Multilingual Syntax-aware Language Modeling through Dependency Tree Conversion

Anonymous ACL submission

Abstract

Incorporating stronger syntactic biases into neural language models (LMs) is a long-standing goal, but research in this area often focuses on modeling English text, where constituent treebanks are readily available. Extending constituent tree-based LMs to the multilingual setting, where dependency treebanks are more common, is possible via dependency-to-constituency conversion methods. However, this raises the question of which tree formats are best for learning the model, and for which languages. We investigate this question by training recurrent neural network grammars (RNNGs) using various conversion methods, and evaluating them empirically in a multilingual setting. We examine the effect on LM performance across nine conversion methods and five languages through seven types of syntactic tests. On average, the performance of our best model represents a 19 % increase in accuracy over the worst choice across all languages. Our best model shows the advantage over sequential/overparameterized LMs, suggesting the positive effect of syntax injection in a multilingual setting. Our experiments highlight the importance of choosing the right tree formalism, and provide insights into making an informed decision.

1 Introduction

The importance of language modeling in recent years has grown considerably, as methods based on large pre-trained neural language models (LMs) have become the state-of-the-art for many problems (Devlin et al., 2019; Radford et al., 2019). However, these neural LMs are based on general architectures and therefore do not explicitly model linguistic constraints, and have been shown to capture only a subset of the syntactic representations typically found in constituency treebanks (Warstadt et al., 2020). An alternative line of LM research aims to explicitly model the parse tree in order to make the LM syntax-aware. A representative example of this paradigm, reccurent neural network grammar (RNNG, Dyer et al., 2016), is reported to perform better than sequential LMs on tasks that require complex syntactic analysis (Kuncoro et al., 2019; Hu et al., 2020; Noji and Oseki, 2021).

The aim of this paper is to extend LMs that inject syntax to the multilingual setting. This attempt is important mainly in two ways. Firstly, English has been dominant in researches on syntax-aware LM. While multilingual LMs have received increasing attention in recent years, most of their approaches do not explicitly model syntax, such as multilingual BERT (mBERT, Devlin et al., 2019) or XLM-R (Conneau et al., 2020). Although these models have shown high performance on some cross-lingual tasks (Conneau et al., 2018), they perform poorly on a syntactic task (Mueller et al., 2020). Secondly, syntax-aware LMs have interesting features other than their high syntactic ability. One example is the validity of RNNG as a cognitive model under an English-based setting, as demonstrated in Hale et al. (2018). Since human cognitive functions are universal, while natural languages are diverse, it would be ideal to conduct this experiment based on multiple languages.

The main obstacle for multilingual syntax-aware modeling is that it is unclear how to inject syntactic information while training. A straightforward approach is to make use of a multilingual treebank, such as Universal Dependencies (UD, Nivre et al., 2016; Nivre et al., 2020), where trees are represented in a dependency tree (DTree) formalism. Matthews et al. (2019) evaluated parsing and language modeling performance on three typologically different languages, using a generative dependency model. Unfortunately, they revealed that dependency-based models are less suited to language modeling than comparable constituency-based models, highlighting the apparent difficulty of extending syntax-aware LMs to other languages using existing resources.

	Partial tree **Stack-LSTM**	**Action**
0		NT(S)
1	(S $[\mathbf{e}_{\mathrm{S}}]$	NT(NP)
2	(S (NP $[\mathbf{e}_{\mathrm{S}}\ \mathbf{e}_{\mathrm{NP}}]$	GEN(The)
3	(S (NP The $[\mathbf{e}_{\mathrm{S}}\ \mathbf{e}_{\mathrm{NP}}\ \mathbf{e}_{\mathrm{The}}]$	GEN(pilot)
4	(S (NP The pilot $[\mathbf{e}_{\mathrm{S}}\ \underline{\mathbf{e}_{\mathrm{NP}}\ \mathbf{e}_{\mathrm{The}}\ \mathbf{e}_{\mathrm{pilot}}}]$	REDUCE
5	(S $\underline{\text{(NP The pilot)}}$ $[\mathbf{e}_{\mathrm{S}}\ \mathbf{e}_{\mathrm{NP'}}]$	NT(VP)
6	(S (NP The pilot) (VP $[\mathbf{e}_{\mathrm{S}}\ \mathbf{e}_{\mathrm{NP'}}\ \mathbf{e}_{\mathrm{VP}}]$	...

Figure 1: The illustration of stack-RNNG behavior. Stack-LSTM represents the current partial tree, in which adjacent vectors are connected in the network. At REDUCE action, the corresponding vector is updated with composition function (as underlined).

This paper revisits the issue of the difficulty of constructing multilingual syntax-aware LMs, by exploring the performance of multilingual language modeling using constituency-based models. Since our domain is a multilingual setting, our focus turns to how dependency-to-constituency conversion techniques result in different trees, and how these trees affect the model's performance. We obtain constituency treebanks from UD-formatted dependency treebanks of five languages using nine tree conversion methods. These treebanks are in turn used to train an RNNG, which we evaluate on perplexity and CLAMS (Mueller et al., 2020).

Our contributions are: (1) We propose a methodology for training multilingual syntax-aware LMs through the dependency tree conversion. (2) We found an optimal structure that brings out the potential of RNNG across five languages. (3) We demonstrated the advantage of our multilingual RNNG over sequential/overparameterized LMs.

2 Background

2.1 Recurrent Neural Network Grammars

RNNGs are generative models that estimate joint probability of a sentence $\boldsymbol{x}$ and a constituency tree (CTree) $\boldsymbol{y}$. The probability $p(\boldsymbol{x}, \boldsymbol{y})$ is estimated with top-down constituency parsing actions $\boldsymbol{a} = (a_1, a_2, \cdots, a_n)$ that produce $\boldsymbol{y}$:

$$p(\boldsymbol{x}, \boldsymbol{y}) = \prod_{t=1}^{n} p(a_t | a_1, \cdots, a_{t-1})$$

Kuncoro et al. (2017) proposed a stack-only RNNG that computes the next action probability based on the current partial tree. Figure 1 illustrates the behavior of it. The model represents the current partial tree with a stack-LSTM, which consists of three types of embeddings: nonterminal, word, and closed-nonterminal. The next action is estimated with the last hidden state of a stack-LSTM. There are three types of actions as follows:

- NT(X): Push nonterminal embedding of X ($\mathbf{e}_X$) onto the stack.
- GEN(w): Push word embedding of w ($\mathbf{e}_w$) onto the stack.
- REDUCE: Pop elements from the stack until a nonterminal embedding shows up. With all the embeddings which are popped, compute closed-nonterminal embedding $\mathbf{e}_{X'}$ using composition funcion COMP:

$$\mathbf{e}_{X'} = \textsc{Comp}(\mathbf{e}_X, \mathbf{e}_{w_1}, \cdots, \mathbf{e}_{w_m})$$

RNNG can be regarded as a language model that injects syntactic knowledge explicitly, and various appealing features have been reported (Kuncoro et al., 2017; Kuncoro et al., 2017; Hale et al., 2018). We focus on its high performance on *syntactic evaluation*, which is described below.

Difficulty in extending to other languages In principle, RNNG can be learned with any corpus as long as it contains CTree annotation. However, it is not evident which tree formats are best in a multilingual setting. Using the same technique as English can be inappropriate because each language has its own characteristic, which can be different from English. This question is the fundamental motivation of this research.

2.2 Cross-linguistic Syntactic Evaluation

To investigate the capability of LMs to capture syntax, previous work has attempted to create an evaluation set that requires analysis of the sentence structure (Linzen et al., 2016). One typical example is a subject-verb agreement, a rule that the form of a verb is determined by the grammatical category of the subject, such as person or number:

The pilot that the guards love laughs/*laugh. (1)

In (1), the form of *laugh* is determined by the subject *pilot*, not *guards*. This judgment requires

Algorithm 1: `lf` is short for left-first conversion. We omit right-first conversion because it can be defined just by swapping the codeblocks 6-9 and 10-13 of left-first conversion.

```
1  Function flat(w, ldeps, rdeps):
2      lNT ← [flat(lw, lw.ldeps, lw.rdeps) for lw in ldeps];
3      rNT ← [flat(rw, rw.ldeps, rw.rdeps) for rw in rdeps];
4      return [lNT [w] rNT].removeEmptyList;
5  Function lf(w, ldeps, rdeps):
6      if ldeps is not empty then
           /* Pop left-most dependent */
7          lw ← ldeps.pop();
8          lNT ← [lf(lw, lw.ldeps, lw.rdeps)];
9          rNT ← [lf(w, ldeps, rdeps)];
10     else if rdeps is not empty then
           /* Pop right-most dependent */
11         rw ← rdeps.pop();
12         lNT ← [lf(w, ldeps, rdeps)];
13         rNT ← [lf(rw, rw.ldeps, rw.rdeps)];
14     else return [w];
15     return [lNT rNT];
```

syntactic analysis; *guards* is not a subject of target verb *laugh* because it is in the relative clause of the real subject *pilot*.

Marvin and Linzen (2018) designed the English evaluation set using a grammatical framework. Mueller et al. (2020) extended this framework to other languages (French, German, Hebrew, and Russian) and created an evaluation set named CLAMS (Cross-Linguistic Assessment of Models on Syntax). CLAMS covers 7 categories of agreement tasks, including local agreement (e.g. The author laughs/*laugh) and non-local agreement that contains an intervening phrase between subject and verb as in (1). They evaluated LMs on CLAMS and demonstrated that sequential LMs often fail to assign a higher probability to the grammatical sentence in cases that involve non-local dependency.

Previous work has attempted to explore the syntactic capabilities of LMs with these evaluation sets. Kuncoro et al. (2019) compared the performance of LSTM LM and RNNG using the evaluation set proposed in Marvin and Linzen (2018), demonstrating the superiority of RNNG in predicting the agreement. Noji and Takamura (2020) suggested that LSTM LMs potentially have a limitation in handling object relative clauses. Since these analyses are performed on the basis of English text, it is unclear whether they hold or not in a multilingual setting. In this paper, we attempt to investigate this point by learning RNNGs in other languages and evaluating them on CLAMS.

3 Method: Dependency Tree Conversion

As a source of multilingual syntactic information, we use Universal Dependencies (UD), a collection of cross-linguistic dependency treebanks with a consistent annotation scheme. Since RNNG requires a CTree-formatted dataset for training, we perform DTree-to-CTree conversions, which are completely algorithmic to make it work regardless of language. Our method consists of two procedures: **structural conversion** and **nonterminal labeling**; obtaining a CTree skeleton with unlabeled nonterminal nodes, then assigning labels by leveraging syntactic information contained in the dependency annotations. While our structural conversion is identical to the baseline approach of Collins et al. (1999), we include a novel labeling method that relies on dependency relations, not POS tags.

Structural conversion We performed three types of structural conversion: *flat*, *left-first*, and *right-first*. Algorithm 1 shows the pseudo code and Figure 2 illustrates the actual conversions. These approaches construct CTree in a top-down manner following this procedure: 1) Introduce the root nonterminal of the head of a sentence ($\mathrm{NT}_{\mathrm{give}}$). 2) For each NT_w, introduce new nonterminals according to the dependent(s) of w. Repeat this procedure recursively until w has no dependents.

The difference between the three approaches is the ordering of introducing nonterminals. We describe their behaviors based on the example in Figure 2. (a) flat approach lets w and its dependents be children in CTree simultaneously. For example, $\mathrm{NT}_{\mathrm{give}}$ has four children: $\mathrm{NT}_{\mathrm{man}}$, $\mathrm{NT}_{\mathrm{give}}$, $\mathrm{NT}_{\mathrm{him}}$, $\mathrm{NT}_{\mathrm{box}}$, because they are dependents of the head word *give*. As the name suggests, this approach tends to produce a flat-structured CTree because each nonterminal can have multiple children. (b) left-first approach introduces the nonterminals from the left-most dependent. If there is no left dependent, the right-most dependent is introduced. In the example of Figure 2, the root $\mathrm{NT}_{\mathrm{give}}$ has a left child $\mathrm{NT}_{\mathrm{man}}$ because *man* is the left-most dependent of the head *give*. (c) right-first approach is the inversed version of left-first; handling the right-most dependent first. For methods (b) and (c), the resulting CTree is always a binary tree.

Nonterminal labeling We define three types of labeling methods for each NT_w; 1) X-label: Assign "X" to all the nonterminals. 2)POS-label: Assign POS tag of w. 3) DEP-label: Assign dependency

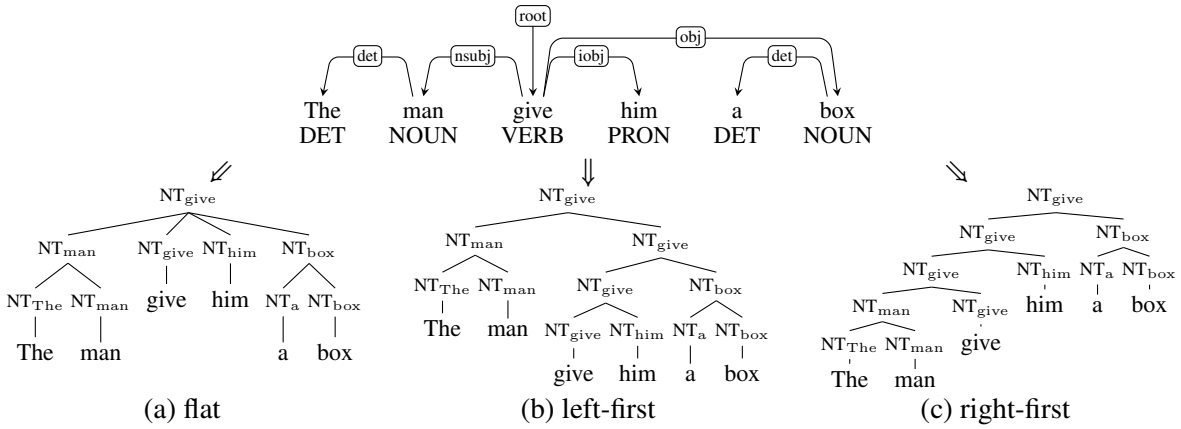

Figure 2: The illustration of structural conversion. NT_w is a temporal label of nonterminal which will be assigned at nonterminal labeling phase.

	X-label	POS-label	DEP-label
$\mathrm{NT}_{\mathrm{The}}$	X	DETP	det
$\mathrm{NT}_{\mathrm{man}}$	X	NOUNP	nsubj
$\mathrm{NT}_{\mathrm{give}}$	X	VERBP	root
$\mathrm{NT}_{\mathrm{him}}$	X	PRONP	iobj
NT_{a}	X	DETP	det
$\mathrm{NT}_{\mathrm{box}}$	X	NOUNP	obj

Table 1: Actual labels assigned to nonterminals.

relation between w and its head. Table 1 shows the actual labels that are assigned to CTrees in Figure 2.

Each method has its own intent. X-label drops the syntactic category of each phrase, which minimizes the structural information of the sentence. POS-label would produce the most common CTree structure because traditionally nonterminals are labeled based on POS tag of the head word. DEP-label is a more fine-grained method than POS-label because words in a sentence can have the same POS tag but different dependency relation, as in *man* and *box* in Figure 2.

Finally, we performed a total of nine types of conversions (three structures × three labelings). Although they have discrete features, they are common in that they embody reasonable phrase structures that are useful for capturing syntax. Figure 3 shows the converted structure of an actual instance from CLAMS. In all settings, the main subject phrase is correctly dominated by $\mathrm{NT}_{\mathrm{pilot}}$, which should contribute to solving the task.

4 What Is the Robust Conversion Which Works Well in Every Language?

In Section 3, we proposed language-independent multiple conversions from DTree to CTree. The intriguing question is; Is there a robust conversion that brings out the potential of RNNG in every language? To answer this question, we conducted a thorough experiment to compare the performances of RNNGs trained in each setting.

4.1 Experimental Setup

Treebank preparation Following Mueller et al. (2020), we extracted Wikipedia articles of target languages using WikiExtractor[1] to create corpora[2]. We fed it to UDify (Kondratyuk and Straka, 2019), a multilingual neural dependency parser trained on the entire UD treebanks, to generate a CoNLL-U formatted dependency treebank. Sentences are tokenized beforehand using Stanza (Qi et al., 2020) because UDify requires tokenized text for prediction. The resulting dependency treebank is converted into the constituency treebank using methods proposed in Section 3. Our treebank contains around 10% non-projective DTrees for all the language (between 9% in Russian and 14% in Hebrew), and we omit them in the conversion phase because we cannot obtain valid CTrees from them[3]. As a training set, we picked sentences with 10M tokens at random for each language. For a validation and a test set, we picked 5,000 sentences respectively.

Training details We used batched RNNG (Noji and Oseki, 2021) to speed up our training. Following Noji and Oseki (2021), we used subword units (Sennrich et al., 2016) with a vocabulary size of

[1] https://github.com/attardi/wikiextractor

[2] Although Mueller et al. (2020) publishes corpora they used, we extracted the dataset ourselves because they contain <unk> token which would affect parsing.

[3] Since other language can contain more non-projective DTrees, we have to consider how to handle it in the future.

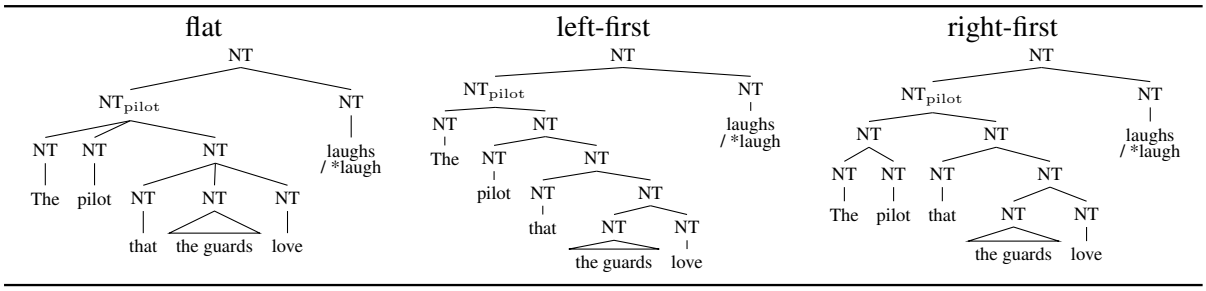

Figure 3: Examples of converted CTrees. A sentence is taken from CLAMS, which requires recognition of long distance dependency intervened by object relative clause (sentence (1)). For simplicity, we omit the corresponding word of each nonterminal except for *pilot*, the main subject of the sentence.

30K. We set the hyperparameters so as to make the model size 35M. We trained each model for 24 hours on a single GPU.

Evaluation metrics To compare the performance among conversions, we evaluated the model trained on each dataset in two aspects: **perplexity** and **syntactic ability** based on CLAMS.

Perplexity is a standard metric for assessing the quality of LM. Since we adopt subword units, we regard a word probability as a product of its subwords' probabilities. To compute it on RNNG, we performed word-synchronous beam search (Stern et al., 2017), a default approach implemented in batched RNNG. Following Noji and Oseki (2021), we set a beam size k as 100, a word beam size k_w as 10, and fast-track candidates k_s as 1. Syntactic ability is assessed by accuracy on CLAMS, which is calculated by comparing the probabilities assigned to a grammatical and an ungrammatical sentence. If the model assigns a higher probability to a grammatical sentence, then we regard it as correct. Chance accuracy is 0.5.

We run the experiment three times with different random seeds for initialization of the model, and report the average score with standard deviation.

4.2 Result

From now on, we refer to each conversion method according to a naming of the procedure, such as "left-first structure" or "flat-POS conversion".

Perplexity Table 2 shows the perplexities in each setting. As a whole, flat structures show the lowest perplexity, followed by left-first and right-first, which is consistent across languages. While flat structure produces stable and relatively low perplexity regardless of labeling methods and languages, left-first and right-first structures perform very poorly on X-label.

	flat	left	right	
X	$259_{\pm1}$	$707_{\pm19}$	$1507_{\pm14}$	English
POS	$278_{\pm3}$	$417_{\pm2}$	$512_{\pm3}$	
DEP	$\mathbf{241}_{\pm30}$	$390_{\pm4}$	$463_{\pm1}$	
X	$133_{\pm0}$	$405_{\pm10}$	$691_{\pm10}$	French
POS	$\mathbf{129}_{\pm1}$	$206_{\pm2}$	$262_{\pm1}$	
DEP	$137_{\pm22}$	$190_{\pm5}$	$223_{\pm2}$	
X	$341_{\pm1}$	$830_{\pm8}$	$1124_{\pm18}$	German
POS	$366_{\pm1}$	$321_{\pm3}$	$482_{\pm2}$	
DEP	$330_{\pm43}$	$\mathbf{291}_{\pm3}$	$398_{\pm4}$	
X	$100_{\pm1}$	$294_{\pm3}$	$450_{\pm8}$	Hebrew
POS	$97_{\pm0}$	$153_{\pm1}$	$183_{\pm1}$	
DEP	$\mathbf{93}_{\pm1}$	$143_{\pm1}$	$161_{\pm1}$	
X	$508_{\pm5}$	$1413_{\pm16}$	$1910_{\pm59}$	Russian
POS	$527_{\pm3}$	$845_{\pm2}$	$1067_{\pm16}$	
DEP	$\mathbf{473}_{\pm61}$	$834_{\pm5}$	$1030_{\pm27}$	

Table 2: Test set perplexity of each setting. Lower is better. "left" and "right" in the table are abbreviations of "left-first" and "right-first", respectively.

Syntactic ability Figure 4 shows the accuracies of CLAMS in each setting, and Table 3 shows the average scores. From Table 3, we observe clear distinctions across methods; the best model (shown in bold) is 19% more accurate in average than the worst one (shown in italic), across all languages, indicating the model's certain preference for the structure. Similar to perplexity, flat structure performs better and more stably than the others, regardless of labels and languages. While Mueller et al. (2020) reported a high variability in scores across languages when an LSTM LM is used, flat structure-based RNNGs do not show such a tendency; almost all the accuracies are above 90%.

Looking closely at the Figure 4, we can see that left-first and right-first structures exhibit unstable behavior depending on the labeling; the accuracy on X-label tends to be lower especially for the categories that require the resolution of a long-distance dependency, such as 'VP coord (long)', 'Across subj. rel.', 'Across obj. rel.', and 'Across prep'.

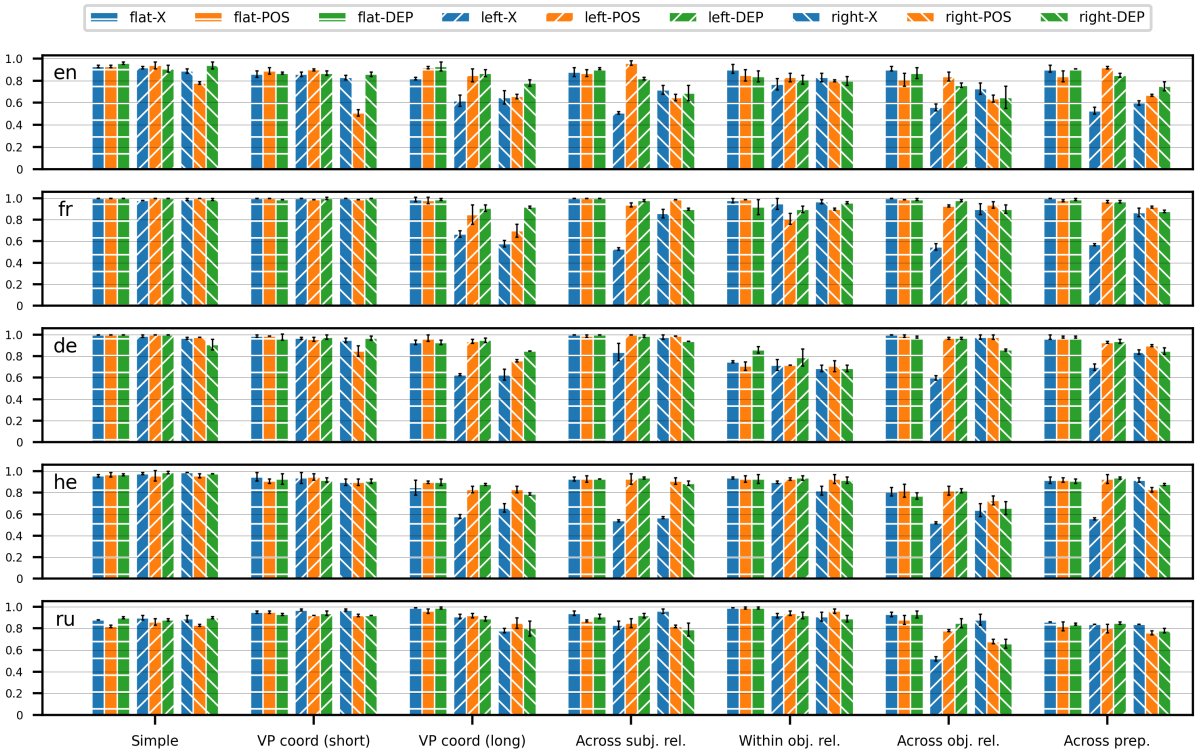

Figure 4: Accuracies of CLAMS for RNNGs trained on each setting.

	flat	left	right	
X	$0.89_{\pm.01}$	$0.68_{\pm.01}$	$0.75_{\pm.01}$	English
POS	$0.87_{\pm.02}$	$0.89_{\pm.01}$	$\mathit{0.67}_{\pm.01}$	
DEP	$\mathbf{0.90}_{\pm.02}$	$0.84_{\pm.01}$	$0.78_{\pm.04}$	
X	$\mathbf{0.99}_{\pm.00}$	$\mathit{0.75}_{\pm.00}$	$0.88_{\pm.02}$	French
POS	$\mathbf{0.99}_{\pm.00}$	$0.93_{\pm.02}$	$0.92_{\pm.01}$	
DEP	$0.98_{\pm.01}$	$0.96_{\pm.01}$	$0.94_{\pm.01}$	
X	$0.95_{\pm.00}$	$\mathit{0.78}_{\pm.01}$	$0.86_{\pm.01}$	German
POS	$0.95_{\pm.01}$	$0.93_{\pm.01}$	$0.88_{\pm.01}$	
DEP	$\mathbf{0.96}_{\pm.01}$	$0.95_{\pm.02}$	$0.87_{\pm.02}$	
X	$0.91_{\pm.01}$	$\mathit{0.72}_{\pm.01}$	$0.78_{\pm.01}$	Hebrew
POS	$0.91_{\pm.01}$	$0.91_{\pm.03}$	$0.87_{\pm.01}$	
DEP	$0.90_{\pm.01}$	$\mathbf{0.92}_{\pm.00}$	$0.86_{\pm.01}$	
X	$\mathbf{0.93}_{\pm.00}$	$0.84_{\pm.01}$	$0.89_{\pm.02}$	Russian
POS	$0.90_{\pm.01}$	$0.87_{\pm.01}$	$0.83_{\pm.01}$	
DEP	$\mathbf{0.93}_{\pm.01}$	$0.89_{\pm.00}$	$\mathit{0.82}_{\pm.01}$	

Table 3: CLAMS scores averaged by task category.

Discussion Basically, we observed a similar tendency in perplexity and CLAMS score; (1) flat structures show the highest scores. (2) left-first and right-first structures perform poorly on X-label. We conjecture that these tendencies are due to the resulting structure of each conversion; while flat structure is non-binary, the rest two are binary. Since nonterminals in a non-binary tree can have multiple words as children, parsing actions obtained from it contain more continuous GEN actions than a binary tree. This nature helps the model to predict the next word by considering lexical relations, which would contribute to its lower perplexity. Although binary trees get better with the hint of informative labels (POS/DEP), it is difficult to reach the performance of flat structures due to their confused actions; GEN actions tend to be interrupted by other actions. Besides, there are too many NT actions in a binary tree, which can hurt the prediction because the information of an important nonterminal (e.g. NT_{pilot} in Figure 3) can be diluted through the actions. The situation becomes worse on X-label; the model cannot distinguish the nonterminal of the main subject and that of the other, resulting in missing what the subject is.

It is worth noting that perplexity does not always reflect the CLAMS accuracy. For example, while right-X conversion produces the worst perplexity for all the languages, it achieves better CLAMS accuracy than left-X conversion for almost all the cases. This observation is in line with Hu et al. (2020), who report a dissociation between perplexity and syntactic performance for English.

4.3 Why Does Flat Structure Perform Well?

As one possible reason why flat structure is optimal among the three structures presented, we conjecture that the parseability of the structure is involved. To test this hypothesis, we calculated the F1 score

	flat	left	right	
X	$0.80_{\pm.00}$	$0.34_{\pm.00}$	$0.48_{\pm.00}$	English
POS	$0.79_{\pm.00}$	$0.57_{\pm.00}$	$0.70_{\pm.00}$	
DEP	$\mathbf{0.82}_{\pm.01}$	$0.59_{\pm.01}$	$0.70_{\pm.00}$	
X	$0.79_{\pm.00}$	$0.37_{\pm.00}$	$0.58_{\pm.00}$	French
POS	$\mathbf{0.86}_{\pm.00}$	$0.63_{\pm.00}$	$0.74_{\pm.00}$	
DEP	$\mathbf{0.86}_{\pm.01}$	$0.65_{\pm.01}$	$0.75_{\pm.00}$	
X	$0.90_{\pm.00}$	$0.44_{\pm.00}$	$0.59_{\pm.00}$	German
POS	$0.85_{\pm.00}$	$0.74_{\pm.00}$	$0.76_{\pm.00}$	
DEP	$\mathbf{0.91}_{\pm.08}$	$0.76_{\pm.00}$	$0.77_{\pm.00}$	
X	$0.81_{\pm.01}$	$0.41_{\pm.00}$	$0.58_{\pm.00}$	Hebrew
POS	$\mathbf{0.83}_{\pm.00}$	$0.65_{\pm.00}$	$0.73_{\pm.00}$	
DEP	$\mathbf{0.83}_{\pm.00}$	$0.65_{\pm.00}$	$0.72_{\pm.00}$	
X	$0.80_{\pm.00}$	$0.41_{\pm.00}$	$0.59_{\pm.00}$	Russian
POS	$\mathbf{0.83}_{\pm.00}$	$0.62_{\pm.00}$	$0.73_{\pm.00}$	
DEP	$0.82_{\pm.01}$	$0.58_{\pm.00}$	$0.68_{\pm.00}$	

Table 4: F1 score of predicted CTree. We regard a resulting CTree of each conversion as a gold tree.

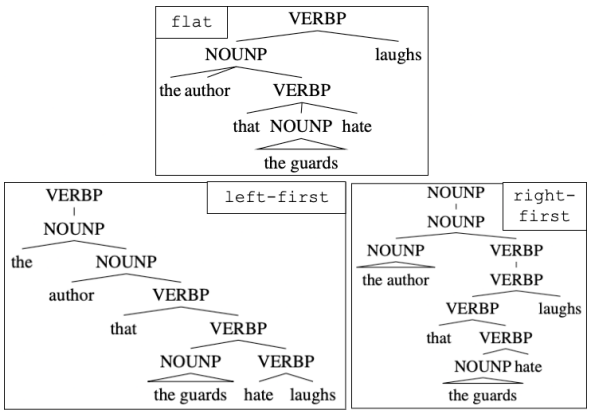

Figure 5: Structures of a CLAMS example predicted by {flat, left-first, right-first}-POS RNNG. This example is solvable only by flat-POS RNNG across all seeds.

between the gold CTrees of the test set and the structures predicted by RNNG for each setting. Table 4 shows the result. The tendencies of F1 scores are consistent across languages: 1) Flat structures show highest F1 score. 2) While scores of flat structures are stable regardless of their labelings, the rest two structures exhibit lower score on X-label. As a whole, the result reflects the tendency discussed in Section 4.2, which supports our hypothesis.

To further investigate the link between parseability and the capability of solving the task, we obtained parse trees of CLAMS examples that are solvable only by flat RNNG across all seeds. We found that only flat RNNG produces a correct constituency tree, and structures obtained from left-first and right-first RNNGs are incorrect on a critical point. For example, in Figure 5, while the relation between the subject "author" and the target verb "laughs" is analyzed clearly in the flat structure, it is ambiguous in the rest, possibly causing the misinterpretation that the subject is "guards".

These findings indicate the importance of choosing the correct tree structure for syntax-aware language modeling; it should be not only hierarchical, but also as parseable as possible.

Through analysis of the conversions, we found that (1) flat structure performs stably well in every setting. (2) while CLAMS accuracy of flat structure does not differ significantly depending on its labeling, for perplexity, flat-DEP performs the best for more than half of the languages and no inferiority can be observed for the other languages. Therefore, we conclude that *flat-DEP* conversion is the most robust conversion among languages.

5 Advantage of Syntax Injection to LMs in a Multilingual Setting

In this section, we demonstrate the benefits of injecting syntactic biases into the model in a multilingual setting. We obtained the CLAMS score of RNNG trained on the flat-DEP treebank (*flat-DEP RNNG* for short) and compared it against baselines.

Experimental setup The experiment was conducted in as close setting to the previous work as possible. Following Mueller et al. (2020), we extracted Wikipedia articles of 80M tokens as training set. The hyperparameters of LSTM LM are set following Noji and Takamura (2020) because it performs the best for the dataset of Marvin and Linzen (2018)[4]. We used subword units with a vocabulary size of 30K, and the sizes of RNNG and LSTM LM are set to be the same (35M).

Result Table 5 shows the result. In addition to scores from the models we trained (flat-DEP RNNG, LSTM (N20)), we display scores of LSTM LM and mBERT reported in the original paper (LSTM (M20) and mBERT (M20), Mueller et al., 2020). Overall, we can see the superiority of RNNG across languages, especially for the tasks that require analysis on long distance dependency; 'VP coord (long)', 'Across subj. rel.', 'Across obj. rel.', and 'Across prep'. While previous work suggested that LSTM LMs potentially have a limitation in handling object relative clauses (Noji and Takamura, 2020), our result suggests that RNNG does not have such a limitation thanks to explicitly injected syntactic biases.

[4]Since English set of CLAMS is a subset of Marvin and Linzen (2018), it is reasonable to choose this model to validate the multilingual extendability.

	Simple	VP coord (short)	VP coord (long)	Across subj. rel.	Within obj rel.	Across obj rel.	Across prep.	Average	
flat-DEP RNNG	$0.99_{\pm.01}$	$0.87_{\pm.02}$	$0.91_{\pm.04}$	$0.95_{\pm.02}$	$0.92_{\pm.05}$	$0.92_{\pm.06}$	$0.93_{\pm.04}$	$\mathbf{0.93}_{\pm.02}$	English
LSTM (N20)	$0.93_{\pm.03}$	$0.85_{\pm.01}$	$0.83_{\pm.04}$	$0.85_{\pm.04}$	$0.83_{\pm.05}$	$0.77_{\pm.04}$	$0.87_{\pm.02}$	$0.85_{\pm.02}$	
LSTM (M20)	$1.00_{\pm.00}$	$0.94_{\pm.01}$	$0.76_{\pm.06}$	$0.60_{\pm.06}$	$0.89_{\pm.01}$	$0.55_{\pm.05}$	$0.63_{\pm.02}$	$0.77_{\pm.03}$	
mBERT (M20)	1.00	1.00	0.92	0.88	0.83	0.87	0.92	0.92	
flat-DEP RNNG	$1.00_{\pm.00}$	$1.00_{\pm.00}$	$1.00_{\pm.00}$	$1.00_{\pm.00}$	$1.00_{\pm.00}$	$1.00_{\pm.00}$	$1.00_{\pm.00}$	$\mathbf{1.00}_{\pm.00}$	French
LSTM (N20)	$1.00_{\pm.00}$	$1.00_{\pm.00}$	$0.97_{\pm.03}$	$0.92_{\pm.06}$	$0.85_{\pm.03}$	$0.75_{\pm.01}$	$1.00_{\pm.00}$	$0.93_{\pm.01}$	
LSTM (M20)	$1.00_{\pm.00}$	$0.97_{\pm.01}$	$0.85_{\pm.05}$	$0.71_{\pm.05}$	$0.99_{\pm.01}$	$0.52_{\pm.01}$	$0.74_{\pm.02}$	$0.83_{\pm.02}$	
mBERT (M20)	1.00	1.00	0.98	0.57	—	0.86	0.57	0.83	
flat-DEP RNNG	$1.00_{\pm.00}$	$0.99_{\pm.01}$	$0.98_{\pm.01}$	$1.00_{\pm.00}$	$0.88_{\pm.04}$	$0.99_{\pm.01}$	$0.97_{\pm.02}$	$\mathbf{0.97}_{\pm.01}$	German
LSTM (N20)	$0.99_{\pm.01}$	$0.97_{\pm.03}$	$0.92_{\pm.05}$	$0.99_{\pm.01}$	$0.72_{\pm.01}$	$0.97_{\pm.02}$	$0.94_{\pm.01}$	$0.93_{\pm.01}$	
LSTM (M20)	$1.00_{\pm.00}$	$0.99_{\pm.02}$	$0.96_{\pm.04}$	$0.94_{\pm.04}$	$0.74_{\pm.03}$	$0.81_{\pm.09}$	$0.89_{\pm.06}$	$0.90_{\pm.04}$	
mBERT (M20)	0.95	0.97	1.00	0.73	—	0.93	0.95	0.92	
flat-DEP RNNG	$0.97_{\pm.01}$	$0.99_{\pm.00}$	$0.92_{\pm.03}$	$0.95_{\pm.02}$	$1.00_{\pm.00}$	$0.84_{\pm.05}$	$0.95_{\pm.01}$	$\mathbf{0.95}_{\pm.01}$	Hebrew
LSTM (N20)	$0.97_{\pm.00}$	$0.95_{\pm.04}$	$0.85_{\pm.02}$	$0.89_{\pm.02}$	$0.94_{\pm.01}$	$0.63_{\pm.04}$	$0.93_{\pm.01}$	$0.88_{\pm.00}$	
LSTM (M20)	$0.95_{\pm.01}$	$1.00_{\pm.01}$	$0.84_{\pm.06}$	$0.91_{\pm.03}$	$1.00_{\pm.01}$	$0.56_{\pm.01}$	$0.88_{\pm.03}$	$0.88_{\pm.02}$	
mBERT (M20)	0.70	0.91	0.73	0.61	—	0.55	0.62	0.69	
flat-DEP RNNG	$0.89_{\pm.02}$	$0.94_{\pm.02}$	$1.00_{\pm.00}$	$0.93_{\pm.00}$	$0.99_{\pm.01}$	$0.92_{\pm.02}$	$0.85_{\pm.03}$	$\mathbf{0.93}_{\pm.01}$	Russian
LSTM (N20)	$0.91_{\pm.01}$	$0.97_{\pm.00}$	$0.97_{\pm.02}$	$0.98_{\pm.00}$	$0.90_{\pm.04}$	$0.85_{\pm.07}$	$0.86_{\pm.02}$	$0.92_{\pm.01}$	
LSTM (M20)	$0.91_{\pm.01}$	$0.98_{\pm.02}$	$0.86_{\pm.04}$	$0.88_{\pm.03}$	$0.95_{\pm.04}$	$0.60_{\pm.03}$	$0.76_{\pm.02}$	$0.85_{\pm.03}$	
mBERT (M20)	0.65	0.80	—	0.70	—	0.67	0.56	0.68	

Table 5: CLAMS scores for flat-DEP RNNG and baselines. LSTM (N20) is a model of which hyperparameters are set as with Noji and Takamura (2020). LSTM (M20) and mBERT (M20) scores are quoted from Table 1, 2 and 5 in Mueller et al. (2020). Hyphen means that all focus verb for the corresponding setting were out-of-vocabulary.

6 Discussion

We discussed the CTree structure that works robustly regardless of the language and the superiority of injecting syntactic bias to the model. Our claim is that we can construct language-independent syntax-aware LMs by seeking the best structure for learning RNNGs, which is backed up by our experiments based on five languages. To make this claim firm, more investigations are needed from two aspects: **fine-grained syntactic evaluation** and **experiment on typologically diverse languages**.

Fine-grained syntactic evaluation The linguistic phenomenon covered in CLAMS is only an agreement. However, previous works have invented evaluation sets that examine more diverse syntactic phenomena for English (Hu et al., 2020, Warstadt et al., 2020). We need such a fine-grained evaluation even in a multilingual setting, as superiority in agreement does not imply superiority in every syntactic knowledge; Kuncoro et al. (2019) suggested that RNNG performs poorer than LSTM LM in capturing sentential complement or simple negative polarity items. It is challenging to design a multiliugnal syntactic test set because even an agreement based on grammatical categories is not a universal phenomenon. It is required to seek reasonable metrics that cover broad syntactic phenomena and are applicable to many languages.

Experiment on typologically diverse languages Languages included in CLAMS (English, French, German, Hebrew and Russian) are actually not typologically diverse. Apart from language-specific features, all of them take the same ordering of (1) subject, verb, and object (SVO) (2) relative clause and noun (Noun-Relative clause) (3) adposition and noun phrase (preposition), and so on[5]. If we run the same experiment for a typologically different language, the result could be somewhat different. Although some previous work focused on syntactic assessment of other languages (Ravfogel et al., 2018; Gulordava et al., 2018), such attempts are scarce. As future work, it is needed to design an evaluation set based on other languages and explore the extendability to more diverse languages.

7 Conclusion

In this paper, we propose a methodology to learn multilingual RNNG through dependency tree conversion. We performed multiple conversions to seek the robust structure which works well multilingually, discussing the effect of multiple structures. We demonstrated the superiority of our model over baselines in capturing syntax in a multilingual setting. Since our research is the first step for multilingual syntax-aware LMs, it is necessary to conduct experiments on more diverse languages to seek a better structure. We believe that this research would contribute to the field of theoretical/cognitive linguistics as well because an ultimate goal of linguistics is finding the universal rule of natural language. Finding a reasonable structure in engineering would yield useful knowledge for that purpose.

[5]Typological information is obtained from WALS: https://wals.info/

References

Michael Collins, Jan Hajic, Lance Ramshaw, and Christoph Tillmann. 1999. A statistical parser for Czech. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 505–512, College Park, Maryland, USA. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California. Association for Computational Linguistics.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

John Hale, Chris Dyer, Adhiguna Kuncoro, and Jonathan Brennan. 2018. Finding syntax in human encephalography with beam search. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2727–2736, Melbourne, Australia. Association for Computational Linguistics.

Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger Levy. 2020. A systematic assessment of syntactic generalization in neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1725–1744, Online. Association for Computational Linguistics.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A. Smith. 2017. What do recurrent neural network grammars learn about syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1249–1258, Valencia, Spain. Association for Computational Linguistics.

Adhiguna Kuncoro, Chris Dyer, Laura Rimell, Stephen Clark, and Phil Blunsom. 2019. Scalable syntax-aware language models using knowledge distillation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3472–3484, Florence, Italy. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Austin Matthews, Graham Neubig, and Chris Dyer. 2019. Comparing top-down and bottom-up neural generative dependency models. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 227–237, Hong Kong, China. Association for Computational Linguistics.

Aaron Mueller, Garrett Nicolai, Panayiota Petrou-Zeniou, Natalia Talmina, and Tal Linzen. 2020. Cross-linguistic syntactic evaluation of word prediction models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5523–5539, Online. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman.

2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Hiroshi Noji and Yohei Oseki. 2021. Effective batching for recurrent neural network grammars. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4340–4352, Online. Association for Computational Linguistics.

Hiroshi Noji and Hiroya Takamura. 2020. An analysis of the utility of explicit negative examples to improve the syntactic abilities of neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3375–3385, Online. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Shauli Ravfogel, Yoav Goldberg, and Francis Tyers. 2018. Can LSTM learn to capture agreement? The case of Basque. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 98–107, Brussels, Belgium. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Mitchell Stern, Daniel Fried, and Dan Klein. 2017. Effective inference for generative neural parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1695–1700, Copenhagen, Denmark. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: A benchmark of linguistic minimal pairs for English. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 409–410, New York, New York. Association for Computational Linguistics.