Multi-features Enhanced Multi-task Learning for Vietnamese Treebank Conversion

Zhenguo Zhang[1,2], **Jianjian Liu**[1,2], **Ying Li**[1,2*]
[1]Yunnan Key Laboratory of Artificial Intelligence, Kunming, China
[2]Faculty of Information Engineering and Automation,
Kunming University of Science and Technology, Kunming, China
zzg@stu.kust.edu.cn,1807164254@qq.com,yingli_hlt@foxmail.com

Abstract

Pre-trained language representation-based dependency parsing models have achieved obvious improvements in rich-resource languages. However, these model performances depend on the quality and scale of training data significantly. Compared with Chinese and English, the scale of Vietnamese Dependency treebank is scarcity. Considering human annotation is labor-intensive and time-consuming, we propose a multi-features enhanced multi-task learning framework to convert all heterogeneous Vietnamese Treebanks to a unified one. On the one hand, we exploit Tree BiLSTM and pattern embedding to extract global and local dependency tree features from the source Treebank. On the other hand, we propose to integrate these features into a multi-task learning framework to use the source dependency parsing to assist the conversion processing. Experiments on the benchmark datasets show that our proposed model can effectively convert heterogeneous treebanks, thus further improving the Vietnamese dependency parsing accuracy by about 7.12 points in LAS.

1 Introduction

Dependency parsing, as a fundamental task of natural language processing, aims at capturing syntactic and semantic information between two words through a dependency tree. The top of Figure 1 shows a dependency tree from the universal dependency treebank, where the directed edge from the headword "mừng (happy)" to the modified word "mà (and)" is a dependency arc, the label "mark" on the arc indicates the type of dependency relation between two words. Recently, dependency parsing has gained more attention due to its simple and understandable structure. The results of dependency parsing have been applied to various artificial intelligence tasks, such as grammatical error correction (Li et al., 2022), machine translation (Pu and Sima'an, 2022), and semantic role labeling (Zhang et al., 2022).

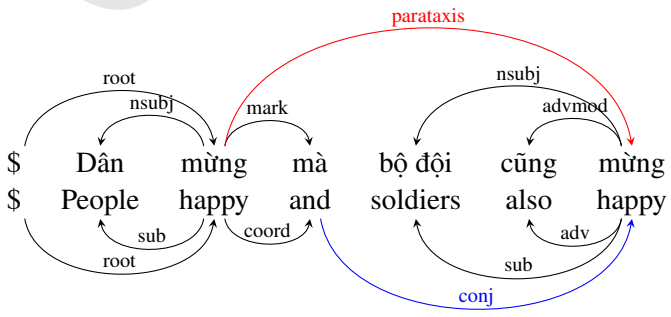

Figure 1: Example of treebank conversion from the target-side UD tree (upper) to the source-side VnDT tree (under).

Proceedings of the 23rd China National Conference on Computational Linguistics, pages 1035-1046, Taiyuan, China, July 25 - 28, 2024.
Volume 1: Main Conference Papers
(c) Technical Committee on Computational Linguistics, Chinese Information Processing Society of China

With the development of neural networks and pre-trained language technologies, dependency parsing models have achieved significant progress in rich-resource languages. Currently, the supervised dependency parsing model has obtained 96.4 LAS and 97.4 UAS on English Penn Treebank (Amini et al., 2023), and 92.5 LAS and 93.5 UAS on Chinese Penn Treebank (Yang and Tu, 2022). However, all these supervised parsing models are highly dependent on the quality and scale of training data. For low-resource languages like Vietnamese, the performance and efficiency of the parsing model are not ideal because of the scarcity of Vietnamese annotation corpora. To address this issue, we first survey the existing Vietnamese dependency treebanks, which include Universal Dependencies (UD)[0], Vietnamese Dependency Treebank (VnDT) (Nguyen et al., 2014a), and Vietnamese Language and Speech Processing (VLSP2020)[1]. However, since these treebanks adopt different annotation standards, it is impossible to fuse all treebanks directly for traditional paring model training due to the potential semantic conflicts. As shown in Figure 1, the upper tree is generated based on the UD standard, while the bottom tree is from VnDT. We can see that even though the two dependency trees have the same input sentence, the tree structures still have some differences in arcs and labels. This study terms such treebanks with different standards or linguistic theories as heterogeneous treebanks. Hence, how to convert the heterogeneous treebanks to a unified one is the main challenge to improving the performance of Vietnamese dependency parsing. The key to addressing this problem is to model the mappings between different treebanks. In the past few years, some researchers have attempted to perform conversion between heterogeneous treebanks through rule-based methods (Stymne et al., 2018; Borges Völker et al., 2019; Arnardóttir et al., 2020). However, most of these conversions focus on bi-tree alignment, ignoring the information in the source dependency tree. Hence, we propose a multi-features enhanced multi-task learning method to convert other treebanks into the UD standard. Concretely, we first use the pre-trained language model XLM-RoBARTa to enhance the basic dependency parsing performance. Then, we exploit pattern embedding and Tree BiLSTM to extract the local and global features from the source treebank. Next, both features are fused into a multi-task learning framework to further enhance the capability of conversion. Finally, all converted trees are combined with the original UD for the XLM-RoBARTa-enhanced basic model training.

Experimental results demonstrate that our proposed model can effectively model the mapping between heterogeneous treebanks, and the converted treebank is extremely useful for improving the accuracy of Vietnamese dependency parsing. Detailed analysis shows that both pattern embedding and Tree BiLSTM are helpful in encoding the information of the source tree, thus improving the conversion performance. The comparison experiments indicate that the converted treebank is an important factor in boosting the Vietnamese dependency parsing.

2 Related work

2.1 Dependency Parsing

In the field of dependency parsing, the prevailing methods are graph-based and transition-based dependency parsing. Graph-based dependency parsing has achieved state-of-the-art performance in various languages. With the development of neural networks and pre-trained language models, an increasing number of researchers and developers are applying these technologies to dependency parsing models. These models have a strong capability to understand complex linguistic structures, achieving significant progress in dependency parsing tasks. Dozat and Manning (2017) proposed a biaffine parser model, which uses a BiLSTM for encoder layer. Li et al. (2018) proposed the first end-to-end model for dependency parsing. Kondratyuk and Straka (2019) developed a multilingual multi-task self-attention network fine-tuned on BERT, achieving competitive results in multiple languages. Li et al. (2020) employed a bi-directional LSTM-CNN architecture for context encoding and deep contextually related word representations (ELMo and BERT) to further enhance model performance. Sharefah et al. (2023) delved into fine-tuning the BERT model for Arabic dependency parsing, providing insights for fine-tuning BERT models for specific NLP tasks.

[0] https://universaldependencies.org/
[1] https://vlsp.org.vn/vlsp2020/eval/udp

Proceedings of the 23rd China National Conference on Computational Linguistics, pages 1035-1046, Taiyuan, China, July 25 - 28, 2024.
Volume 1: Main Conference Papers

Vietnamese is part of the Austro-Asiatic language family and is one of the 20 most commonly used languages worldwide. Vietnamese vocabulary contains a large number of Sino-Vietnamese words derived from Chinese. This unique language structure and vocabulary combination makes Vietnamese an interesting research object in the field of NLP. Especially in the field of Vietnamese dependency parsing, significant research progress has been made in recent years. Do et al. (2022) proposed a cross-linguistic model adaptation using English to aid Vietnamese parsing, enhancing accuracy. Thien et al. (2020) combined BiLSTM and GNN. Le-Hong and Cambria (2024) integrated graph embedding with bidirectional recurrent neural networks for better performance. Nguyen and Nguyen (2020) Nguyen and Nguyen (2020) developed PhoBERT, a BERT-based monolingual language model for Vietnamese, showing excellence in POS tagging, dependency parsing, named-entity recognition. Trinh et al. (2022) integrated PhoBERT with cross-view training to enhance performance using labeled and unlabeled data.

2.2 Treebank Conversion

In the early stages of treebank conversion research, manual and rule-based methods were predominant, especially for widely used languages. These approaches relied on linguists and researchers meticulously developing sets of rules to convert syntactic structures and annotations from one standard to another (Frank, 2001; Avgustinova and Zhang, 2010; Candito et al., 2010). Li et al. (2013) used an annotation adaptation algorithm to convert component dependency tree banks from one annotation guideline to another. Nguyen et al. (2014b) designed a heuristic rule to convert Vietnamese component treebank into dependency treebank. Tran and Miyao (2022) Relying on hand-coded rules to assign categories to constituents convert Universal Dependencies (UD) treebanks to Combinatory Categorial Grammar (CCG) treebanks. However, these rule-based methods faced limitations when applied to languages with fewer resources, like Vietnamese, where treebanks are smaller and annotated data is limited, making it challenging to develop comprehensive rule sets.

In recent years, with the advancement of machine learning technologies, learning-based methods have emerged as a new trend in the field of treebank conversion. These methods, by learning the characteristics of different datasets, are more effective in handling the complexities of languages and data (Truong et al., 2022; Wang et al., 2022). The core of treebank conversion is to find the mapping relationship between the source treebank and the target treebank. Therefore, how to effectively utilize the syntactic information of the source treebank is the key to the problem. Jiang et al. (2018) constructed a supervised transformer by manually annotating a certain scale of dual-tree alignment data and using pattern embedding and TreeLSTM to encode the source-side trees. Wan et al. (2022) adopted a graph-to-graph annotation conversion method, utilizing pseudo data and inherit parameters to guide the graph transformation, achieving an automatic conversion from one annotation standard to another.

3 Our Proposed Approach

Treebank conversion aims to convert the source-side tree into the target-side tree. Given the word sequence $S = w_1 w_2 \dots w_n$ of the sentence, the part-of-speech sequence $T = t_1 t_2 \dots t_n$ and the source-side tree d^{src}, the task of treebank conversion is to output a syntactic tree d^{tgt} that conforms to the target-side norms. Therefore, effectively utilizing the information from the source-side treebank to guide the treebank conversion is key to the research. Naturally, we consider encoding the source-side treebank and using it as an additional input to the MLP to enhance the model's conversion capability. Specifically, we employ two strategies, Pattern Embedding(PE) and Tree BiLSTM, to encode the source-side tree from both local and global perspectives. Next, we will introduce the main components of the model, including the input layer, encoder layer, and decoder layer.

3.1 Embedding

Input Layer. We adopt the BiAffine Parser proposed by Dozat and Manning (2017) as our foundational model. As a graph-based dependency parsing model, this parser has achieved outstanding performance across multiple languages. The input layer primarily transforms words in a sentence into dense vectors. Compared to the original parser, we discovered that replacing label representations with character rep-

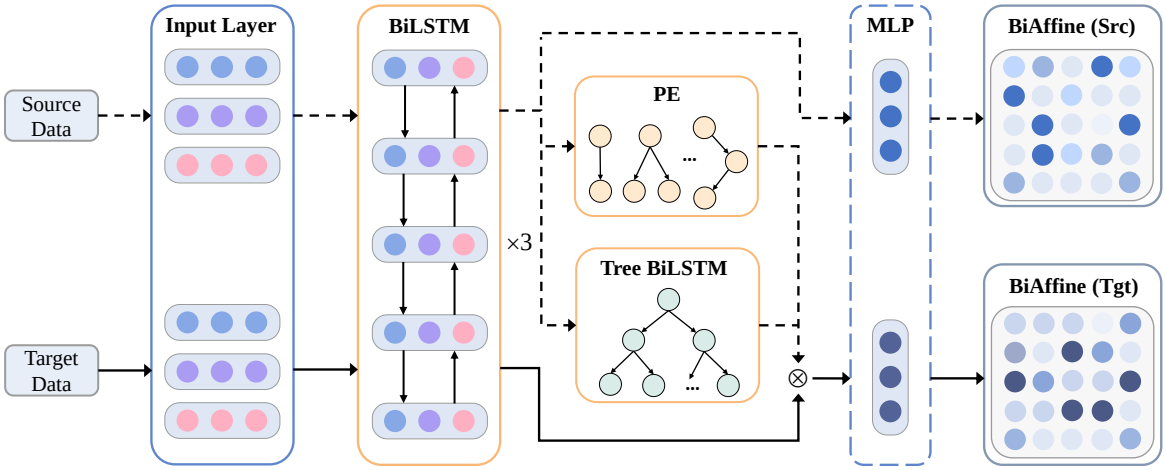

Figure 2: Framework of our proposed model.

resentations could further enhance the model performance. Moreover, to strengthen the representation ability of the model for Vietnamese, we introduced the pre-trained language model XLM-RoBERTa as an additional representation and fine-tuned it using a large volume of unlabeled data to improve its understanding and representation of Vietnamese. Consequently, the final input vector $\mathbf{x}_i$ for the input layer is constructed by concatenating $\mathbf{rep}_i^{\texttt{char}}$, $\mathbf{rep}_i^{\texttt{XLM-R}}$, and $\mathbf{emb}_i^{\texttt{word}}$.

$$\mathbf{x}_i = \mathbf{rep}_i^{\texttt{char}} \oplus \mathbf{rep}_i^{\texttt{XLM-R}} \oplus \mathbf{emb}_i^{\texttt{word}} \tag{1}$$

3.2 Encoder

BiLSTM. The encoder layer uses a three-layer bidirectional LSTM (BiLSTM) network to obtain a context-sensitive representation of each word. Multi-layer BiLSTM is capable of capturing more complex contextual information and deeper linguistic features. BiLSTM provides a comprehensive representation of each word by combining information from both directions of the sentence.

$$\mathbf{h}_0\mathbf{h}_1 \dots \mathbf{h}_n = \mathbf{BiLSTM}(\mathbf{x}_0\mathbf{x}_1 \dots \mathbf{x}_n, \theta_{\text{BiLSTM}}) \tag{2}$$

Pattern Embedding. Li et al. (2012) first proposed a pattern-based treebank conversion method. This method generates pseudo-double tree alignment data by using the syntax model trained by the source-side treebank to parse the target-side treebank. Then, they extracted pattern-based features from the source trees of this pseudo-double-tree alignment data and used these features as additional guidance information to train a more accurate target-side syntax model. However, this definition of pattern features only focuses on syntactic structure information and ignores the dependency relationship of the source tree. Based on this, Jiang et al. (2018) further refined and expanded the pattern features and introduced more label information. Therefore, in order to make full use of the information of the source syntax tree, we adopted 9 detailed pattern classifications based on the work of Jiang et al. (2018), as shown in Table 1.

Taking the calculation of arc $i \leftarrow j$ as an example, first determine the corresponding pattern type $P_{i \leftarrow j}$ based on the positional relationship between word w_i and word w_j in the source tree and refer to Table 1. Then obtain the dependency label corresponding to word w_i and word w_j and put It is represented as l_i and l_j. In addition, the closest relationship between word w_i and word w_j is also considered the dependency label l_a of the common ancestor word a. Finally, these four embedding representations are merged to form the pattern embedding representation $\mathbf{r}_{i \leftarrow j}^{pat}$.

$$\mathbf{r}_{i \leftarrow j}^{pat} = \mathbf{e}^{P_{i \leftarrow j}} \oplus \mathbf{e}^{l_i} \oplus \mathbf{e}^{l_j} \oplus \mathbf{e}^{l_a} \tag{3}$$

Target-side arc	Source-side arc	Pattern
$i \leftarrow j$	$i \leftarrow j$	consistent
	$i \leftarrow k \leftarrow j$	grand
	$i \leftarrow k \rightarrow j$	sibling
	$i \rightarrow j$	reverse
	$i \rightarrow k \rightarrow j$	reverse grand
	$i \leftarrow k \rightarrow m \rightarrow j, i \leftarrow k \leftarrow m \leftarrow j, \ldots$	Distance of 3
	$\ldots$	Distance of 4-5, 6 and ≥ 7

Table 1: Path-Pattern Correspondence Table.

Finally, the vector $\mathbf{r}_{i\leftarrow j}^{pat}$ is concatenated with h_i and h_j respectively, serving as the inputs for MLP^{D} and MLP^{H}.

Tree BiLSTM. To better leverage the information from the source-side tree, we naturally thought of using Tree BiLSTM to encode the source-side tree. Tree BiLSTM is an extension of BiLSTM that is more adept at handling tree-structured data. Tree BiLSTM aggregates and transfers information according to the tree structure of the input data, capturing hierarchical information by recursively processing each node in the tree. The state of each node not only depends on its child nodes but may also rely on the states of its parent or sibling nodes, allowing the model to capture long-distance dependencies and complex relationships within the tree structure.

To ensure that the Tree BiLSTM can fully capture the contextual information of the sentence, we directly utilize the output of the top-level BiLSTM as its input. Moreover, to introduce the dependency relations of the source-side tree, we also map the dependency labels of the source-side tree to embedding vectors as an additional input for the Tree BiLSTM. Taking the word w_k as an example, $\mathbf{r_k}$ is its top-level BiLSTM output vector, l_k is the dependency relation label of word w_k in the source tree, and e^{l_k} is the embedding vector mapped from the label. Therefore, the input for the word w_k in Tree BiLSTM is:

$$x_k = \mathbf{r_k} \oplus \mathbf{e}^{l_k} \tag{4}$$

In Tree BiLSTM, the hidden state vector of each node is calculated by combining the input vector of the node with the hidden state vectors of all its child nodes. Let x_a be the input vector for node a, and $C_{(a)}$ represent the set of all its child nodes. The hidden state vector h_k and cell state vector c_k correspond to the hidden and cell states of its children, respectively. Thus, the Tree LSTM computation process for node a can be described as follows:

$$\begin{aligned}
\hat{\mathbf{h}}_a &= \sum_{k \in C(a)} \mathbf{h}_k \\
\mathbf{i}_a &= \sigma(\mathbf{W}^{(i)}\mathbf{x}_a + \mathbf{U}^{(i)}\hat{\mathbf{h}}_a + \mathbf{b}^{(i)}) \\
\mathbf{f}_{ak} &= \sigma(\mathbf{W}^{(f)}\mathbf{x}_a + \mathbf{U}^{(f)}\mathbf{h}_k + \mathbf{b}^{(f)}) \\
\mathbf{o}_a &= \sigma(\mathbf{W}^{(o)}\mathbf{x}_a + \mathbf{U}^{(o)}\hat{\mathbf{h}}_a + \mathbf{b}^{(o)}) \\
\mathbf{u}_a &= \tanh(\mathbf{W}^{(u)}\mathbf{x}_a + \mathbf{U}^{(u)}\hat{\mathbf{h}}_a + \mathbf{b}^{(u)}) \\
\mathbf{c}_a &= \mathbf{i}_a \odot \mathbf{u}_a + \sum_{\mathbf{k} \in C(a)} \mathbf{f}_{ak} \odot \mathbf{c}_k \\
\mathbf{h}_a &= \mathbf{o}_a \odot \tanh(\mathbf{c}_a)
\end{aligned} \tag{5}$$

Where W and U are the weight matrices to be learned, b is the bias term, and σ is the sigmoid activation function.

After encoding through the Tree BiLSTM, we obtain the top-down word sequence representation $h_k^{\downarrow}$ and the bottom-up word sequence representation $h_k^{\uparrow}$ of the source-side tree. We then concatenate these

representations, and the output of the Tree BiLSTM is $\mathbf{r}_k^{tree} = h_k^{\downarrow} \oplus h_k^{\uparrow}$. Finally, $\mathbf{r}_k^{tree}$ is used as an additional input for the MLP layer.

3.3 Decoder

MLP layer. The MLP layer contains two independent multilayer perceptrons (MLP), one for predicting head words in dependencies and the other for predicting dependant words. This layer further merges the output of BiLSTM with the output of PE and Tree BiLSTM and converts it into features for predicting dependencies.

$$\begin{aligned} \mathbf{r}_i^{\mathrm{D}} &= \mathrm{MLP}^{\mathrm{D}}\left(\mathbf{h}_i \oplus \mathbf{r}_{i\leftarrow j}^{pat} \oplus \mathbf{r}_i^{tree}\right) \\ \mathbf{r}_j^{\mathrm{H}} &= \mathrm{MLP}^{\mathrm{H}}\left(\mathbf{h}_j \oplus \mathbf{r}_{i\leftarrow j}^{pat} \oplus \mathbf{r}_j^{tree}\right) \end{aligned} \tag{6}$$

BiAffine layer. The BiAffine layer is responsible for calculating the dependency scores between words, where the score represents the dependency arc score from the head word W_j to the dependent word w_i.

$$\mathtt{score}(i \leftarrow j) = \begin{bmatrix} \mathbf{r}_i^{\mathrm{D}} \\ 1 \end{bmatrix}^{\mathrm{T}} \mathbf{U}^1 \mathbf{r}_j^{\mathrm{H}} \tag{7}$$

where the dependency arc from W_j to w_i is determined by the weight matrix $\mathbf{U}^1$.

Parsing loss. Assuming that the correct core word of word W_i is W_j and the dependency relation label is l, the loss function of the model is defined as follows:

$$\begin{aligned} \mathcal{L}^{\mathrm{par}} = &- \log \frac{e^{\mathtt{score}(i\leftarrow j)}}{\sum\limits_{0\le k\le n, k\ne i} e^{\mathtt{score}(i\leftarrow k)}} \\ &- \log \frac{e^{score(i \xleftarrow{l} j)}}{\sum_{l'\in\mathcal{L}} e^{score(i \xleftarrow{l'} j)}} \end{aligned} \tag{8}$$

3.4 Multi-task Learning

In order to maximize the use of corpus information on related tasks, we adopt a multi-task learning strategy to achieve the contribution of all corpora to each task by training multiple tasks at the same time. We treat dependency parsing of source-side trees and target-side trees as two independent tasks. These two tasks jointly learn syntactic knowledge by sharing the parameters of the Input Layer and Encoder Layer, while using independent parameter configurations on the MLP layer and BiAffine layer to capture the unique characteristics of their respective specifications. This design not only emphasizes the commonalities between tasks but also retains their unique grammatical features, thereby effectively enhancing the performance of the model on dependency parsing tasks.

4 Experiments

4.1 Experiment Settings

Data. To ensure a fair comparison of the effects of treebank conversion, we selected several published Vietnamese dependency treebanks for our experimental analysis, including UD, VnDT, and VLSP2020. Where UD and VLSP2020 follow the same annotation standards. This chapter primarily discusses the processing and analysis of multiple datasets, including the source treebank VnDT, the target treebank UD, and the bi-tree aligned data UD^{VnDT}. For the bi-tree aligned data, we randomly selected 2900 sentences from VnDT and parsed them using a well-trained target-end dependency analysis model, generating pseudo-aligned tree data with noise. To minimize the impact of annotation errors on model performance, we have manually corrected these data. From the bi-tree aligned data, we randomly selected 300 sentences as the dev set and 600 sentences as the test set. The remaining approximately 2000 sentences were used as the training set. The scale of each dataset is detailed in Table 2.

	UD	VLSP2020	VnDT	UD^{VnDT}
train	1,400	8,152	8,977	1,996
dev	800	-	200	300
test	800	1,123	1020	600

Table 2: Vietnamese dependency treebanks

Evaluation. We employed the standard Unlabeled Attachment Score (UAS), which considers only the correctness of dependency arcs, and the Labeled Attachment Score (LAS), which evaluates the accuracy of both dependency arcs and dependency relations, to assess the performance of dependency parsing models.

Hyper-parameters. In our model, we use 100-dimensional XLM-RoBERTa pre-trained word vectors and 100-dimensional char embedding vectors. We set the number of layers of the BiLSTM encoding layer to 3, the output dimension to 300, and the dimensions of the MLP layer to 500 and 100. The dimensionality of the pattern embedding vectors is 50, the dimensionality of the dependency relation labels is 50, and the output dimensionality of the Tree BiLSTM is 200.

In the training phase, our batch size is 40 sentences. After training all the training corpus, it is regarded as one iteration. After each iteration, the performance of the model on the development set is evaluated. When the performance of the model does not improve after 50 iterations, training is stop. For the multi-task learning model,we proceed by first training a batch of UD data followed by a batch of VnDT data, considering the completion of one training cycle on VnDT data as one iteration.

4.2 Treebank Conversion Results

Table 3 presents the results of the treebank conversion. Given that UD and VnDT adopt completely different annotation standards, we compared the consistency of dependency arcs and dependency labels within UD^{VnDT}, where the consistency of dependency arcs was 52.4%, and the consistency of dependency labels was only 21.3%. Among the two methods we compared, the PE method employs local encoding to encode the source-side trees, while Tree BiLSTM uses global encoding. The experimental results indicate that the encoding approach of Tree BiLSTM achieved the best performance. It shows that the global encoding of the source-side trees can achieve better conversion performance when the data consistency is low. On the contrary, the PE method relies more on data consistency and has certain limitations.

	UD^{VnDT}			
	LAS	UAS	LAS	UAS
	with Multi-task		w/o Multi-task	
Tree BiLSTM	**65.88**	**81.06**	64.80	80.09
PE	62.73	77.85	62.02	76.86
Tree BiLSTM+PE	63.89	78.53	63.33	79.69

Table 3: Conversion accuracy on UD^{VnDT}.

At the same time, it naturally occurred to us to combine the Tree BiLSTM method with the PE methods, with the last row of the table presenting the experimental results. Contrary to our intuition, the experiment combining Tree BiLSTM and PE resulted in a decrease in performance, which we believe is due to information redundancy. As shown in Figure 3, we selected sentences from the test set for analysis. When encoding was done solely with Tree BiLSTM, the parser could produce results perfectly accurately. However, when employing both Tree BiLSTM and PE for encoding, the discrepancy between the UD and VnDT annotation rules was further amplified, leading to erroneous outputs from the model. Although the two methods attempt to encode the source-side syntactic tree from two different perspectives, local and global, the encoding through Tree BiLSTM already contains sufficient syntactic information, resulting in

redundancy rather than complementarity of the final syntactic information.

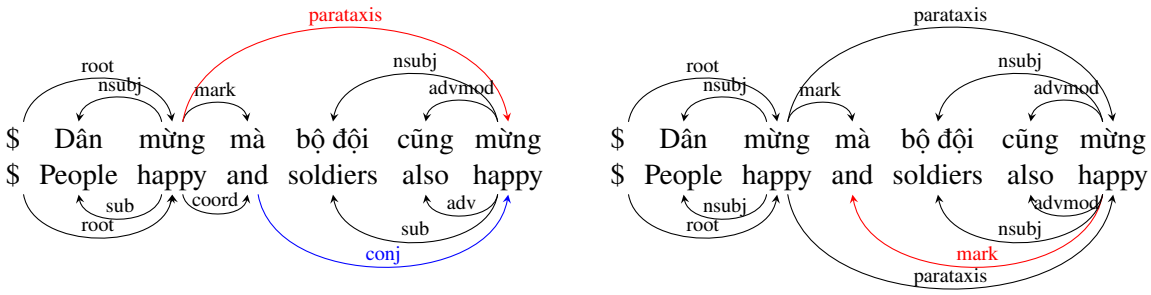

Figure 3: On the left, the upper half shows the target-side UD tree, and the under half shows the source-side VnDT tree. On the right, the upper half shows the output after Tree BiLSTM encoding, and the under half shows the output after combining Tree BiLSTM and PE.

4.3 Analysis

Ablation study. As the syntactic information fused in the model comprises a combination of multiple syntactic features, we conducted ablation experiments on these features. The results in table 4 illustrate the effects of removing fused source-side syntactic features from two different methods. For the PE method, the removal of any feature resulted in a slight decrease in model performance, indicating that incorporating source-side syntactic information positively impacts the model. The most significant impact on the UAS metric was the removal of l_i, which represents the dependency relation between word w_i and its head word, with a decrease of 1.52%. This suggests that source-side dependency relations indeed assist in treebank conversion. For the Tree BiLSTM methods, introducing dependency relation labels from the source-side trees slightly improved the model's conversion effectiveness. After integrating the source-side trees, LAS improved by 1.69% and UAS improved by 1.49%.

	UD^{VnDT}			
	LAS	UAS	LAS	UAS
	with Multi-task		w/o Multi-task	
PE	**62.73**	**77.85**	62.02	76.86
w/o l_i	61.38	76.33	61.03	76.11
w/o l_j	61.47	76.45	61.14	76.20
w/o l_a	60.89	76.82	60.52	76.24
TreeLSTM	**65.88**	**81.06**	64.80	80.90
w/o labels	64.81	80.16	64.19	79.57

Table 4: Feature ablation.

Error analysis. Inspired by the ablation experiments, we delved deeper into the effects of label types and both absolute and relative distances on conversion performance of the model. As shown in Figure 4, we analyzed the top ten most frequent label types in the dataset and their respective accuracy rates. The results indicated that the higher the frequency of occurrence for a label, the higher the accuracy of conversion, aligning with our understanding. However, some labels deviated from this pattern. We believe this discrepancy results from significant differences in the annotation standards of the dataset. For instance, the label "nsubj" under the UD dataset's annotation standards can be further subdivided into "nsubj", "nsubj:nn", "nsubj:pass","nsubj:xsubj" etc., which could significantly impact conversion accuracy of the model. Furthermore, we also evaluated the consistency between the UD and VnDT datasets, revealing label consistency at only 52.4% and arc consistency at 21.3%. This further validates the significant impact of annotation standard differences on the model's conversion performance.

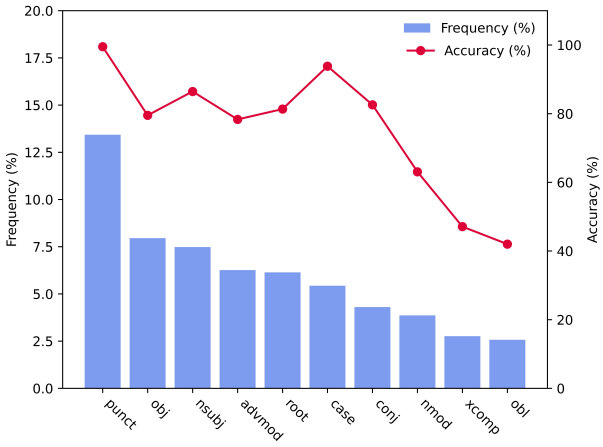

Figure 4: Statistics on the accuracy of the distances.

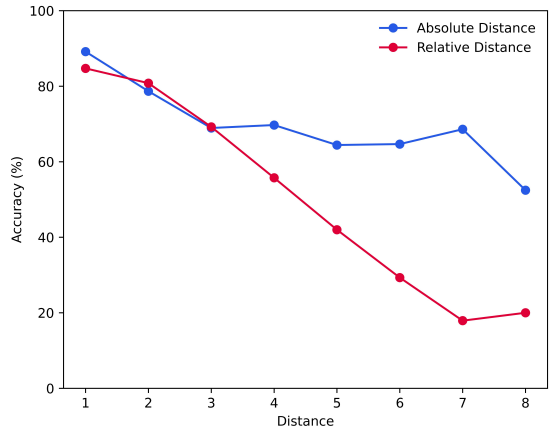

Figure 5: Statistics on the frequency and accuracy of the dependency labels.

In addition, we also analyzed the impact of both relative and absolute distances between words in a sentence on accuracy, as illustrated in Figure 4. Regardless of whether it was relative or absolute distance, accuracy tended to decrease as the distance increased. However, interestingly, when the relative distance was 8, there was a slight increase in accuracy. Upon analysis, we believe this anomaly is due to the fewer instances of dependency arcs with a relative distance of 8 in the dataset. Hence, we counted the number of dependency arcs with relative distances ranging from 1 to 8 in the dual-tree aligned data UD^{VnDT}. We found that there were 232 dependency arcs at a distance of 6, and only 10 at a distance of 8. Therefore, the reduction in the number of instances to some extent could cause a slight increase in accuracy.

A similar situation exists for absolute distance. As shown in Figure 5, when the absolute distance is 7, the accuracy rate is improved by 3.83% compared with when the absolute distance is 6. We found that at an absolute distance of 6, a total of 24 different labels appeared; While at an absolute distance of 7, this number was reduced to 18. Further analysis found that “punct" is the most frequently occurring tag among all tag types. Specifically when the absolute distance is 6, the proportion of “punct" label is 31%, and when the absolute distance is 7, this proportion rises to 40%. From the analysis in Figure 4, we can see that the “punct" is the label with the highest frequency of occurrence and the highest accuracy. The high proportion of “punct" labels is directly related to the improvement of its accuracy. At the same time, the reduction of label types also helps to improve accuracy to a certain extent.

4.4 Utilization of Conversion Data

Table 5 presents the final experimental results utilizing the converted data. Choudhary and O’riordan (2023) employ BERT fine-tuned in 12 languages as an encoder on an End-to-end Seq2seq Dependency Parser. Le-Hong and Cambria (2024) integrate dependency graph embeddings into the model using GRU. Compared with these works, our model achieves excellent results even without additional heterogeneous data, further verifying the effectiveness of our model.

Model	Training Data	LAS	UAS
Choudhary and O’riordan (2023)	UD	-	68.37
Le-Hong and Cambria (2024)	UD	47.24	52.06
Our Model	UD	**65.10**	**78.38**
Multi-task	UD&VnDT	65.25	78.65
Our Model	UD&Converted VnDT	72.22	82.23

Table 5: Performance of different dependency parsing models on the test set.

We test the base model, a multi-task learning model, and a conversion model employing the Tree BiLSTM method on the UD dataset. The results indicate that compared to the base model, the multi-task learning model did not demonstrate significant performance improvements. This may be due to the low consistency between the source treebank and the target treebank, and the two share fewer features, resulting in insufficient improvement in model performance. Furthermore, we use the trained conversion model to convert the VnDT training set, and then merge the converted VnDT treebank and UD treebank as a large training set to train the model. It can be seen that as the data size increases, the performance of the model has been greatly improved. LAS and UAS have improved by 7.12% and 3.85% respectively, which proves that the treebank conversion method can effectively utilize the syntactic information of the source treebank. In summary, the treebank conversion method that introduces source-side syntactic information can effectively improve the performance of the target-side syntactic analysis model; and when the consistency of the data is low, compared with the multi-task learning method, the treebank conversion method can be more direct and efficient Utilize the source treebank.

5 Conclusion

In this work, we presented a multi-feature enhanced multi-task learning framework for Vietnamese treebank conversion, addressing the challenges of limited corpora and the scarcity of large-scale annotated data in Vietnamese dependency parsing. Our approach leverages Tree BiLSTM and pattern embedding to capture both global and local syntactic features from the source treebank, and integrates these features into a multi-task learning framework, enhancing the representation capability for Vietnamese. The experiments conducted on benchmark datasets demonstrate the effectiveness of our model, which not only facilitates the conversion of heterogeneous Vietnamese Treebanks into a unified one but also significantly improves the Vietnamese dependency parsing accuracy.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (U21B2027, 62366027, 62266028, 62306129), Yunnan Provincial Major Science and Technology Special Plan Projects (202103AA080015, 202202AD080003, 202203AA080004), Yunnan Fundamental Research Projects (202401BC070021, 202301AS070047, 202401CF070121), Kunming University of Science and Technology's "Double First-rate" Construction Joint Project (202201BE070001-021, 202301BE070001-027), Yunnan High and New Technology Industry Project (201606). The authors would like to thank the anonymous reviewers for their valuable comments and suggestions for improving this paper.

References

Sharefah Al-Ghamdi, Hend Al-Khalifa, and Abdulmalik Al-Salman. 2023. Fine-tuning bert-based pre-trained models for arabic dependency parsing. *Applied Sciences*, 13(7).

Afra Amini, Tianyu Liu, and Ryan Cotterell. 2023. Hexatagging: Projective dependency parsing as tagging. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1453–1464, Toronto, Canada, July. Association for Computational Linguistics.

órunn Arnardóttir, Hinrik Hafsteinsson, Einar Freyr Sigursson, Kristín Bjarnadóttir, Anton Karl Ingason, Hildur Jónsdóttir, and Steinór Steingrímsson. 2020. A Universal Dependencies conversion pipeline for a Penn-format constituency treebank. In *Proceedings of the Fourth Workshop on Universal Dependencies (UDW 2020)*, pages 16–25, Barcelona, Spain (Online), December. Association for Computational Linguistics.

Tania Avgustinova and Yi Zhang. 2010. Conversion of a russian dependency treebank into hpsg derivations. In *Ninth International Workshop on Treebanks and Linguistic Theories*, page 7.

Emanuel Borges Völker, Maximilian Wendt, Felix Hennig, and Arne Köhn. 2019. HDT-UD: A very large Universal Dependencies treebank for German. In *Proceedings of the Third Workshop on Universal Dependencies (UDW, SyntaxFest 2019)*, pages 46–57, Paris, France, August. Association for Computational Linguistics.

Marie Candito, Benoît Crabbé, and Pascal Denis. 2010. Statistical french dependency parsing: treebank conversion and first results. In *Seventh International Conference on Language Resources and Evaluation-LREC 2010*, pages 1840–1847. European Language Resources Association (ELRA).

Chinmay Choudhary and Colm O'riordan. 2023. Multilingual end-to-end dependency parsing with linguistic typology knowledge. In *Proceedings of the 5th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 12–21.

Duc Do, Dien Dinh, An-Vinh Luong, and Thao Do. 2022. Adapting cross-lingual model to improve vietnamese dependency parsing. In *Artificial Intelligence in Data and Big Data Processing*, pages 97–108, Cham. Springer International Publishing.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Anette Frank. 2001. Treebank conversion. converting the negra treebank to an ltag grammar. In *Proceedings of the Workshop on Multi-layer Corpus-based Analysis*, pages 29–43.

Xinzhou Jiang, Zhenghua Li, Bo Zhang, Min Zhang, Sheng Li, and Luo Si. 2018. Supervised treebank conversion: Data and approaches. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2706–2716.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally.

Phuong Le-Hong and Erik Cambria. 2024. Integrating graph embedding and neural models for improving transition-based dependency parsing. *Neural Computing and Applications*, 36(6):2999–3016.

Zhenghua Li, Ting Liu, and Wanxiang Che. 2012. Exploiting multiple treebanks for parsing with quasi-synchronous grammars. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–684.

Xiang Li, Wenbin Jiang, Yajuan Lü, and Qun Liu. 2013. Iterative transformation of annotation guidelines for constituency parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–596.

Zuchao Li, Jiaxun Cai, Shexia He, and Hai Zhao. 2018. Seq2seq dependency parsing. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3203–3214, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

Zuchao Li, Hai Zhao, and Kevin Parnow. 2020. Global greedy dependency parsing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8319–8326, Apr.

Zuchao Li, Kevin Parnow, and Hai Zhao. 2022. Incorporating rich syntax information in grammatical error correction. *Information Processing Management*, 59(3):102891.

Dat Quoc Nguyen and Anh Tuan Nguyen. 2020. PhoBERT: Pre-trained language models for Vietnamese. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1037–1042, Online, November. Association for Computational Linguistics.

Dat Quoc Nguyen, Dai Quoc Nguyen, Son Bao Pham, Phuong-Thai Nguyen, and Minh Le Nguyen. 2014a. From treebank conversion to automatic dependency parsing for vietnamese. In *Natural Language Processing and Information Systems*, pages 196–207, Cham. Springer International Publishing.

Dat Quoc Nguyen, Dai Quoc Nguyen, Son Bao Pham, Phuong-Thai Nguyen, and Minh Le Nguyen. 2014b. From treebank conversion to automatic dependency parsing for vietnamese. In *Natural Language Processing and Information Systems: 19th International Conference on Applications of Natural Language to Information Systems, NLDB 2014, Montpellier, France, June 18-20, 2014. Proceedings 19*, pages 196–207. Springer.

Dongqi Pu and Khalil Sima'an. 2022. Passing parser uncertainty to the transformer: Labeled dependency distributions for neural machine translation. In *Proceedings of the 23rd Annual Conference of the European Association for Machine Translation*, pages 41–50, Ghent, Belgium, June. European Association for Machine Translation.

Sara Stymne, Miryam de Lhoneux, Aaron Smith, and Joakim Nivre. 2018. Parser training with heterogeneous treebanks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 619–625, Melbourne, Australia, July. Association for Computational Linguistics.

Nguyen Duc Thien, Nguyen Thi Thu Trang, and Truong Dang Quang. 2020. Applying graph neural networks for Vietnamese dependency parsing. In *Proceedings of the 7th International Workshop on Vietnamese Language and Speech Processing*, pages 54–59, Hanoi, Vietnam, December. Association for Computational Lingustics.

Tu-Anh Tran and Yusuke Miyao. 2022. Development of a multilingual ccg treebank via universal dependencies conversion. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5220–5233.

Duc Huu Trinh, Trinh Le-Phuong Ngo, Long Hong Buu Nguyen, and Dien Dinh. 2022. Applying cross-view training for dependency parsing in vietnamese. *ICIC express letters. Part B, Applications: an international journal of research and surveys*, 13(3):249–259.

Chau M. Truong, Tai V. Pham, Minh N. Phan, Nhan D. T. Le, Thinh V. Nguyen, and Quy T. Nguyen. 2022. Converting a constituency treebank to dependency treebank for vietnamese. In *2022 RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 256–261.

Yuxuan Wang, Zhilin Lei, Yuqiu Ji, and Wanxiang Che. 2022. Simple and effective graph-to-graph annotation conversion. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5450–5460, Gyeongju, Republic of Korea, October. International Committee on Computational Linguistics.

Songlin Yang and Kewei Tu. 2022. Headed-span-based projective dependency parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2188–2200, Dublin, Ireland, May. Association for Computational Linguistics.

Yu Zhang, Qingrong Xia, Shilin Zhou, Yong Jiang, Guohong Fu, and Min Zhang. 2022. Semantic role labeling as dependency parsing: Exploring latent tree structures inside arguments. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4212–4227, Gyeongju, Republic of Korea, October. International Committee on Computational Linguistics.