, we can see that the source-domain BC-train provides little help, indicating again that BC and PC diverges too much to be helpful for the latter. We leave the investigation of the deeper reason behind this for future.

Overall, it is clear that the major conclusions drawn in Li+19 still stand on the datasets of this shared task. First, the domain embedding approach with

Table 3. Final results on the test data. SJ_superLZ reruns the experiments and updates their results at the posterior evaluation stage, due to incidental experimental mistakes for un-open subtask 3.

	PC		PB		ZX		Averaged	
	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
Un-closed Subtask 1								
SyntaxError	**49.82**	**36.86**	**71.48**	**65.43**	**73.90**	**66.54**	**65.07**	**56.28**
AntNLP	43.64	30.47	70.50	63.83	72.88	65.07	62.34	53.12
Ours (Biaffine)	38.80	25.87	66.83	60.28	69.27	61.37	58.30	49.17
SJ_superLZ	35.63	21.33	62.55	54.35	50.04	38.31	49.40	38.00
14yhl9days	26.72	10.92	41.50	27.38	40.45	26.44	36.22	21.58
Semi-closed Subtask 2								
SyntaxError	72.18	**64.12**	82.57	**77.83**	**80.53**	**75.84**	**78.43**	**72.60**
AntNLP	**72.23**	63.96	**82.94**	77.27	79.89	74.80	78.35	72.01
Ours (DOEMB)	68.88	59.96	79.22	74.40	79.60	74.50	75.90	69.62
BLCU_Parser	70.89	61.72	79.30	74.25	77.84	72.58	76.01	69.51
SJ_superLZ	69.88	59.43	77.52	71.33	77.84	71.55	75.08	67.44
14yhl9days	47.38	26.26	47.26	32.88	45.40	32.00	46.68	30.38
Un-open Subtask 3								
SJ_superLZ (BERT, Post-Eval)	**60.50**	**49.49**	**81.61**	**76.77**	**79.74**	**74.32**	**73.95**	**66.86**
Ours (+FT-ELMo)	53.04	39.48	77.15	71.54	74.68	67.51	68.29	59.51
Semi-open Subtask 4								
SJ_superLZ (BERT, Post-Eval)	**75.25**	**67.77**	**85.53**	**81.51**	**86.14**	**81.65**	**82.30**	**76.98**
Ours (DOEMB+FT-ELMo)	73.16	64.33	83.05	78.57	82.09	77.08	79.43	73.33
SJ_superLZ (BERT)	72.40	64.37	80.79	76.13	83.15	78.61	78.78	73.04
NNU	70.97	61.82	80.59	75.85	79.33	74.35	76.96	70.68

corpus weighting is the most effective way to combine the source- and target-domain training data in the semi-supervised scenario (maybe adversarial training can further improve). Second, fine-tuning ELMo on the target-domain unlabeled data is extremely helpful due to the effect of bridging the knowledge between general open-domain texts and target-domain texts.

6.2 Final Results on the Test Data

Table 3 shows the final results on the test datasets. SyntaxError dominates the un-closed subtask 1. Our Li+19 baseline biaffine parser ranks the third place. We guess we can further improve our result by (1) using charLSTM word representations; (2) performing model ensemble; (3) utilizing unlabeled data.

For the semi-closed subtask 2, though still being the best, SyntaxError outperforms AntNLP by a small margin. We do not know why the gap between them becomes much smaller than for the un-closed subtask 1. They used different techniques to handle the target-domain training data. SyntaxError uses CONCAT and adversarial training, whereas AntNLP directly combine source- and target-domain data (CONCAT).

For the un-open subtask 3, SJ_superLZ is the only submission and their posterior evaluation results outperform our Li+19 best method by 7.4 in LAS.

For the semi-open subtask 4, SJ_superLZ outperforms NNU. However, the DOEMB method with fine-tuned ELMo in Li+19 slightly outperforms SJ_superLZ, which is very interesting because BERT-based systems usually achieve much higher performance than ELMo-based ones. After retraining their models at the posterior evaluation stage, their averaged LAS increases by nearly 4 points.

7 Conclusions and Future Works

Overall, we feel that the attention so far is far from extensive for our first-year organization of the shared task on cross-domain dependency parsing. However, we have strong confidence on the future, and plan to organize more shared tasks by gradually releasing more labeled data covering more genres. We require all organizations that use our data in their experiments to release their full codes when publishing papers, in order to promote replicability.

References

1. Buchholz, S., Marsi, E.: CoNLL-X shared task on multilingual dependency parsing. In: In Proceedings of CoNLL, pp. 149–164 (2006)
2. Che, W., Li, Z., Liu, T.: Chinese dependency treebank 1.0 (Ldc2012t05). In: Philadelphia: Linguistic Data Consortium (2012)
3. Chen, D., Manning, C.: A fast and accurate dependency parser using neural networks. In: Proceedings of EMNLP, pp. 740–750 (2014)
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL, pp. 4171–4186 (2019)
5. Dredze, M., Blitzer, J., Pratim Talukdar, P., Ganchev, K., Graca, J.a., Pereira, F.: Frustratingly hard domain adaptation for dependency parsing. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007, pp. 1051–1055 (2007)
6. Fernández-González, D., Gómez-Rodríguez, C.: Left-to-right dependency parsing with pointer networks. In: Proceedings of NAACL, pp. 710–716 (2019)
7. Hajič, J., et al.: The CoNLL-2009 shared task: syntactic and semantic dependencies in multiple languages. In: Proceedings of CoNLL (2009)
8. Ji, T., Wu, Y., Lan, M.: Graph-based dependency parsing with graph neural networks. In: Proceedings of ACL (2019)
9. Jiang, X., Li, Z., Zhang, B., Zhang, M., Li, S., Si, L.: Supervised treebank conversion: data and approaches. In: Proceedings of ACL, pp. 2706–2716 (2018)
10. Li, Z., Xue, P., Zhang, M., Wang, R., Si, L.: Semi-supervised domain adaptation for dependency parsing. In: Proceedings of ACL (2019)
11. Li, Z., Zhou, J., Zhao, H., Wang, R.: Cross-domain transfer learning for dependency parsing (2019)
12. Ma, X., Hu, Z., Liu, J., Peng, N., Graham, N., Eduard, H.: Stack-pointer networks for dependency parsing. In: Proceedings of ACL, pp. 1403–1414 (2018)

13. Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., Yuret, D.: The coNLL 2007 shared task on dependency parsing. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007, pp. 915–932 (2007)
14. Nivre, J., et al.: Universal dependencies v1: a multilingual treebank collection. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016) (2016)
15. Peters, M.E., et al.: Deep contextualized word representations. In: Proceedings of NAACL-HLT, pp. 2227–2237 (2018)
16. Petrov, S., McDonald, R.: Overview of the 2012 shared task on parsing the web. In: Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL) (2012)
17. Ruder, S., Peters, M.E., Swayamdipta, S., Wolf, T.: Transfer learning in natural language processing. In: Proceedings of NAACL: Tutorials, pp. 15–18 (2019)
18. Sagae, K., Lavie, A.: Parser combination by reparsing. In: Proceedings of NAACL, pp. 129–132 (2006)
19. Surdeanu, M., Johansson, R., Meyers, A., Màrquez, L., Nivre, J.: The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In: CoNLL-2008 (2008)
20. Xia, Z., Wang, L., Qu, W., Zhou, J., Gu, Y.: Neural network based deep transfer learning for cross-domain dependency parsing. arXiv (2019)
21. Xue, N., Xia, F., Chiou, F.D., Palmer, M.: The Penn Chinese treebank: phrase structure annotation of a large corpus. Nat. Lang. Eng. **11**(02), 207–238 (2005)
22. Yu, N., Liu, Z., Zhen, R., Liu, T., Zhang, M., Fu, G.: Domain information enhanced dependency parser (2019)
23. Zeman, D., et al.: CoNLL 2018 shared task: multilingual parsing from raw text to universal dependencies. In: Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pp. 1–21 (2018)
24. Zeman, D., et al.: CoNLL 2017 shared task: multilingual parsing from raw text to universal dependencies. In: Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pp. 1–19, August 2017
25. Zhang, M., Zhang, Y., Che, W., Liu, T.: Type-supervised domain adaptation for joint segmentation and pos-tagging. In: Proceedings of EACL, pp. 588–597 (2014)
26. Zhang, Y., Li, Z., Lang, J., Xia, Q., Zhang, M.: Dependency parsing with partial annotations: an empirical comparison. In: Proceedings of IJCNLP, pp. 49–58 (2019)