Contrastive Learning on LLM Back Generation Treebank for Cross-domain Constituency Parsing

Peiming Guo[‡*], **Meishan Zhang**[‡], **Jianling Li**[§], **Min Zhang**[‡], **Yue Zhang**[¶†]
[‡] Institute of Computing and Intelligence, Harbin Institute of Technology (Shenzhen), China
[§] School of New Media and Communication, Tianjin University, China
[¶] School of Engineering, Westlake University, China
guopeiming.gpm@gmail.com, mason.zms@gmail.com, jianlingl@tju.edu.cn
zhangmin2021@hit.edu.cn, yue.zhang@wias.org.cn

Abstract

Cross-domain constituency parsing is still an unsolved challenge in computational linguistics since the available multi-domain constituency treebank is limited. We investigate automatic treebank generation by large language models (LLMs) in this paper. The performance of LLMs on constituency parsing is poor, therefore we propose a novel treebank generation method, LLM back generation, which is similar to the reverse process of constituency parsing. LLM back generation takes the incomplete cross-domain constituency tree with only domain keyword leaf nodes as input and fills the missing words to generate the cross-domain constituency treebank. Besides, we also introduce a span-level contrastive learning pre-training strategy to make full use of the LLM back generation treebank for cross-domain constituency parsing. We verify the effectiveness of our LLM back generation treebank coupled with contrastive learning pre-training on five target domains of MCTB. Experimental results show that our approach achieves state-of-the-art performance on average results compared with various baselines.

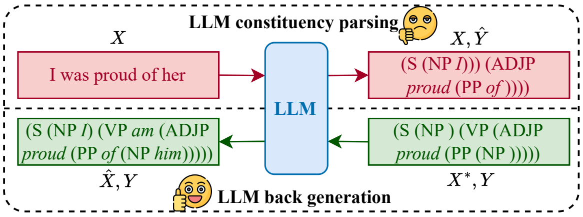

LLM constituency parsing
X
I was proud of her
LLM
$X, \hat{Y}$
(S (NP *I*))) (ADJP *proud* (PP *of*))))
(S (NP *I*) (VP *am* (ADJP *proud* (PP *of* (NP *him*)))))
$\hat{X}, Y$
(S (NP) (VP (ADJP *proud* (PP (NP)))))
X^*, Y
LLM back generation

Figure 1: LLM constituency parsing usually predicts the wrong constituency tree structure $\widehat{Y}$ for the input sentence X. However, LLM back generation can generate valid constituency tree structure with the appropriate sentence $\widehat{X}$ based on the masked sentence X^* and bare constituency tree structure Y.

1 Introduction

Constituency parsing is a fundamental task in computational linguistics that aims to build a hierarchical syntax tree for the given sentence. Although chart-based parsers (Stern et al., 2017; Kitaev and Klein, 2018; Teng and Zhang, 2018) achieve state-of-the-art results for the in-domain scenario (at least 95% F1 score for the news domain) based on the supervised learning and large-scale treebanks (Kitaev et al., 2019; Zhang et al., 2020; Tian et al., 2020; Cui et al., 2022), there is a performance gap in out-of-domain settings, since available multi-domain constituency treebank is limited (Yang et al., 2022; Li et al., 2023; Guo et al., 2024). Therefore, stable cross-domain constituency parsing performance is still a challenge for constituency parsing.

Using large language models (LLMs) (Brown et al., 2020; Ouyang et al., 2022; OpenAI, 2023) is a promising solution for dataset annotation in various natural language processing tasks, such as text classification (Törnberg, 2023), named entity recognition (Zhang et al., 2023a), semantic search (Bansal and Sharma, 2023), etc. For cross-domain constituency parsing, Li et al. (2023) first employ ChatGPT to generate unlabeled raw sentences based on grammar rules, and then the chart-based parser annotates a pseudo treebank on them. Integrating with self-training, this two-stage approach gains state-of-the-art cross-domain constituency parsing performance.

However, this approach indirectly uses LLM for treebank annotation, not taking full advantage of LLM abilities in domain generalization and language comprehension. Besides, the two-stage pipeline inevitably introduces noise and error propagation, resulting in parse trees with errors and limited cross-domain constituency transfer. Compared with an unlabeled raw corpus, guiding LLMs to directly generate the labeled constituency treebank in the target domain could be a more direct and effec-

[*] This work was done when Peiming Guo was a research assistant at Westlake University.

[†] Corresponding author: Yue Zhang.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 27446–27458
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

tive approach for cross-domain constituency parsing. Concretely, direct treebank generation resorts to powerful LLMs to generate the target domain sentence and the corresponding constituency tree simultaneously, which can directly train a cross-domain constituency parser.

To this end, we explore how to utilize LLMs to generate a cross-domain constituency treebank effectively in this paper. Since automatic treebank annotation by LLMs on the unlabeled sentence has poor performance (Bai et al., 2023), we propose a novel treebank generation method as shown in Figure 1, LLM back generation, which is similar to the reverse process of constituency parsing. Yang et al. (2022) indicate that both syntactical structure and domain vocabulary are crucial influence factors for cross-domain constituency parsing. Therefore, LLM back generation builds the cross-domain constituency treebank by filling in missing sentential words based on the target domain constituency tree structure and domain keywords in the tree. Concretely, we first extract the constituency tree and domain keywords on the target domain sentence. Then we reserve domain keywords and remove other sentential words from the target domain constituency tree, which implies the character of the target domain on the syntactical structure and domain vocabulary. Finally, we supply the LLM with the masked constituency tree and guide it to output the complete cross-domain parse tree.

In order to alleviate the noise in the LLM back generation treebank and reduce the cost and scale of LLM treebank generation, we design a span-level contrastive learning pre-training strategy for cross-domain constituency parsing, which can expand pre-training data significantly. For each constituent span, the span-level contrastive learning pre-training distinguishes the related valid constituent spans and invalid spans with adjacent boundaries. Specifically, we mine the left child, right child, parent and brother nodes as positive instances and the corresponding fifteen invalid spans as negative instances. To the best of our knowledge, we are the first to introduce contrastive learning into constituency parsing.

We conduct experiments to verify the effectiveness of our LLM back generation treebank and contrastive learning pre-training strategy for cross-domain constituency parsing. The news-domain constituency treebank PTB (Marcus et al., 1993) is selected as the source, and a multi-domain constituency treebank MCTB (Yang et al., 2022) as the target, which consists of five domains. Experimental results show that the LLM back generation treebank coupled with contrastive learning pre-training achieves state-of-the-art cross-domain parsing performance on the average F1 score, outperforming various baselines, including natural corpus treebank, conventional parsers, masked language modeling pre-training, previous cross-domain methods and large language models[1].

2 Related Work

Cross-domain Constituency Parsing. Constituency parsing is an important and fundamental task in computational linguistics, which has not been completely solved. The main challenge is stable cross-domain parsing performance. Early work of constituency parsing focuses on the news domain (Collins, 1997; Stern et al., 2017) and short sentences (McClosky et al., 2006, 2008). In recent years, the natural language processing community has begun to pay attention to constituency parsing on different domains. So there has been limited work investigating cross-domain constituency parsing. McClosky et al. (2010) propose multiple source parser adaptation, which trains constituency parsers on multiple domain treebanks and combines these models by linear regression. Joshi et al. (2018) study single source domain adaptation based on the contextualized word representations, where they train the parsers on PTB only for similar target domains. For syntactically distant target domains, they employ a dozen partial annotations to improve cross-domain constituency parsing performance. Fried et al. (2019) and Yang et al. (2022) perform a systematic analysis on various constituency parsers. Yang et al. (2022) annotate a constituency treebank MCTB, which contains five target domains. Guo et al. (2024) improve cross-domain constituency parsing performance by leveraging heterogeneous data from different types of tasks.

Researchers have investigated the effect of LLMs on cross-domain constituency parsing in recent years. Bai et al. (2023) conduct a comprehensive experiment on various LLMs, including ChatGPT, GPT-4, OPT, LLaMA and Alpaca. They also explore the influence of different linearizations and LLM settings, including zero-shot, few-shot and fine-tuning. Li et al. (2023) use grammar rules and target domain sentences as the input restriction

[1] Our code is public on https://github.com/guopeiming/Back_Parsing_LLM

and reference to guide ChatGPT to generate raw corpora. Our work is this line since we also focus on LLMs and cross-domain constituency parsing. However, we propose LLM back generation, which exploits the incomplete cross-domain constituency tree as the input restriction and generates the full parse tree for cross-domain constituency parsing.

Contrastive Learning. Contrastive learning (Chopra et al., 2005; Schroff et al., 2015; Sohn, 2016) is first proposed and widely applied in the computer vision community (He et al., 2020; Chen et al., 2020; Caron et al., 2020). Then researchers introduce contrastive learning into various natural language processing tasks, including sentence representation (Gao et al., 2021), event extraction (Wang et al., 2021), retrivel (Yang et al., 2023; Zhang et al., 2023b), vision-language pre-training (Radford et al., 2021; Singh et al., 2023), etc. To our best knowledge, we are the first to introduce contrastive learning into constituency parsing. Concretely, contrastive learning is a technique that pulls similar examples (positive instances) and pushes different examples (negative instances). In this work, we adopt contrastive learning on the span level to distinguish valid constituent spans and invalid spans, which can expand pre-training data and reduce the cost of LLM back generation significantly. Besides, contrastive learning usually builds positive instances by data augmentation (e.g., image rotation or crop, token dropout or sentence paraphrase), and negative instances by other examples in the same batch. However, our proposed strategy takes the left child, right child, parent and brother nodes as positive instances, and the fifteen corresponding invalid spans as negative instances.

3 Method

In this section, the process of LLM back generation (§ 3.1) is first introduced to generate the cross-domain constituency treebank. Then we describe the chart-based parser (Kitaev and Klein, 2018) briefly (§ 3.2). Finally, based on the parser, we propose the contrastive learning pre-training strategy (§ 3.3), which acquires a better constituent span representation model by the LLM back generation treebank for cross-domain constituency parsing.

3.1 LLM Back Generation

Constituency treebank $\{(X, Y)\}$ is crucial for a high-performance constituency parser, where X and Y are the input sentence with n words $X = x_1 \cdots x_n$ and the corresponding constituency parse tree, respectively. Treebank annotation can be extremely expensive and time-consuming, and only a few domains have large annotated treebanks. Although LLMs can substitute human labor to annotate datasets in some natural language processing tasks (Zhang et al., 2023a; Bansal and Sharma, 2023), their performance on elaborate structure tasks like constituency parsing is poor (Bai et al., 2023), underperforming conventional chart-based parsers. One possible reason may be that LLMs are trained for dialogue rather than structure extraction. Consequently, hallucinations in autoregressive generation make it difficult for the generated syntax trees to conform to the constraints of high-quality, valid tree structures. Additionally, LLMs can not find strict annotation specifications from limited demonstration samples. Therefore, we propose LLM back generation to alleviate these challenges in LLM treebank annotation. As illustrated in Figure 2, LLM back generation takes the incomplete cross-domain constituency tree as input and fills masked words, which can ensure syntactic structure validity from the input end.

Cross-domain constituency tree preparation. As the reference, constraint and guidance, the incomplete target domain constituency tree is crucial for the LLM to generate the effective full parse tree. Yang et al. (2022) indicate that both syntactic structure and domain vocabulary are important influence factors for cross-domain constituency parsing. Therefore, we prepare the target domain masked constituency parse tree from these two aspects by extracting the bare constituency tree of the target domain raw sentence (syntactic structure) and removing all sentential words except target domain keywords (domain vocabulary). Figure 2 shows the target domain masked constituency parse tree.

For cross-domain syntactic structure, we first train the state-of-the-art chart-based parser (Kitaev and Klein, 2018) and then parse the constituency tree corresponding to the unlabeled target domain raw sentence. Generally, the output constituency syntax tree will imply some cross-domain syntactical tree structures, since the raw sentence is derived from the target domain.

For domain vocabulary, we first extract domain keywords from the sentence based on the KeyBERT (Grootendorst, 2020), which computes all embedding similarities between words and the sen-

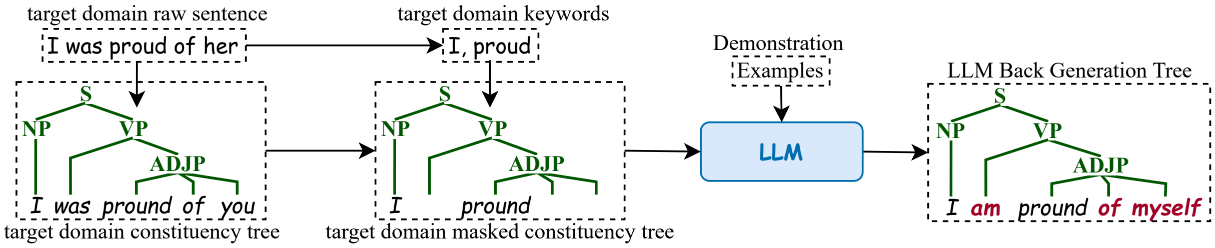


Figure 2: Overview of LLM Back Generation. We first extract the target domain constituency tree and domain keywords. Then we mask all sentential words except domain keywords from the constituency tree. Finally, the LLM back generates the whole syntax tree based on the masked tree and some demonstration examples.

tence and selects topK words as keywords. Then we retain 25% sentential words that are the most similar to the origin sentence as domain keywords and remove the left words from the constituency parse tree. In particular, our approach is equivalent to Li et al. (2023) when all words are reserved, while domain vocabulary can not be controlled when all words are masked.

LLM back generation. As shown in Figure 2, LLM back generation generates the full constituency syntax tree by in-context-learning (ICL) (Brown et al., 2020). First, the target domain incomplete constituency syntax tree is fed to the LLM. Second, we also input some demonstration examples, which help the LLM to better comprehend the treebank back generation task. Concretely, the demonstration example is a pair of masked and full constituency trees, which are derived from the target domain raw corpus. We append a detailed prompt string of LLM back generation in § A.1. Third, the LLM fills the masked syntax tree with appropriate words to generate a new sentence $\widehat{X}$ conforming the constituency parse tree structure Y, imitating demonstration D:

$$(\widehat{X}, Y) = \text{LLM}(Y, X^*, D),$$

where X^* denotes the masked sentence, and the tuple $(\widehat{X}, Y)$ is the LLM back generation parse tree, which implies domain characteristics.

3.2 Chart-based Constituency Parser

We briefly introduce the chart-based constituency parser (Kitaev and Klein, 2018), which is the foundation of our span-level contrastive learning pre-training. Concretely, the parser first adopts BERT (Devlin et al., 2019) to vectorize the input sentence, then encodes the context information by a partitioned transformer, and computes span representations $\boldsymbol{r}$ with a span encoder [2]. Subsequently, a multi-layer perceptron assigns a score $s(i, j, l)$ to each labeled span, which represents the score of the span as a constituent with the syntactic label l. Finally, the score of the constituency tree $s(T)$ is computed by summing the scores of all the labeled spans within it. Particularly, the chart-based parser exploits the CKY algorithm to efficiently search for the syntax tree with the highest score as the predicted output $\widehat{T}$. For training, the tree-based max-margin loss minimizes the difference between the gold-standard tree T^* and the predicted tree $\widehat{T}$:

$$\mathcal{L} = s(\widehat{T}) - s(T^*) + \varDelta(\widehat{T}, T^*),$$

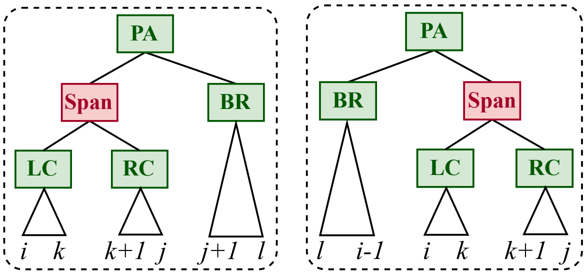
where $\varDelta$ represents the Hamming difference.

3.3 Contrastive Learning Pre-training

In this subsection, we present our span-level contrastive learning pre-training strategy, which can make full use of the LLM back generation treebank. For one thing, as pre-training is robust for the noise in the labeled datasets, we attempt to utilize the LLM back generation treebank to pre-train a remarkable constituent span representation model. For another thing, the size of the LLM back generation treebank is limited due to the cost of LLM generation, thus our contrastive learning strategy is based on span-level not example-level, which can significantly expand pre-training data. Specifically, we start with the introduction of the positive and negative instances for each constituent span (i, j) in the LLM back generation tree. The goal of the chart-based parser is to distinguish all valid constituent spans, thus the positive and negative instances are valid and invalid spans naturally. Then, the contrastive constituent representation model is presented based on them.

Positive instances. Non-local high-order features of upper and lower constituent nodes are essential for constituent recognition in the process of

[2] Parser details refer to Kitaev and Klein (2018).



(a) The constituent span as left sub-tree.

(b) The constituent span as right sub-tree.

Figure 3: Positive instances (green node) and indexes for the constituent span (red node). LC, RC, PA and BR are left child, right child, parent, and brother, respectively.

PosInst	NegInst	
	left sub-tree	right sub-tree
Span	(i, j±1), (i±1, j), (i±1, j±1)	
LC	(i, k-1), (i, k+1)	
RC	(k, j), (k+2, j)	
PA	(i, l-1), (i, l+1)	(l-1, j), (l+1, j)
BR	(j, l)	(l, i)

Table 1: Negative instances for the constituent span.

building a hierarchical parse tree (Cui et al., 2022; Shi et al., 2022). As illustrated in Figure 3, we select the left child (LC), right child (RC), parent (PA) and brother (BR) nodes as the positive instances for each constituent span. Particularly, a constituency tree is a binary tree after binarization in the chart-based parser, so the constituent span is either a left sub-tree (Figure 3a) or a right sub-tree (Figure 3b). When the constituent span (i, j) is a left sub-tree, the set of the positive instances $(i, j)^+$ is $\{(i, k), (k{+}1, j), (i, l), (j{+}1, l)\}$. When the constituent span (i, j) is a right sub-tree, the set of the positive instances $(i, j)^+$ is $\{(i, k), (k + 1, j), (l, j), (l, i - 1)\}$.

Negative instances. In fact, span recognition in the chart-based parser is essentially boundary recognition. Invalid spans with similar boundaries can impact the accuracy of recognition on constituent span (i, j), such as $(i - 1, j - 1)$ and $(i + 1, j + 1)$. Therefore, for each positive instance and constituent span itself, we mine corresponding negative instances, which are listed in Table 1. Specifically, the set of negative instances $(i, j)^-$ encompasses fifteen spans for constituent span (i, j). When the constituent span is a left or right sub-tree, twelve negative instances are the same but three are different. Specially, some negative instances may be valid constituent spans in the constituency parse tree. We filter these negative instances because we only need to pull all valid spans together and push all invalid spans apart. Otherwise, it will destroy the contrastive constituent representation model.

Contrastive constituent representation model. Our contrastive constituent representation model is the same as the chart-based parser (§ 3.2), but replaces the multi-label classification layer and max-margin tree loss with contrastive objective only. Therefore, after the contrastive constituent representation model is pre-trained on the LLM back generation treebank, we can transfer it into the constituency parser easily. For each span with the start index i and end index j, our contrastive objective $\mathcal{L}$ is based on its span representation $\boldsymbol{r}$:

$$\mathcal{L} = -\sum_{m\in(i,j)^+} \log \frac{e^{f(\boldsymbol{r},\boldsymbol{r}_m^+)}}{\sum\limits_{n\in(i,j)^-} e^{f(\boldsymbol{r},\boldsymbol{r}_m^+)} + e^{f(\boldsymbol{r},\boldsymbol{r}_n^-)}},$$

where f indicates the cosine similarity function divided by temperature factor τ.

We pre-train BERT (Devlin et al., 2019), partitioned transformer and span encoder by span-level contrastive learning on the LLM back generation treebank, and get the contrastive constituent representation model, which can generate a better span representation for cross-domain constituency parsing. After pre-training, we equip the constituent representation model with max-margin tree loss and fine-tune it on the combination of the limited source domain human annotation treebank and the target domain LLM back generation treebank.

4 Experiments

4.1 Experimental Setup

Datasets and Hyperparameters. Following Li et al. (2023), we use MCTB (Yang et al., 2022) as the target constituency parsing dataset, which includes five target domains: dialogue (Dia), forum (For), law, literature (Lit) and review (Rev). Based on `gpt-4-1106-preview` (OpenAI, 2023), we generate 10,000 constituency trees as the LLM back generation treebank containing the above five domains. We attempt other large language models (e.g., ChatGPT and Llama-3) as well, but they mostly either fail to generate constituency trees or produce trees with errors. Table 4 in A.3 reports their results. Besides, the other dataset details and

Method	Option	Dia	For	Law	Lit	Rev	Avg.
Large Language Models							
ChatGPT	full	30.54	18.86	24.93	11.96	28.67	22.99
	valid	70.38	70.36	80.70	74.74	69.08	73.05
GPT-4	full	37.89	25.73	31.25	18.06	33.71	29.33
	valid	77.64	76.27	84.49	79.58	75.63	78.72
Ours							
Natural Corpus Treebank	DAPT	86.25	87.04	92.19	86.42	83.86	87.15
	NOPT	86.54	87.47	92.26	86.64	83.98	87.38
	CTPT	87.33	87.80	92.54	86.91	84.35	87.79
LLM Back Generation Treebank	DAPT	86.50	87.28	92.43	86.71	84.02	87.39
	NOPT	87.75	87.43	92.57	87.01	84.28	87.81
	CTPT	**87.92**	**88.13**	93.22	87.50	**85.86**	**88.52**
Previous Work							
Kitaev and Klein (2018)	–	86.10	86.92	92.07	86.28	84.32	87.14
Liu and Zhang (2017)	–	85.56	86.33	91.50	84.96	83.89	86.45
Li et al. (2023)	–	87.59	87.55	**93.29**	**87.54**	85.58	88.31

Table 2: Main results on MCTB benchmark. DAPT, NOPT and CTPT are short for domain adaptive pre-training, no pre-training, and our contrastive learning pre-training, respectively.

the hyperparameters of proposed contrastive learning are also placed in A.2 for limited space.

Evaluation. F1 score of labeled bracketed spans is used to evaluate the performance of cross-domain constituency parsing. We conduct the experiments on three different random seeds and report the average results, ignoring punctuations following Kitaev and Klein (2018) and Li et al. (2023).

Baselines. We compare our approach with various constituency parsers: (1) a strong Transition-based constituency parser (Liu and Zhang, 2017), (2) a strong chart-based constituency parser (Kitaev and Klein, 2018), which is re-implemented by us as the basic constituency parser and (3) a state-of-the-art cross-domain constituency parsing method (Li et al., 2023), which utilizes the LLM to generate unlabeled raw sentences in the target domain.

We also report the cross-domain constituency parsing performances of *ChatGPT* (Brown et al., 2020; Ouyang et al., 2022) and *GPT-4*. We use `gpt-3.5-turbo` and `gpt-4` to generate bracketed parse trees with in-context-learning (Brown et al., 2020), where 10 constituency tree examples from the source treebank PTB are prepended before the testing instance as demonstrations. Notably, the outputs can contain numerous errors, including unmatched brackets, omitted words from input sentences, and responses lacking bracketed parse trees, because LLMs predict the next token auto-regressively and do not guarantee the validity of generated constituency trees. We report results both considering and not considering invalid trees in the lines of "full" and "valid" in Table 2.

For the contrastive learning pre-training (CTPT) strategy, we compare it with two pre-training methods: (1) no pre-training (NOPT) directly fine-tunes the chart-based parser on the source treebank and LLM back generation treebank. (2) domain adaptive pre-training (DAPT) (Gururangan et al., 2020) continues pre-training BERT from `BERT-large-uncased` on the target domain raw corpus, which comprises 500k sentences in the five target domains, 100k sentences for each domain.

In order to verify the effectiveness of our LLM back generation treebank, we conduct a contrast experiment on the *Natural Corpus Treebank*, which is parsed from the natural target domain raw corpora by a basic parser (Kitaev and Klein, 2018).

4.2 Main Results

Table 2 reports F1 scores of different cross-domain constituency parsing methods on MCTB, which consists of five targe domains.

First, we examine the parsing performances of large language models. *ChatGPT* and *GPT-4* show poor performance on all domains, which suggests that such generative LLMs can be less capable of solving cross-domain constituency parsing. Besides, we find that ChatGPT and GPT-4 tend to generate invalid parse trees. Take the input sentence "*He is right .*" for example, LLMs might generate unmatched brackets (e.g., *"[S [NP [PRP He]] [VP [VBD was] [ADJP [JJ right] [. .]"*) or drop sentential words (e.g., *"[S [VP [VBD was] [ADJP [JJ right]]] [. .]]"*). Therefore, when taking all outputs (the second line "full") into evaluation, their performances decrease severely. The poor parsing results and invalid parse tree prove that it is hard for LLMs to annotate treebank directly.

Second, we look at the performances of LLM back generation treebank and natural corpus treebank. As shown in Table 2, our LLM back generation treebank can boost the parsing results significantly, leading the gains on the average F1 score by $87.39 - 87.15 = 0.24$, $87.81 - 87.38 = 0.67$ and $88.52 - 87.79 = 0.73$ for three pre-training methods, respectively. Regardless of which pre-training method is used, the performance of our LLM back generation treebank is significantly superior to that of natural corpus treebank, which demonstrates the effectiveness of LLM back generation treebank.

Third, we observe the results of different pre-training strategies. DAPT is inferior to NOPT and close to basic chart-based parser (Kitaev and Klein, 2018), which is trained only in the source treebank. One reason could be that the pre-training format of masked language modeling is far from constituency parsing. Although the model is trained on the target domain corpora, the knowledge is hard to transfer to constituent recognition. CTPT significantly improves the parsing performance across five domains compared with DAPT and NOPT. The observation suggests that our span-level contrastive learning pre-training effectively acquires constituent knowledge from the LLM back generation treebank.

Finally, we make comparisons with the previous work. Our LLM back generation treebank coupled with span-level contrastive learning pre-training obtains state-of-the-art cross-domain parsing results on the average F1 score. Compared with raw corpus generation from LLM (Li et al., 2023), our approach gains the stronger average F1 score, which shows the effectiveness of LLM back generation treebank generation. For the dialogue, forum and review domain, we perform a statistical significance analysis between our approach and Li et al. (2023), where $p < 0.05$ verifies the effectiveness of our proposed method. For the law and literature domain, we suspect the reason for lower results might be differences in domain distribution and sentence length. The law and literature domains usually involve more formal and official expressions and long sentences, while more colloquial and short sentences exist in the dialogue, forum and review domains. For one thing, improving the performance of long sentences may be harder than short sentences. For another thing, our method handles all five domains by only one parser, which needs to balance different domain differences, however, Li et al. (2023) train a separate parser for each domain, which only considers sentences from the same domain with a similar distribution. More analyses and experiments are in A.4.

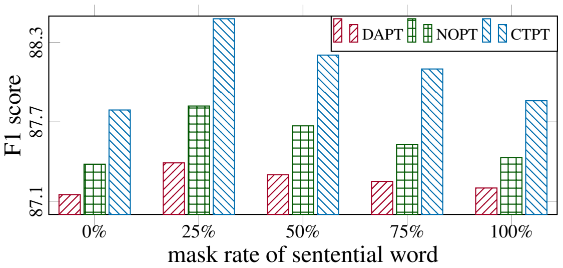


Figure 4: F1 score of three pre-training strategies on the LLM back generation treebanks with different mask rates of sentential word.

4.3 Analyses

We conduct detailed experimental analyses in this subsection to offer several findings of our LLM back generation treebank and span-level contrastive learning pre-training.

Mask rate for LLM back generation treebank. The mask rate of sentential words in the target constituency tree may influence LLM back generation treebank generation, which is the core of our approach. We conduct an experiment to examine the relation between cross-domain constituency parsing performance and mask rate for LLM back generation treebank. Figure 4 shows the results based on the average F1 score of five target domains. When the mask rate is 0%, all sentential words are reserved, thus it is the natural corpus treebank in fact. When the mask rate is 100%, all sentential words are masked, which exploits LLM to generate a treebank based on the bare constituency tree only.

First, we can see that all LLM back generation treebanks including 25%, 50% and 100% mask rates outperform the natural corpus treebank. This phenomenon shows the effectiveness of our LLM back generation treebank. Second, the 25% mask rate achieves the best F1 score. As the mask rate becomes larger, the cross-domain parsing perfor-

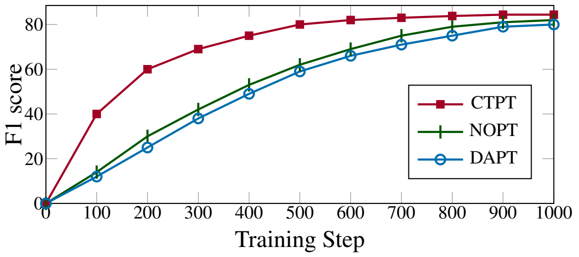


Figure 5: Convergence curve of different pre-training strategies on the LLM back generation treebank.

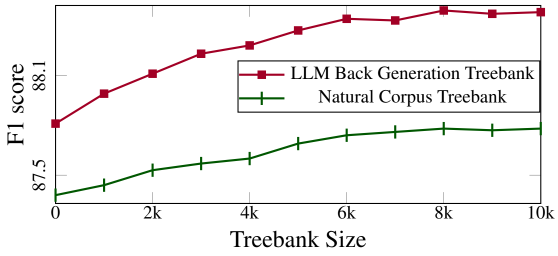


Figure 6: F1 score of contrastive learning pre-training on natural corpus treebank and LLM back generation treebank with respect to the size of treebank.

mance decreases gradually. A reasonable explanation might be that fewer retained domain keywords cause LLM generation to be freer. Therefore, the final LLM back generation treebank will shift from the target domain. Third, our contrastive learning pre-training strategy significantly improves the results compared with DAPT and NOPT in all settings, which verifies its effectiveness.

Convergence for contrastive learning pre-training. Contrastive learning pre-training is also one of the key contents of this paper. We find that convergences of different pre-training strategies are different in the fine-tuning phase. Specifically, we evaluate the cross-domain constituency parsing performance every 100 training steps when pre-trained models are fine-tuned on the constituency treebank. The results are illustrated in Figure 5, where the x-axis denotes the training step and the y-axis denotes the average F1 score on all five target domains. When the training step is zero, the F1 scores of three pre-training methods are zero, because the parsers are only pre-trained not fine-tuned on the treebank. As the training step grows larger in the initial phase, the three curves increase significantly. Afterward, the performance stops increasing and gradually converges.

Figure 5 illustrates that our contrastive learning pre-training converges significantly faster compared with domain adaptive pre-training and no pre-training. Concretely, CTPT converges at 600 training steps while DAPT and NOPT stop increasing after 1000 training steps. Besides, the curve of CTPT not only improves fast but also gains the best results finally. The observation suggests that our span-level contrastive learning pre-training strategy pre-trains the contrastive constituency representation model that can compute better span representations. Our proposed pre-training acquires and transfers the knowledge of constituent recognition into cross-domain constituency parsing successfully.

Contrastive learning pre-training treebank size. Intuitively, the final average cross-domain constituency parsing results (y-axis) should be dependent on the number of constituency trees extracted from natural corpus or LLM back generation (x-axis). We conduct a experiment to observe the performance differences with respect to the size of contrastive learning pre-training treebank. Figure 6 demonstrates the results. When treebank size is 0, no constituency trees are used in contrastive learning pre-training, which is equivalent to NOPT. The cross-domain parsing performance would stop increasing after 8k for both natural corpus treebank and LLM back generation treebank. Although the size of pre-training treebank is limited, our proposed span-level contrastive learning pre-training can expand the limited sentence-level constituency trees to a large number of span-level spans. Concretely, the average number of spans in the LLM back generation treebank is about 25. Thus, our span-level contrastive learning pre-training is performed on $25 * 10,000 = 250,000$ examples, which significantly reduces the cost and scale of the LLM back parsing generation treebank.

5 Conclusion

In this paper, we proposed LLM back generation, which takes a masked cross-domain constituency tree as the input instruction and fills in missing words to generate the LLM back generation treebank. Besides, we presented a span-level contrastive learning pre-training strategy for the LLM back generation treebank. To the best of our knowledge, this is the first work to introduce contrastive learning into constituency parsing. Experimental results showed that our LLM back generation treebank coupled with span-level contrastive learning pre-training gains state-of-the-art results on averaged F1 score compared with various baselines, including natural corpus treebank, conventional parsers, masked language modeling pre-training, previous cross-domain constituency parsing methods and large language models.

Limitations

Our approach is verified on the only English dataset, so its effectiveness is not clear for the other languages. To the best of our knowledge, there are no available cross-domain constituency treebanks for other languages, as dataset annotation is time-consuming and expensive, especially for parsing tasks. So it is difficult to perform experiments on diverse languages to verify the effectiveness of the proposed method. In fact, our approach can be applied directly to any language, and fortunately, GPT-4 is qualified in most languages. In addition, when developers face a specific language, we also suggest trying other representative and popular LLMs for this language, such as Qwen or DeepSeek for Chinese, HyperCLOVA-X for Korean, and so on.

Acknowledgements

We sincerely thank the reviewers for their invaluable feedback, especially Reviewer N7Sb in ARR 2024 December. This work was supported by the National Natural Science Foundation of China under Grant 61976180 and 62176180, and the Shenzhen Science and Technology Program under Grant GXWD20231130140414001 and ZDSYS20230626091203008.

References

Xuefeng Bai, Jialong Wu, Yulong Chen, Zhongqing Wang, and Yue Zhang. 2023. Constituency parsing using llms. *arXiv*.

Parikshit Bansal and Amit Sharma. 2023. Large language models as annotators: Enhancing generalization of nlp models at minimal cost. *arXiv*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in neural information processing systems*.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2020. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*.

Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*.

Leyang Cui, Sen Yang, and Yue Zhang. 2022. Investigating non-local features for neural constituency parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Daniel Fried, Nikita Kitaev, and Dan Klein. 2019. Cross-domain generalization of neural constituency parsers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Maarten Grootendorst. 2020. Keybert: Minimal keyword extraction with bert.

Peiming Guo, Meishan Zhang, Yulong Chen, Jianling Li, Min Zhang, and Yue Zhang. 2024. Cross-domain constituency parsing by leveraging heterogeneous data. *Journal of Artificial Intelligence Research*, 81:771–791.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Vidur Joshi, Matthew Peters, and Mark Hopkins. 2018. Extending a parser to distant domains using a few dozen partially annotated examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Jianling Li, Meishan Zhang, Peiming Guo, Min Zhang, and Yue Zhang. 2023. LLM-enhanced self-training for cross-domain constituency parsing. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Jiangming Liu and Yue Zhang. 2017. In-order transition-based constituent parsing. *Transactions of the Association for Computational Linguistics*.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*.

David McClosky, Eugene Charniak, and Mark Johnson. 2008. When is self-training effective for parsing? In *Proceedings of the 22nd International Conference on Computational Linguistics*.

David McClosky, Eugene Charniak, and Mark Johnson. 2010. Automatic domain adaptation for parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

OpenAI. 2023. Gpt-4 technical report. *arXiv*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Tianyu Shi, Zhicheng Wang, Liyin Xiao, and Cong Liu. 2022. Fast rule-based decoding: Revisiting syntactic rules in neural constituency parsing. *arXiv*.

Harman Singh, Pengchuan Zhang, Qifan Wang, Mengjiao Wang, Wenhan Xiong, Jingfei Du, and Yu Chen. 2023. Coarse-to-fine contrastive learning in image-text-graph space for improved vision-language compositionality. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A minimal span-based neural constituency parser. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

Øyvind Stiansen and Erik Voeten. 2019. *ECtHR judgments*. Harvard Dataverse.

Zhiyang Teng and Yue Zhang. 2018. Two local models for neural constituent parsing. In *Proceedings of the 27th International Conference on Computational Linguistics*.

Yuanhe Tian, Yan Song, Fei Xia, and Tong Zhang. 2020. Improving constituency parsing with span attention. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Petter Törnberg. 2023. Chatgpt-4 outperforms experts and crowd workers in annotating political twitter messages with zero-shot learning. *arXiv preprint arXiv:2304.06588*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint*.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. 2017. TL;DR: Mining Reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*.

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juanzi Li, and Jie Zhou. 2021. CLEVE: Contrastive Pre-training for Event Extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Junhan Yang, Zheng Liu, Chaozhuo Li, Guangzhong Sun, and Xing Xie. 2023. Longtriever: a pre-trained long text encoder for dense document retrieval. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Sen Yang, Leyang Cui, Ruoxi Ning, Di Wu, and Yue Zhang. 2022. Challenges to open-domain constituency parsing. In *Findings of the Association for Computational Linguistics: ACL 2022*.

Ruoyu Zhang, Yanzeng Li, Yongliang Ma, Ming Zhou, and Lei Zou. 2023a. LLMaAA: Making large language models as active annotators. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.

Xin Zhang, Zehan Li, Yanzhao Zhang, Dingkun Long, Pengjun Xie, Meishan Zhang, and Min Zhang. 2023b. Language models are universal embedders. *arXiv*.

Yu Zhang, Houquan Zhou, and Zhenghua Li. 2020. Fast and accurate neural crf constituency parsing. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*.

A Appendix

A.1 LLM Back Generation

Figure 7 displays the prompt and output of LLM back generation. Concretely, we first prompt the LLM as a professional linguist in the system prompt. Then, the question and answer of the demonstration are appended to the system prompt. The former is the incomplete constituency tree with both the syntactical structure and domain keywords of the target domain, and the latter is the original target domain full constituency syntax tree. Following, the masked constituency tree to be processed is placed at the end of the input instruction. Finally, the LLM fills the incomplete syntax tree with appropriate words to generate a new sentence conforming to the constituency parse tree structure, imitating the demonstration.

A.2 Datasets and Hyperparameters

We use PTB (Marcus et al., 1993) and MCTB (Yang et al., 2022) as the source and target constituency parsing datasets, respectively. Dataset statistics are listed in Table 3, where #Sent and AS denote the number of sentences and average number of spans in binary constituency parse trees. For the five target domain raw corpora, we collect unlabelled sentences with sources matching the corresponding target treebank in MCTB, including Wizard (Dinan et al., 2019), Reddit (Völske et al., 2017), ECtHR (Stiansen and Voeten, 2019), Gutenberg[3], and Amazon (He and McAuley, 2016).

For LLM back generation treebank generation, we set 2 demonstrations in the prompt for LLM back generation, which are placed in the `name` field of the system message. For postags in unlabeled raw sentences, we train a BERT-LSTM-CRF model for prediction. The training datasets are extracted from constituency treebank PTB (Marcus et al., 1993) directly. For keyword extraction, we use the default word representation model (i.e., `all-MiniLM-L6-v2`) of the KeyBERT python library (Grootendorst, 2020). We randomly selected the sentential words to mask the sentence in preliminary experiments, but the results were limited as random masking would hurt the target vocabulary domain characteristics. For contrastive learning pre-training, we use BERT-large-uncased as the pretrained language model backbone (Devlin et al., 2019) for the cross-domain constituency parser and the other hyperparameters are the same as Kitaev and Klein (2018). We use the AdamW algorithm with learning rate 3e-5, batch size 64 to pre-train LLM back generation treebank for 10 epochs. For each constituency tree in a batch, only 20% constituents are sampled as examples to compute the contrastive learning loss. Temperature factor τ is 0.05. For constituency parsing fine-tuning, we use the AdamW algorithm with learning rate 1e-5, batch size 64, and linear learning rate warmup over the first 400 steps to optimize parameters. We stop early training when the F1 score does not increase on the PTB development set for 4 epochs. We attempt to mix the standard fine-tuning loss and the contrastive learning pre-training loss in preliminary experiments. But there were no improvements and the training would unstable. So we use the standard loss only during the final fine-tuning stage. We run all experiments on a single Nvidia V100.

A.3 Results of Different LLMs

Generally, the quality of LLM back generation treebank is related to the used LLM. We perform a experiment to analyse the results of different LLMs, which are reported in Table 4. We adapt extra

[3] https://www.gutenberg.org/

LLM Back Generation Prompt

System Prompt:

You are a professional linguist.

Demonstration:

(SQ (VBP ***Have***) (NP (PRP)) (ADVP (DT)) (VP (VBN) (NN ***skiing***)))

(SQ (VBP ***Have***) (NP (PRP ***you***)) (ADVP (DT ***ever***)) (VP (VBN ***gone***) (NN ***skiing***)))

Masked Constituency Tree:

(S (NP (PRP ***I***)) (VP (VBD) (ADJP (JJ ***proud***) (PP (IN) (NP (PRP))))))

LLM Back Generation Output

LLM Back Generation Tree:

(S (NP (PRP ***I***)) (VP (VBD ***am***) (ADJP (JJ ***proud***) (PP (IN ***of***) (NP (PRP ***myself***))))))

Figure 7: The prompt and output of LLM back generation. The blue texts are only shown for illustrative purposes and not in the actual prompt.

Dataset	Domain	#Sent	AS
PTB	news	39,832	25.54
MCTB	dialogue	1,000	15.92
	forum	1,000	26.33
	law	1,000	27.47
	literature	1,000	26.91
	review	1,000	15.19

Table 3: Datasets statistics.

two representative large language models, ChatGPT and Llama-3-8B (Touvron et al., 2023), to produce LLM back generation treebanks. Then the proposed span-level contrastive learning strategy is conducted on the LLM back generation treebanks. For the close-source model, ChatGPT generates more constituency trees with syntax errors. For the open-source model, Llama-3 usually refuses to generate constituency trees, which may be short of pre-trained knowledge of constituency parsing. Therefore, we fine-tune Llama-3 by LoRA (Hu et al., 2022) on the target domain corpora with the task format in Figure 7. Concretely, ChatGPT and Llama-3 achieve average F1 scores of 88.28 and 88.05 respectively, underperforming GPT-4. The large language model with stronger understanding and generation capabilities can generate high-quality treebanks, leading to better cross-domain constituency parsing performance.

A.4 Performance Comparison with Previous Method

The method of Li et al. (2023) is closely related to ours, therefore we conduct detailed analyses and experiments for comparison here. Compared with Li et al. (2023), our method has other significant advantages except for a higher average parsing score in Table 2. For one thing, our method needs only 10,000 sentences generated by LLMs, but Li et al. (2023) use ChatGPT to produce 200,000 sentences. The size of the generated corpus is 20 times more than ours, which leads to an unaffordable and unavoidable high cost. We guess it is an important reason why they do not apply more powerful GPT-4 to generate sentences. For another thing, our parser can handle sentences from all five domains simultaneously. But Li et al. (2023) train a separate parser for each domain. In a real application scenario, our method deploys one model to parse different domains, while Li et al. (2023) need more models. This will waste valuable GPU resources.

Besides, a fair quantitative comparison between our method and Li et al. (2023) is important. We run our model on the same settings as Li et al. (2023): 1) using GPT-3.5-turbo, 2) scaling to 200,000 sentences, and 3) training a parser for each domain. Table 5 shows the experimental results. Our method achieves better parsing performances on five domains, which verifies the effectiveness of our proposed LLM back generation treebank and span-level contrastive learning pretraining. Besides, we also conduct statistical significance test experiments, where $p < 0.05$ shows the stability of experiments.

LLM	Access	Dia	For	Law	Lit	Rev	Avg.
GPT-4	close	**87.92**	**88.13**	**93.22**	**87.50**	**85.86**	**88.52**
ChatGPT	close	87.63	87.91	93.03	87.29	85.56	88.28
Llama-3	open	87.39	87.65	92.80	87.04	85.38	88.05

Table 4: Results on different LLMs.

Method	Dia	For	Law	Lit	Rev	Avg.
Li et al. (2023)	87.59	87.55	93.29	87.54	85.58	88.31
Ours	**87.70**	**87.76**	**93.33**	**87.62**	**85.69**	**88.42**

Table 5: Result comparison with Li et al. (2023) in the same settings.

Model	Base chart parser	Our final method
F1 score	95.64	95.71

Table 6: Results on the source domain.

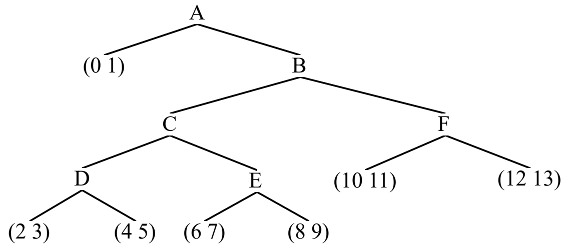


Figure 8: A constituency tree.

A.5 Performance on Source Domain

Though our approach achieves impressive results on target domain, performances on source domain standard benchmarks are also important. We test our final parser on the PTB dataset, which achieves an F1-score of 95.71 as shown in Table 6. The result of the basic chart-based parser is 95.64, which is trained on the source domain PTB dataset. The comparison verifies that our parser can maintain or even improve the performance on the standard benchmark.

A.6 Example of Positive and Negative Instances

Given a constituency tree as illustrated in Figure 8, uppercase letters (i.e., A, B, C, ...) are syntax labels, and numbers (i.e., 1, 2, 3, ...) are words. Take the constituent node (C, 2, 9) for example, the positive instances are the left child (2, 5), right child (6, 9), parent (2, 13) and brother (10, 13). The negative instances are invalid spans related to it: (1, 9), (3, 9), (2, 8), (2, 10), (1, 10), (3, 8), (1, 8), (3, 10), (2, 4), (2, 6), (5, 9), (7, 9), (2, 12), (2, 14), (9, 13).