

# Deep Contextualized Word Embeddings in Transition-Based and Graph-Based Dependency Parsing – A Tale of Two Parsers Revisited\*

**Artur Kulmizev Miryam de Lhoneux Johannes Gontrum Elena Fano Joakim Nivre**

Department of Linguistics and Philology, Uppsala University

{artur.kulmizev,miryam.de\_lhoneux,joakim.nivre}@lingfil.uu.se

{johannes.gontrum.4608,elena.fano.3249}@student.uu.se

## Abstract

Transition-based and graph-based dependency parsers have previously been shown to have complementary strengths and weaknesses: transition-based parsers exploit rich structural features but suffer from error propagation, while graph-based parsers benefit from global optimization but have restricted feature scope. In this paper, we show that, even though some details of the picture have changed after the switch to neural networks and continuous representations, the basic trade-off between rich features and global optimization remains essentially the same. Moreover, we show that deep contextualized word embeddings, which allow parsers to pack information about global sentence structure into local feature representations, benefit transition-based parsers more than graph-based parsers, making the two approaches virtually equivalent in terms of both accuracy and error profile. We argue that the reason is that these representations help prevent search errors and thereby allow transition-based parsers to better exploit their inherent strength of making accurate local decisions. We support this explanation by an error analysis of parsing experiments on 13 languages.

## 1 Introduction

For more than a decade, research on data-driven dependency parsing has been dominated by two approaches: transition-based parsing and graph-based parsing (McDonald and Nivre, 2007, 2011). Transition-based parsing reduces the parsing task to scoring single parse actions and is often combined with local optimization and greedy search algorithms. Graph-based parsing decomposes parse trees into subgraphs and relies on global optimization and exhaustive (or at least non-greedy)

search to find the best tree. These radically different approaches often lead to comparable parsing accuracy, but with distinct error profiles indicative of their respective strengths and weaknesses, as shown by McDonald and Nivre (2007, 2011).

In recent years, dependency parsing, like most of NLP, has shifted from linear models and discrete features to neural networks and continuous representations. This has led to substantial accuracy improvements for both transition-based and graph-based parsers and raises the question whether their complementary strengths and weaknesses are still relevant. In this paper, we replicate the analysis of McDonald and Nivre (2007, 2011) for neural parsers. In addition, we investigate the impact of deep contextualized word representations (Peters et al., 2018; Devlin et al., 2019) for both types of parsers.

Based on what we know about the strengths and weaknesses of the two approaches, we hypothesize that deep contextualized word representations will benefit transition-based parsing more than graph-based parsing. The reason is that these representations make information about global sentence structure available locally, thereby helping to prevent search errors in greedy transition-based parsing. The hypothesis is corroborated in experiments on 13 languages, and the error analysis supports our suggested explanation. We also find that deep contextualized word representations improve parsing accuracy for longer sentences, both for transition-based and graph-based parsers.

## 2 Two Models of Dependency Parsing

After playing a marginal role in NLP for many years, dependency-based approaches to syntactic parsing have become mainstream during the last fifteen years. This is especially true if we consider languages other than English, ever since the influ-

\* We gratefully acknowledge the inspiration for our subtitle in the seminal paper by Zhang and Clark (2008).

ential CoNLL shared tasks on dependency parsing in 2006 (Buchholz and Marsi, 2006) and 2007 (Nivre et al., 2007) with data from 19 languages.

The transition-based approach to dependency parsing was pioneered by Yamada and Matsumoto (2003) and Nivre (2003), with inspiration from history-based parsing (Black et al., 1992) and data-driven shift-reduce parsing (Veenstra and Daelemans, 2000). The idea is to reduce the complex parsing task to the simpler task of predicting the next parsing action and to implement parsing as greedy search for the optimal sequence of actions, guided by a simple classifier trained on local parser configurations. This produces parsers that are very efficient, often with linear time complexity, and which can benefit from rich non-local features defined over parser configurations but which may suffer from compounding search errors.

The graph-based approach to dependency parsing was developed by McDonald et al. (2005a,b), building on earlier work by Eisner (1996). The idea is to score dependency trees by a linear combination of scores of local subgraphs, often single arcs, and to implement parsing as exact search for the highest scoring tree under a globally optimized model. These parsers do not suffer from search errors but parsing algorithms are more complex and restrict the scope of features to local subgraphs.

The terms *transition-based* and *graph-based* were coined by McDonald and Nivre (2007, 2011), who performed a contrastive error analysis of the two top-performing systems in the CoNLL 2006 shared task on multilingual dependency parsing: MaltParser (Nivre et al., 2006) and MSTParser (McDonald et al., 2006), which represented the state of the art in transition-based and graph-based parsing, respectively, at the time. Their analysis shows that, despite having almost exactly the same parsing accuracy when averaged over 13 languages, the two parsers have very distinctive error profiles. MaltParser is more accurate on short sentences, on short dependencies, on dependencies near the leaves of the tree, on nouns and pronouns, and on subject and object relations. MSTParser is more accurate on long sentences, on long dependencies, on dependencies near the root of the tree, on verbs, and on coordination relations and sentence roots.

McDonald and Nivre (2007, 2011) argue that these patterns can be explained by the complementary strengths and weaknesses of the systems. The

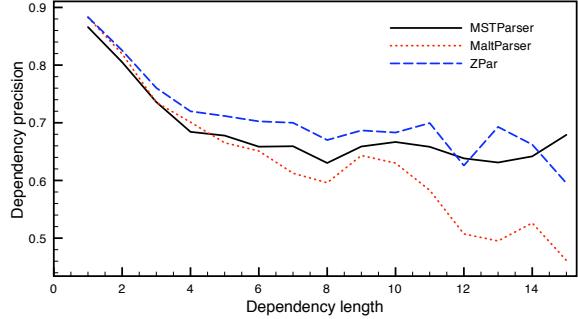


Figure 1: Labeled precision by dependency length for MST (global-exhaustive-graph), Malt (local-greedy-transition) and ZPar (global-beam-transition). From Zhang and Nivre (2012).

transition-based MaltParser prioritizes rich structural features, which enable accurate disambiguation in local contexts, but is limited by a locally optimized model and greedy algorithm, resulting in search errors for structures that require longer transition sequences. The graph-based MSTParser benefits from a globally optimized model and exact inference, which gives a better analysis of global sentence structure, but is more restricted in the features it can use, which limits its capacity to score local structures accurately.

Many of the developments in dependency parsing during the last decade can be understood in this light as attempts to mitigate the weaknesses of traditional transition-based and graph-based parsers without sacrificing their strengths. This may mean evolving the model structure through new transition systems (Nivre, 2008, 2009; Kuhlmann et al., 2011) or higher-order models for graph-based parsing (McDonald and Pereira, 2006; Carreras, 2007; Koo and Collins, 2010); it may mean exploring alternative learning strategies, in particular for transition-based parsing, where improvements have been achieved thanks to global structure learning (Zhang and Clark, 2008; Zhang and Nivre, 2011; Andor et al., 2016) and dynamic oracles (Goldberg and Nivre, 2012, 2013); it may mean using alternative search strategies, such as transition-based parsing with beam search (Johansson and Nugues, 2007; Titov and Henderson, 2007; Zhang and Clark, 2008) or exact search (Huang and Sagae, 2010; Kuhlmann et al., 2011) or graph-based parsing with heuristic search to cope with the complexity of higher-order models, especially for non-projective parsing (McDonald and Pereira, 2006; Koo et al., 2010; Zhang and McDonald, 2012); or it may mean hybrid or en-

semble systems (Sagae and Lavie, 2006; Nivre and McDonald, 2008; Zhang and Clark, 2008; Bohnet and Kuhn, 2012). A nice illustration of the impact of new techniques can be found in Zhang and Nivre (2012), where an error analysis along the lines of McDonald and Nivre (2007, 2011) shows that a transition-based parser using global learning and beam search (instead of local learning and greedy search) performs on par with graph-based parsers for long dependencies, while retaining the advantage of the original transition-based parsers on short dependencies (see Figure 1).

Neural networks for dependency parsing, first explored by Titov and Henderson (2007) and Attardi et al. (2009), have come to dominate the field during the last five years. While this has dramatically changed learning architectures and feature representations, most parsing models are still either transition-based (Chen and Manning, 2014; Dyer et al., 2015; Weiss et al., 2015; Andor et al., 2016; Kiperwasser and Goldberg, 2016) or graph-based (Kiperwasser and Goldberg, 2016; Dozat and Manning, 2017). However, more accurate feature learning using continuous representations and nonlinear models has allowed parsing architectures to be simplified. Thus, most recent transition-based parsers have moved back to local learning and greedy inference, seemingly without losing accuracy (Chen and Manning, 2014; Dyer et al., 2015; Kiperwasser and Goldberg, 2016). Similarly, graph-based parsers again rely on first-order models and obtain no improvements from using higher-order models (Kiperwasser and Goldberg, 2016; Dozat and Manning, 2017).

The increasing use of neural networks has also led to a convergence in feature representations and learning algorithms for transition-based and graph-based parsers. In particular, most recent systems rely on an encoder, typically in the form of a BiLSTM, that provides contextualized representations of the input words as input to the scoring of transitions – in transition-based parsers – or of dependency arcs – in graph-based parsers. By making information about the *global* sentence context available in *local* word representations, this encoder can be assumed to mitigate error propagation for transition-based parsers and to widen the feature scope beyond individual word pairs for graph-based parsers. For both types of parsers, this also obviates the need for complex structural feature templates, as recently shown by

Falenska and Kuhn (2019). We should therefore expect neural transition-based and graph-based parsers to be not only more accurate than their non-neural counterparts but also more similar to each other in their error profiles.

### 3 Deep Contextualized Word Representations

Neural parsers rely on vector representations of words as their primary input, often in the form of pretrained word embeddings such as word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), or fastText (Bojanowski et al., 2016), which are sometimes extended with character-based representations produced by recurrent neural networks (Ballesteros et al., 2015). These techniques assign a single static representation to each word type and therefore cannot capture context-dependent variation in meaning and syntactic behavior.

By contrast, deep contextualized word representations encode words with respect to the sentential context in which they appear. Like word embeddings, such models are typically trained with a language-modeling objective, but yield sentence-level tensors as representations, instead of single vectors. These representations are typically produced by transferring a model’s entire feature encoder – be it a BiLSTM (Hochreiter and Schmidhuber, 1997) or Transformer (Vaswani et al., 2017) – to a target task, where the dimensionality of the tensor  $S$  is typically  $S \in \mathbb{R}^{N \times L \times D}$  for a sentence of length  $N$ , an encoder with  $L$  layers, and word-level vectors of dimensionality  $D$ . The advantage of such models, compared to the parser-internal encoders discussed in the previous section, is that they not only produce contextualized representations but do so over several layers of abstraction, as captured by the model’s different layers, and are pre-trained on corpora much larger than typical treebanks.

Deep contextualized embedding models have proven to be adept at a wide array of NLP tasks, achieving state-of-the-art performance in standard Natural Language Understanding (NLU) benchmarks, such as GLUE (Wang et al., 2019). Though many such models have been proposed, we adopt the two arguably most popular ones for our experiments: ELMo and BERT. Both models have previously been used for dependency parsing (Che et al., 2018; Jawahar et al., 2018; Lim et al., 2018;

Kondratyuk, 2019; Schuster et al., 2019), but there has been no systematic analysis of their impact on transition-based and graph-based parsers.

### 3.1 ELMo

ELMo is a deep contextualized embedding model proposed by Peters et al. (2018), which produces sentence-level representations yielded by a multi-layer BiLSTM language model. ELMo is trained with a standard language-modeling objective, in which a BiLSTM reads a sequence of  $N$  learned context-independent embeddings  $\mathbf{w}_1, \dots, \mathbf{w}_N$  (obtained via a character-level CNN) and produces a context-dependent representation  $\mathbf{h}_{j,k} = \text{BiLSTM}(\mathbf{w}_{1:N}, k)$ , where  $j$  ( $1 \leq j \leq L$ ) is the BiLSTM layer and  $k$  is the index of the word in the sequence. The output of the last layer  $\mathbf{h}_{L,k}$  is then employed in conjunction with a softmax layer to predict the next token at  $k + 1$ .

The simplest way of transferring ELMo to a downstream task is to encode the input sentence  $S = w_1, \dots, w_N$  by extracting the representations from the BiLSTM at layer  $L$  for each token  $w_k \in S$ :  $\mathbf{h}_{L,1}, \dots, \mathbf{h}_{L,N}$ . However, Peters et al. (2018) posit that the best way to take advantage of ELMo’s representational power is to compute a linear combination of BiLSTM layers:

$$\text{ELMo}_k = \gamma \sum_{j=0}^L s_j \mathbf{h}_{j,k} \quad (1)$$

where  $s_j$  is a softmax-normalized task-specific parameter and  $\gamma$  is a task-specific scalar. Peters et al. (2018) demonstrate that this scales the layers of linguistic abstraction encoded by the BiLSTM for the task at hand.

### 3.2 BERT

BERT (Devlin et al., 2019) is similar to ELMo in that it employs a language-modeling objective over unannotated text in order to produce deep contextualized embeddings. However, BERT differs from ELMo in that, in place of a BiLSTM, it employs a bidirectional Transformer (Vaswani et al., 2017), which, among other factors, carries the benefit of learning potential dependencies between words directly. This lies in contrast to recurrent models, which may struggle to learn correspondences between constituent signals when the time-lag between them is long (Hochreiter et al., 2001). For a token  $w_k$  in sentence  $S$ , BERT’s input representation is composed by summing a

word embedding  $\mathbf{x}_k$ , a position embedding  $\mathbf{i}_k$ , and a WordPiece embedding  $\mathbf{s}_k$  (Wu et al., 2016):  $\mathbf{w}_k = \mathbf{x}_k + \mathbf{i}_k + \mathbf{s}_k$ .

Each  $\mathbf{w}_k \in S$  is passed to an  $L$ -layered Bi-Transformer, which is trained with a masked language modeling objective (i.e., randomly masking a percentage of input tokens and only predicting said tokens). For use in downstream tasks, Devlin et al. (2019) propose to extract the Transformer’s encoding of each token  $w_k \in S$  at layer  $L$ , which effectively produces  $\text{BERT}_k$ .

## 4 Hypotheses

Based on our discussion in Section 2, we assume that transition-based and graph-based parsers still have distinctive error profiles due to the basic trade-off between rich structural features, which allow transition-based parsers to make accurate local decisions, and global learning and exact search, which give graph-based parsers an advantage with respect to global sentence structure. At the same time, we expect the differences to be less pronounced than they were ten years ago because of the convergence in neural architectures and feature representations. But how will the addition of deep contextualized word representations affect the behavior of the two parsers?

Given recent work showing that deep contextualized word representations incorporate rich information about syntactic structure (Goldberg, 2019; Liu et al., 2019; Tenney et al., 2019; Hewitt and Manning, 2019), we hypothesize that transition-based parsers have most to gain from these representations because it will improve their capacity to make decisions informed by global sentence structure and therefore reduce the number of search errors. Our main hypothesis can be stated as follows:

Deep contextualized word representations are more effective at reducing errors in transition-based parsing than in graph-based parsing.

If this holds true, then the analysis of McDonald and Nivre (2007, 2011) suggests that the differential error reduction should be especially visible on phenomena such as:

1. longer dependencies,
2. dependencies closer to the root,
3. certain parts of speech,
4. certain dependency relations,
5. longer sentences.

The error analysis will consider all these factors as well as non-projective dependencies.

## 5 Experimental Setup

### 5.1 Parsing Architecture

To be able to compare transition-based and graph-based parsers under equivalent conditions, we use and extend UUParser<sup>1</sup> (de Lhoneux et al., 2017a; Smith et al., 2018a), an evolution of bilstparser (Kiperwasser and Goldberg, 2016), which supports transition-based and graph-based parsing with a common infrastructure but different scoring models and parsing algorithms.

For an input sentence  $S = w_1, \dots, w_N$ , the parser creates a sequence of vectors  $\mathbf{w}_{1:N}$ , where the vector  $\mathbf{w}_k = \mathbf{x}_k \circ \text{BiLSTM}(c_{1:M})$  representing input word  $w_k$  is the concatenation of a pre-trained word embedding  $\mathbf{x}_k$  and a character-based embedding  $\text{BiLSTM}(c_{1:M})$  obtained by running a BiLSTM over the character sequence  $c_{1:M}$  of  $w_k$ . Finally, each input element is represented by a BiLSTM vector,  $\mathbf{h}_k = \text{BiLSTM}(\mathbf{w}_{1:N}, k)$ .

In transition-based parsing, the BiLSTM vectors are input to a multi-layer perceptron (MLP) for scoring transitions, using the arc-hybrid transition system from Kuhlmann et al. (2011) extended with a SWAP transition to allow the construction of non-projective dependency trees (Nivre, 2009; de Lhoneux et al., 2017b). The scoring is based on the top three words on the stack and the first word of the buffer, and the input to the MLP includes the BiLSTM vectors for these words as well as their leftmost and rightmost dependents (up to 12 words in total).

In graph-based parsing, the BiLSTM vectors are input to an MLP for scoring all possible dependency relations under an arc-factored model, meaning that only the vectors corresponding to the head and dependent are part of the input (2 words in total). The parser then extracts a maximum spanning tree over the score matrix using the Chu-Liu-Edmonds (CLE) algorithm<sup>2</sup> (Edmonds, 1967) which allows us to construct non-projective trees.

It is important to note that, while we acknowledge the existence of graph-based parsers that outperform the implementation of Kiperwasser and Goldberg (2016), such models do not meet our criteria for systematic comparison. The parser

by Dozat et al. (2017) is very similar, but employs the MLP as a further step in the featurization process prior to scoring via a biaffine classifier. To keep the comparison as exact as possible, we forego comparing our transition-based systems to the Dozat et al. (2017) parser (and its numerous modifications). In addition, preliminary experiments showed that our chosen graph-based parser outperforms its transition-based counterpart, which was itself competitive in the CoNLL 2018 shared task (Zeman et al., 2018).

### 5.2 Input Representations

In our experiments, we evaluate three pairs of systems – differing only in their input representations. The first is a baseline that represents tokens by  $\mathbf{w}_k = \mathbf{x}_k \circ \text{BiLSTM}(c_{1:M})$ , as described in Section 5.1. The word embeddings  $\mathbf{x}_k$  are initialized via pretrained fastText vectors ( $\mathbf{x}_k \in \mathbb{R}^{300}$ ) (Grave et al., 2018), which are updated for the parsing task. We term these transition-based and graph-based baselines TR and GR.

For the ELMo experiments, we make use of pretrained models provided by Che et al. (2018), who train ELMo on 20 million words randomly sampled from raw WikiDump and Common Crawl datasets for 44 languages. We encode each gold-segmented sentence in our treebank via the ELMo model for that language, which yields a tensor  $\mathbf{S}_{\text{ELMo}} = \mathbb{R}^{N \times L \times D}$ , where  $N$  is the number of words in the sentence,  $L = 3$  is the number of ELMo layers, and  $D = 1024$  is the ELMo vector dimensionality. Following Peters et al. (2018) (see Eq. 1), we learn a linear combination and a task-specific  $\gamma$  of each token’s ELMo representation, which yields a vector  $\text{ELMo}_k \in \mathbb{R}^{1024}$ . We then concatenate this vector with  $\mathbf{w}_k$  and pass it to the BiLSTM. We call the transition-based and graph-based systems enhanced with ELMo TR+E and GR+E.

For the BERT experiments, we employ the pretrained multilingual cased model provided by Google,<sup>34</sup> which is trained on the concatenation of WikiDumps for the top 104 languages with the largest Wikipedias.<sup>5</sup> The model’s parameters feature a 12-layer transformer trained with 768 hid-

<sup>3</sup><https://github.com/google-research/bert>

<sup>4</sup>Except for Chinese, for which we make use of a separate, pretrained model.

<sup>5</sup>See sorted list here: [https://meta.wikimedia.org/wiki/List\\_of\\_Wikipedias](https://meta.wikimedia.org/wiki/List_of_Wikipedias)

<sup>1</sup><https://github.com/UppsalaNLP/uuparser>

<sup>2</sup>We use the implementation from Qi et al. (2018).

den units and 12 self-attention heads. In order to obtain a word-level vector for each token in a sentence, we experimented with a variety of representations: namely, concatenating each transformer layer’s word representation into a single vector  $\mathbf{w}_{concat} \in \mathbb{R}^{768*12}$ , employing the last layer’s representation, or learning a linear combination over a range of layers, as we do with ELMo (e.g., via Eq. 1). In a preliminary set of experiments, we found that the latter approach over layers 4–8 consistently yielded the best results, and thus chose to adopt this method going forward. Regarding tokenization, we select the vector for the first subword token, as produced by the native BERT tokenizer. Surprisingly, this gave us better results than averaging subword token vectors in a preliminary round of experiments. Like with the ELMo representations, we concatenate each BERT vector  $\mathbf{BERT}_k \in \mathbb{R}^{768}$  with  $\mathbf{w}_k$  and pass it to the respective TR+B and GR+B parsers.

It is important to note that while the ELMo models we work with are *monolingual*, the BERT model is *multilingual*. In other words, while the standalone ELMo models were trained on the tokenized WikiDump and CommonCrawl for each language respectively, the BERT model was trained only on the former, albeit simultaneously for 104 languages. This means that the models are not strictly comparable, and it is an interesting question whether either of the models has an advantage in terms of training regime. However, since our purpose is not to compare the two models but to study their impact on parsing, we leave this question for future work.

### 5.3 Language and Treebank Selection

For treebank selection, we rely on the criteria proposed by [de Lhoneux et al. \(2017c\)](#) and adapted by [Smith et al. \(2018b\)](#) to have languages from different language families, with different morphological complexity, different scripts and character set sizes, different training sizes and domains, and with good annotation quality. This gives us 13 treebanks from UD v2.3 ([Nivre et al., 2018](#)), information about which is shown in Table 1.

### 5.4 Parser Training and Evaluation

In all experiments, we train parsers with default settings<sup>6</sup> for 30 epochs and select the model with

<sup>6</sup>All hyperparameters are specified in the supplementary material (Part A).

Language	Treebank	Family	Order	Train
Arabic	PADT	non-IE	VSO	6.1k
Basque	BDT	non-IE	SOV	5.4k
Chinese	GSD	non-IE	SVO	4.0k
English	EWT	IE	SVO	12.5k
Finnish	TDT	non-IE	SVO	12.2k
Hebrew	HTB	non-IE	SVO	5.2k
Hindi	HDTB	IE	SOV	13.3k
Italian	ISDT	IE	SVO	13.1k
Japanese	GSD	non-IE	SOV	7.1k
Korean	GSD	non-IE	SOV	4.4k
Russian	SynTagRus	IE	SVO	48.8k
Swedish	Talbanken	IE	SVO	4.3k
Turkish	IMST	non-IE	SOV	3.7k

Table 1: Languages and treebanks used in experiments. Family = Indo-European (IE) or not. Order = dominant word order according to WALS ([Haspelmath et al., 2005](#)). Train = number of training sentences.

the best labeled attachment score on the dev set. For each combination of model and training set, we repeat this procedure three times with different random seeds, apply the three selected models to the test set, and report the average result.

### 5.5 Error Analysis

In order to conduct an error analysis along the lines of [McDonald and Nivre \(2007, 2011\)](#), we extract all sentences from the smallest development set in our treebank sample (Hebrew HTB, 484 sentences) and sample the same number of sentences from each of the other development sets (6,292 sentences in total). For each system, we then extract parses of these sentences for the three training runs with different random seeds (18,876 predictions in total). Although it could be interesting to look at each language separately, we follow [McDonald and Nivre \(2007, 2011\)](#) and base our main analysis on all languages together to prevent data sparsity for longer dependencies, longer sentences, etc.<sup>7</sup>

## 6 Results and Discussion

Table 2 shows labeled attachment scores for the six parsers on all languages, averaged over three training runs with random seeds. The results clearly corroborate our main hypothesis. While ELMo and BERT provide significant improvements for both transition-based and graph-based

<sup>7</sup>The supplementary material contains tables for the error analysis (Part B) and graphs for each language (Part C).

Language	TR	GR	TR+E	GR+E	TR+B	GR+B
Arabic	79.1	79.9	82.0	81.7	81.9	81.8
Basque	73.6	77.6	80.1	81.4	77.9	79.8
Chinese	75.3	76.7	79.8	80.4	83.7	83.4
English	82.7	83.3	87.0	86.5	87.8	87.6
Finnish	80.0	81.4	87.0	86.6	85.1	83.9
Hebrew	81.1	82.4	85.2	85.9	85.5	85.9
Hindi	88.4	89.6	91.0	91.2	89.5	90.8
Italian	88.0	88.2	90.9	90.6	92.0	91.7
Japanese	92.1	92.2	93.1	93.0	92.9	92.1
Korean	79.6	81.2	82.3	82.3	83.7	84.2
Russian	88.3	88.0	90.7	90.6	91.5	91.0
Swedish	80.5	81.6	86.9	86.2	87.6	86.9
Turkish	57.8	61.2	62.6	63.8	64.2	64.9
<b>Average</b>	<b>80.5</b>	<b>81.8</b>	<b>84.5</b>	<b>84.6</b>	<b>84.9</b>	<b>84.9</b>

Table 2: Labeled attachment score on 13 languages for parsing models with and without deep contextualized word representations.

parsers, the magnitude of the improvement is greater in the transition-based case: 3.99 vs. 2.85 for ELMo and 4.47 vs. 3.13 for BERT. In terms of error reduction, this corresponds to 21.1% vs. 16.5% for ELMo and 22.5% vs. 17.4% for BERT. The differences in error reduction are statistically significant at  $\alpha = 0.01$  (Wilcoxon).

Although both parsing accuracy and absolute improvements vary across languages, the overall trend is remarkably consistent and the transition-based parser improves more with both ELMo and BERT for every single language. Furthermore, a linear mixed effect model analysis reveals that, when accounting for language as a random effect, there are no significant interactions between the improvement of each model (over its respective baseline) and factors such as language family (IE vs. non-IE), dominant word order, or number of training sentences. In other words, the improvements for both parsers seem to be largely independent of treebank-specific factors. Let us now see to what extent they can be explained by the error analysis.

## 6.1 Dependency Length

Figure 2 shows labeled F-score for dependencies of different lengths, where the length of a dependency between words  $w_i$  and  $w_j$  is equal to  $|i - j|$  (and with root tokens in a special bin on the far left). For the baseline parsers, we see that the curves diverge with increasing length, clearly indicating that the transition-based parser still suffers

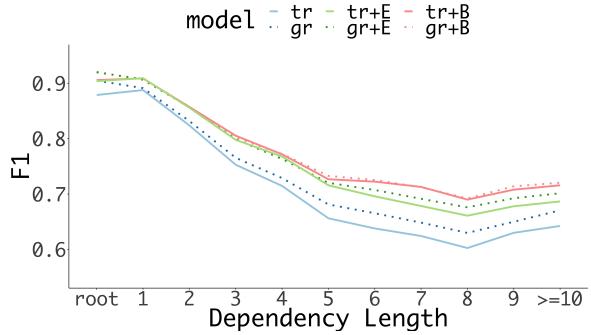


Figure 2: Labeled F-score by dependency length.

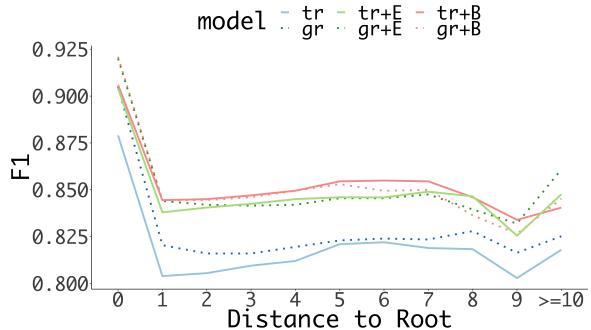


Figure 3: Labeled F-score by distance to root.

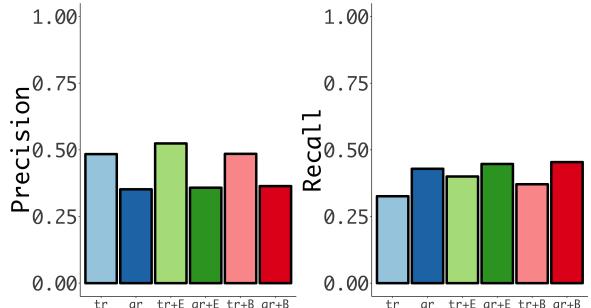


Figure 4: Labeled precision (left) and recall (right) for non-projective dependencies.

from search errors on long dependencies, which require longer transition sequences for their construction. However, the differences are much smaller than in McDonald and Nivre (2007, 2011) and the transition-based parser no longer has an advantage for short dependencies, which is consistent with the BiLSTM architecture providing the parsers with more similar features that help the graph-based parser overcome the limited scope of the first-order model.

Adding deep contextualized word representations clearly helps the transition-based parser to perform better on longer dependencies. For ELMo there is still a discernible difference for dependencies longer than 5, but for BERT the two curves

are almost indistinguishable throughout the whole range. This could be related to the aforementioned intuition that a Transformer captures long dependencies more effectively than a BiLSTM (see Tran et al. (2018) for contrary observations, albeit for different tasks). The overall trends for both baseline and enhanced models are quite consistent across languages, although with large variations in accuracy levels.

## 6.2 Distance to Root

Figure 3 reports labeled F-score for dependencies at different distances from the root of the tree, where distance is measured by the number of arcs in the path from the root. There is a fairly strong (inverse) correlation between dependency length and distance to the root, so it is not surprising that the plots in Figure 3 largely show the mirror image of the plots in Figure 2. For the baseline parsers, the graph-based parser has a clear advantage for dependencies near the root (including the root itself), but the transition-based parser closes the gap with increasing distance.<sup>8</sup> For ELMo and BERT, the curves are much more similar, with only a slight advantage for the graph-based parser near the root and with the transition-based BERT parser being superior from distance 5 upwards. The main trends are again similar across all languages.

## 6.3 Non-Projective Dependencies

Figure 4 shows precision and recall specifically for non-projective dependencies. We see that there is a clear tendency for the transition-based parser to have better precision and the graph-based parser better recall.<sup>9</sup> In other words, non-projective dependencies are more likely to be correct when they are predicted by the transition-based parser using the swap transition, but real non-projective dependencies are more likely to be found by the graph-based parser using a spanning tree algorithm. Interestingly, adding deep contextualized word representations has almost no effect on the graph-based parser,<sup>10</sup> while especially the ELMo em-

<sup>8</sup>At the very end, the curves appear to diverge again, but the data is very sparse in this part of the plot.

<sup>9</sup>Incidentally, the same pattern is reported by McDonald and Nivre (2007, 2011), even though the techniques for processing non-projective dependencies are different in that study: pseudo-projective parsing (Nivre and Nilsson, 2005) for the transition-based parser and approximate second-order non-projective parsing (McDonald and Pereira, 2006) for the graph-based parser.

<sup>10</sup>The breakdown per language shows marginal improvements for the enhanced graph-based models on a few lan-

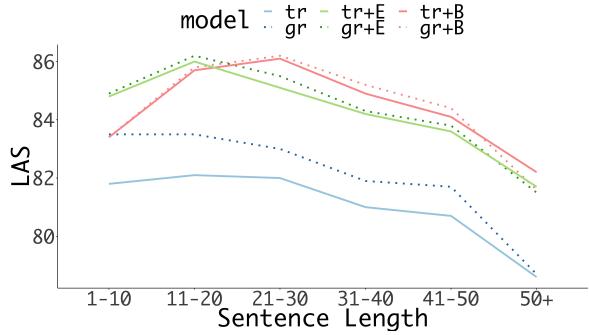


Figure 5: Labeled attachment score by sentence length.

beddings improve both precision and recall for the transition-based parser.

## 6.4 Parts of Speech and Dependency Types

Thanks to the cross-linguistically consistent UD annotations, we can relate errors to linguistic categories more systematically than in the old study. The main impression, however, is that there are very few clear differences, which is again indicative of the convergence between the two parsing approaches. We highlight the most notable differences and refer to the supplementary material (Part B) for the full results.

Looking first at parts of speech, the baseline graph-based parser is slightly more accurate on verbs and nouns than its transition-based counterpart, which is consistent with the old study for verbs but not for nouns. After adding the deep contextualized word representations, both differences are essentially eliminated.

With regard to dependency relations, the baseline graph-based parser has better precision and recall than the baseline transition-based parser for the relations of coordination (conj), which is consistent with the old study, as well as clausal subjects (csubj) and clausal complements (ccomp), which are relations that involve verbs in clausal structures. Again, the differences are greatly reduced in the enhanced parsing models, especially for clausal complements, where the transition-based parser with ELMo representations is even slightly more accurate than the graph-based parser.

## 6.5 Sentence Length

Figure 5 plots labeled attachment score for sentences of different lengths, measured by number of words in bins of 1–10, 11–20, etc. Here we

guages, canceled out by equally marginal degradations on others.

find the most unexpected results of the study. First of all, although the baseline parsers exhibit the familiar pattern of accuracy decreasing with sentence length, it is not the transition-based but the graph-based parser that is more accurate on short sentences and degrades faster. In other words, although the transition-based parser still seems to suffer from search errors, as shown by the results on dependency length and distance to the root, it no longer seems to suffer from error propagation in the sense that earlier errors make later errors more probable. The most likely explanation for this is the improved training for transition-based parsers using dynamic oracles and aggressive exploration to learn how to behave optimally also in non-optimal configurations (Goldberg and Nivre, 2012, 2013; Kiperwasser and Goldberg, 2016).

Turning to the models with deep contextualized word representations, we find that transition-based and graph-based parsers behave more similarly, which is in line with our hypotheses. However, the most noteworthy result is that accuracy *improves* with increasing sentence length. For ELMo this holds only from 1–10 to 11–20, but for BERT it holds up to 21–30, and even sentences of length 31–40 are parsed with higher accuracy than sentences of length 1–10. A closer look at the breakdown per language reveals that this picture is slightly distorted by different sentence length distributions in different languages. More precisely, high-accuracy languages seem to have a higher proportion of sentences of mid-range length, causing a slight boost in the accuracy scores of these bins, and no single language exhibits exactly the patterns shown in Figure 5. Nevertheless, several languages exhibit an increase in accuracy from the first to the second bin or from the second to the third bin for one or more of the enhanced models (especially the BERT models). And almost all languages show a less steep degradation for the enhanced models, clearly indicating that deep contextualized word representations improve the capacity to parse longer sentences.

## 7 Conclusion

In this paper, we have essentially replicated the study of McDonald and Nivre (2007, 2011) for neural parsers. In the baseline setting, where parsers use pre-trained word embeddings and character representations fed through a BiLSTM, we can still discern the basic trade-off identified

in the old study, with the transition-based parser suffering from search errors leading to lower accuracy on long dependencies and dependencies near the root of the tree. However, important details of the picture have changed. The graph-based parser is now as accurate as the transition-based parser on shorter dependencies and dependencies near the leaves of the tree, thanks to improved representation learning that overcomes the limited feature scope of the first order model. And with respect to sentence length, the pattern has actually been reversed, with the graph-based parser being more accurate on short sentences and the transition-based parser gradually catching up thanks to new training methods that prevent error propagation.

When adding deep contextualized word representations, the behavior of the two parsers converge even more, and the transition-based parser in particular improves with respect to longer dependencies and dependencies near the root, as a result of fewer search errors thanks to enhanced information about the global sentence structure. One of the most striking results, however, is that both parsers improve their accuracy on longer sentences, with some models for some languages in fact being more accurate on medium-length sentences than on shorter sentences. This is a milestone in parsing research, and more research is needed to explain it.

In a broader perspective, we hope that future studies on dependency parsing will take the results obtained here into account and extend them by investigating other parsing approaches and neural network architectures. Indeed, given the rapid development of new representations and architectures, future work should include analyses of how all components in neural parsing architectures (embeddings, encoders, decoders) contribute to distinct error profiles (or lack thereof).

## Acknowledgments

We want to thank Ali Basirat, Christian Hardmeier, Jamie Henderson, Ryan McDonald, Paola Merlo, Gongbo Tang, and the EMNLP reviewers and area chairs for valuable feedback on preliminary versions of this paper. We acknowledge the computational resources provided by CSC in Helsinki and Sigma2 in Oslo through NeIC-NPL ([www.nlpl.eu](http://www.nlpl.eu)).

## References

- Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2442–2452.
- Giuseppe Attardi, Felice DellOrletta, Maria Simi, and Joseph Turian. 2009. Accurate dependency parsing with a stacked multilayer perceptron. In *Proceedings of EVALITA 2009*.
- Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2015. Improved transition-based parsing by modeling characters instead of words with LSTMs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 349–359.
- Ezra Black, Frederick Jelinek, John D. Lafferty, David M. Magerman, Robert L. Mercer, and Salim Roukos. 1992. Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings of the 5th DARPA Speech and Natural Language Workshop*, pages 31–37.
- Bernd Bohnet and Jonas Kuhn. 2012. The best of both worlds – a graph-based completion model for transition-based parsers. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 77–87.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.
- Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 149–164.
- Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of the CoNLL Shared Task of EMNLP-CoNLL 2007*, pages 957–961.
- Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64.
- Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750.
- 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*.
- Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of the 5th International Conference on Learning Representations*.
- Timothy Dozat, Peng Qi, and Christopher D. Manning. 2017. Stanford’s graph-based neural dependency parser at the conll 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30.
- Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 334–343.
- Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards*, 71B:233–240.
- Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 340–345.
- Agnieszka Falenska and Jonas Kuhn. 2019. The (non-)utility of structural features in BiLSTM-based dependency parsers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 117–128.
- Yoav Goldberg. 2019. Assessing BERT’s syntactic abilities. *CoRR*, abs/1901.05287.
- Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 959–976.
- Yoav Goldberg and Joakim Nivre. 2013. Training deterministic parsers with non-deterministic oracles. *Transactions of the Association for Computational Linguistics*, 1:403–414.
- Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.
- Martin Haspelmath, Matthew S. Dryer, David Gil, and Bernard Comrie. 2005. *The World Atlas of Language Structures*. Oxford University Press.

- John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.
- Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1077–1086.
- Ganesh Jawahar, Benjamin Muller, Amal Fethi, Louis Martin, Eric Villemonte de la Clergerie, Benoît Sagot, and Djamel Seddah. 2018. ELMoLex: Connecting ELMo and lexicon features for dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 223–237.
- Richard Johansson and Pierre Nugues. 2007. Incremental dependency parsing using online learning. In *Proceedings of the CoNLL Shared Task of EMNLP-CoNLL 2007*, pages 1134–1138.
- Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.
- Daniel Kondratyuk. 2019. 75 languages, 1 model: Parsing universal dependencies universally. *CoRR*, abs/1904.02099.
- Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1–11.
- Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1288–1298.
- Marco Kuhlmann, Carlos Gómez-Rodríguez, and Giorgio Satta. 2011. Dynamic programming algorithms for transition-based dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 673–682.
- Miryam de Lhoneux, Yan Shao, Ali Basirat, Eliyahu Kiperwasser, Sara Stymne, Yoav Goldberg, and Joakim Nivre. 2017a. From raw text to Universal Dependencies – Look, no tags! In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 207–217.
- Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2017b. Arc-hybrid non-projective dependency parsing with a static-dynamic oracle. In *Proceedings of the 15th International Conference on Parsing Technologies*, pages 99–104.
- Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2017c. Old school vs. new school: Comparing transition-based parsers with and without neural network enhancement. In *Proceedings of the 15th Treebanks and Linguistic Theories Workshop (TLT)*.
- KyungTae Lim, Cheoneum Park, Changki Lee, and Thierry Poibeau. 2018. SEx BiST: A multi-source trainable parser with deep contextualized lexical representations. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 143–152.
- Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. *CoRR*, abs/1903.08855.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 91–98.
- Ryan McDonald, Kevin Lerman, and Fernando Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 216–220.
- Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131.
- Ryan McDonald and Joakim Nivre. 2011. Analyzing and integrating dependency parsers. *Computational Linguistics*, pages 197–230.
- Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 81–88.
- Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic̄. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in*

*Natural Language Processing (HLT/EMNLP)*, pages 523–530.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 149–160.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34:513–553.

Joakim Nivre. 2009. Non-projective dependency parsing in expected linear time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, pages 351–359.

Joakim Nivre, Mitchell Abrams, Željko Agić, Lars Ahrenberg, Lene Antonsen, Katya Aplonova, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Virginica Barbu Mititelu, Victoria Basarov, John Bauer, Sandra Bellato, Kepa Bengoetxea, Yevgeni Berzak, Irshad Ahmad Bhat, Riyaz Ahmad Bhat, Erica Biagetti, Eckhard Bick, Rogier Blokland, Victoria Bobicev, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane Boyd, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Gülsen Cebiroğlu Eryigit, Flavio Massimiliano Cecchini, Giuseppe G. A. Celano, Slavomír Čéplö, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Jayeol Chun, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraz, Carly Dickerson, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Tomaž Erjavec, Aline Etienne, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Sebastian Garza, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökirmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta González Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Nizar Habash, Jan Hajic, Jan Hajic jr., Linh Hà Mý, Na-Rae Han, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Radu Ion, Elena Irimia, Olájide Ishola, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşikara, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Boris Katz, Tolga Kayadelen, Jessica Kenney, Václava Kettnerová, Jesse Kirchner, Kamil

Kopacewicz, Natalia Kotyba, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Lucia Lam, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phuong Lê H`ong, Alessandro Lenci, Saran Lerpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, KyungTae Lim, Nikola Ljubešić, Olga Loganova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Niko Miekka, Margarita Misirpashayeva, Anna Missilä, Cătălin Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Keiko Sophie Mori, Shinsuke Mori, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Yugo Murawaki, Kaili Müürisepp, Pinkey Nainwani, Juan Ignacio Navarro Horňiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Luong Nguy`en Thij, Huy`en Nguy`en Thị Minh, Vitaly Nikolaev, Rattima Nitisoroj, Hanna Nurmi, Stina Ojala, Adédayo Olúòkun, Mai Omura, Petya Osenova, Robert Östling, Lilja Øvrelid, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Guilherme Paulino-Passos, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Thierry Poibeau, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przeźiókowski, Tiina Puolakainen, Sampo Pyysalo, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Michael Rießler, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Valentin Roca, Olga Rudina, Jack Rueter, Shoval Sadde, Benoît Sagot, Shadi Saleh, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Baiba Saulīte, Yanin Sawanakunanon, Nathan Schneider, Sebastian Schuster, Djamel Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Muh Shohibussirri, Dmitry Sichinava, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Isabela Soares-Bastos, Carolyn Spadine, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Dima Taji, Yuta Takahashi, Takaaki Tanaka, Isabelle Tellier, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Eric Villemonte de la Clergerie, Veronika Vincze, Lars Wallin, Jing Xian Wang, Jonathan North Washington, Seyi Williams, Mats Wirén, Tsegay Wolde-mariam, Tak-sum Wong, Chunxiao Yan, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Daniel Zeman, Manying Zhang, and Hanzhi Zhu. 2018. *Universal dependencies 2.3*. LINDAT/CLARIN digital library at the Institute of Foreign Languages, University of Prague.

mal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task of EMNLP-CoNLL 2007*, pages 915–932.

Joakim Nivre, Johan Hall, Jens Nilsson, Gülsen Eryiğit, and Svetoslav Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 221–225.

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 950–958.

Joakim Nivre and Jens Nilsson. 2005. Pseudo-projective dependency parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 99–106.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. 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 2227–2237.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the 2018 CoNLL Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, page 160.

Kenji Sagae and Alon Lavie. 2006. Parser combination by reparsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 129–132.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Aaron Smith, Bernd Bohnet, Miryam de Lhoneux, Joakim Nivre, Yan Shao, and Sara Stymne. 2018a. 82 treebanks, 34 models: Universal dependency parsing with multi-treebank models. In *Proceedings of the 2018 CoNLL Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Aaron Smith, Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2018b. An investigation of the interactions between pre-trained word embeddings, character models and pos tags in dependency parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *Proceedings of the 5th International Conference on Learning Representations*.

Ivan Titov and James Henderson. 2007. A latent variable model for generative dependency parsing. In *Proceedings of the 10th International Conference on Parsing Technologies (IWPT)*, pages 144–155.

Ke Tran, Arianna Bisazza, and Christof Monz. 2018. The importance of being recurrent for modeling hierarchical structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4731–4736.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Jorn Veenstra and Walter Daelemans. 2000. A memory-based alternative for connectionist shift-reduce parsing. Technical Report ILK-0012, Tilburg University.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 7th International Conference on Learning Representations*.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 323–333.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 195–206.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthiyersast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Hao Zhang and Ryan McDonald. 2012. Generalized higher-order dependency parsing with cube pruning. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 320–331.

Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 562–571.

Yue Zhang and Joakim Nivre. 2011. Transition-based parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 188–193.

Yue Zhang and Joakim Nivre. 2012. Analyzing the effect of global learning and beam-search on transition-based dependency parsing. In *Proceedings of COLING 2012: Posters*, pages 1391–1400.

# Supplementary Material

## A Hyperparameters

Component	Specification
$w_k$ dim.	500 ( $100 * 2$ (char) + 300 (word))
$ELMo_k$ dim.	1024
$BERT_k$ dim.	768
BiLSTM output dim.	125
MLP output dim.	100
No. BiLSTM layers	2
BiLSTM dropout rate	0.33
No. training epochs	30
Model selection	best dev. performance

## B Error Analysis

### B.1 Dependency Length

	GR		GR+B		GR+E		TR		TR+B		TR+E	
	pr	rc										
root	0.91	0.91	0.92	0.92	0.92	0.92	0.88	0.88	0.91	0.91	0.90	0.90
1	0.89	0.90	0.90	0.91	0.90	0.91	0.88	0.90	0.90	0.91	0.90	0.92
2	0.83	0.84	0.85	0.86	0.86	0.86	0.82	0.83	0.86	0.86	0.85	0.86
3	0.76	0.77	0.80	0.80	0.80	0.80	0.76	0.75	0.81	0.80	0.80	0.80
4	0.74	0.72	0.78	0.77	0.77	0.76	0.73	0.70	0.78	0.76	0.78	0.76
5	0.70	0.67	0.74	0.72	0.73	0.71	0.67	0.64	0.74	0.71	0.73	0.71
6	0.68	0.66	0.73	0.72	0.72	0.70	0.65	0.63	0.73	0.72	0.70	0.69
7	0.66	0.64	0.72	0.70	0.70	0.69	0.64	0.61	0.72	0.71	0.68	0.67
8	0.65	0.61	0.71	0.67	0.70	0.66	0.63	0.58	0.71	0.67	0.68	0.64
9	0.66	0.64	0.71	0.72	0.69	0.70	0.64	0.63	0.71	0.70	0.68	0.67
$\geq 10$	0.69	0.66	0.73	0.71	0.71	0.69	0.65	0.63	0.72	0.71	0.70	0.68

## B.2 Distance to Root

	GR		GR+B		GR+E		TR		TR+B		TR+E	
	pr	rc										
0	0.91	0.91	0.92	0.92	0.92	0.92	0.88	0.88	0.91	0.91	0.90	0.90
1	0.83	0.81	0.85	0.84	0.85	0.84	0.81	0.80	0.85	0.84	0.84	0.83
2	0.81	0.82	0.84	0.85	0.84	0.84	0.81	0.81	0.84	0.85	0.84	0.84
3	0.81	0.82	0.85	0.85	0.84	0.84	0.81	0.81	0.85	0.85	0.84	0.84
4	0.82	0.82	0.85	0.85	0.84	0.84	0.81	0.82	0.85	0.85	0.85	0.85
5	0.82	0.83	0.85	0.86	0.84	0.85	0.81	0.83	0.85	0.86	0.84	0.85
6	0.82	0.83	0.84	0.86	0.84	0.85	0.82	0.83	0.85	0.86	0.84	0.85
7	0.82	0.83	0.85	0.86	0.85	0.85	0.81	0.83	0.85	0.86	0.85	0.85
8	0.82	0.84	0.83	0.85	0.84	0.84	0.81	0.83	0.84	0.85	0.85	0.85
9	0.83	0.81	0.83	0.82	0.84	0.83	0.79	0.81	0.84	0.83	0.83	0.82
$\geq 10$	0.81	0.84	0.84	0.85	0.85	0.87	0.80	0.84	0.83	0.85	0.83	0.87

## B.3 Projectivity

	GR		GR+B		GR+E		TR		TR+B		TR+E	
	pr	rc										
Non-proj	0.35	0.43	0.36	0.45	0.36	0.45	0.48	0.33	0.49	0.37	0.52	0.40
Proj	0.83	0.83	0.86	0.85	0.85	0.85	0.81	0.82	0.85	0.85	0.85	0.85

## B.4 Part of Speech

	GR	GR+B	GR+E	TR	TR+B	TR+E
ADJ	0.82	0.85	0.85	0.82	0.85	0.85
ADP	0.95	0.96	0.96	0.95	0.96	0.96
ADV	0.78	0.80	0.80	0.77	0.80	0.81
AUX	0.93	0.93	0.94	0.92	0.93	0.94
CCONJ	0.85	0.88	0.87	0.84	0.87	0.86
DET	0.94	0.95	0.95	0.94	0.95	0.95
INTJ	0.50	0.44	0.48	0.44	0.45	0.48
NOUN	0.77	0.81	0.80	0.76	0.80	0.80
NUM	0.83	0.84	0.84	0.82	0.84	0.84
PART	0.86	0.89	0.87	0.84	0.90	0.88
PRON	0.83	0.85	0.86	0.82	0.85	0.86
PROPN	0.79	0.84	0.82	0.78	0.84	0.83
PUNCT	0.82	0.85	0.84	0.81	0.86	0.84
SCONJ	0.90	0.93	0.93	0.89	0.93	0.93
SYM	0.85	0.84	0.84	0.82	0.85	0.86
VERB	0.76	0.80	0.79	0.74	0.79	0.78
X	0.64	0.67	0.65	0.64	0.68	0.67

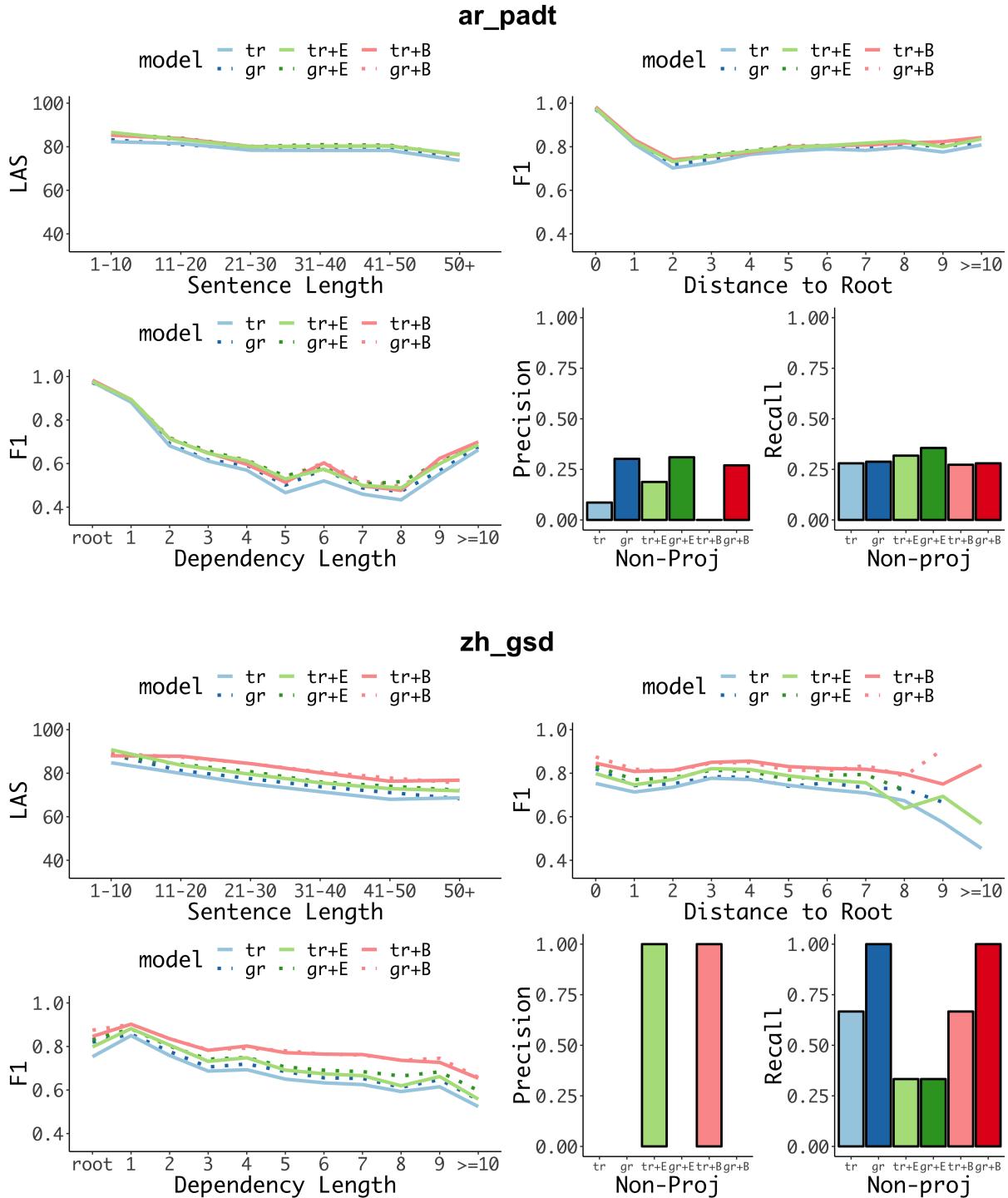
## B.5 Dependency Relation

	GR		GR+B		GR+E		TR		TR+B		TR+E	
	pr	rc										
acl	0.69	0.69	0.75	0.76	0.73	0.73	0.66	0.66	0.75	0.75	0.73	0.73
advcl	0.68	0.69	0.68	0.71	0.70	0.72	0.66	0.67	0.69	0.71	0.70	0.71
advmmod	0.77	0.77	0.79	0.78	0.80	0.79	0.76	0.76	0.79	0.80	0.80	0.80
amod	0.86	0.87	0.88	0.89	0.89	0.90	0.85	0.87	0.88	0.89	0.88	0.90
appos	0.55	0.54	0.61	0.63	0.59	0.60	0.54	0.57	0.62	0.64	0.59	0.61
aux	0.95	0.95	0.95	0.95	0.96	0.96	0.95	0.95	0.95	0.95	0.96	0.96
case	0.95	0.96	0.96	0.97	0.96	0.96	0.95	0.95	0.96	0.97	0.96	0.96
cc	0.84	0.85	0.87	0.88	0.87	0.87	0.83	0.83	0.87	0.88	0.86	0.86
ccomp	0.67	0.64	0.71	0.67	0.71	0.66	0.65	0.60	0.72	0.68	0.71	0.68
clf	0.87	0.90	0.94	0.95	0.92	0.92	0.89	0.87	0.95	0.94	0.92	0.90
compound	0.85	0.84	0.86	0.87	0.87	0.86	0.84	0.84	0.87	0.87	0.86	0.87
conj	0.68	0.69	0.76	0.75	0.72	0.72	0.64	0.65	0.74	0.74	0.71	0.70
cop	0.84	0.85	0.86	0.86	0.87	0.87	0.83	0.84	0.85	0.86	0.86	0.87
csubj	0.59	0.51	0.62	0.54	0.63	0.56	0.55	0.49	0.64	0.53	0.63	0.55
dep	0.68	0.56	0.74	0.60	0.72	0.58	0.65	0.56	0.73	0.60	0.69	0.57
det	0.93	0.94	0.94	0.95	0.94	0.95	0.92	0.93	0.95	0.95	0.94	0.95
discourse	0.74	0.61	0.78	0.62	0.75	0.65	0.72	0.60	0.79	0.65	0.76	0.56
dislocated	0.40	0.32	0.43	0.32	0.44	0.39	0.28	0.26	0.34	0.36	0.35	0.44
expl	0.75	0.81	0.79	0.82	0.79	0.81	0.75	0.79	0.79	0.83	0.81	0.82
fixed	0.78	0.72	0.81	0.76	0.82	0.77	0.78	0.73	0.82	0.76	0.82	0.78
flat	0.79	0.78	0.85	0.83	0.83	0.81	0.79	0.78	0.83	0.82	0.82	0.82
goeswith	0.71	0.44	0.92	0.31	0.68	0.44	0.61	0.28	0.84	0.41	0.75	0.46
iobj	0.83	0.81	0.82	0.80	0.86	0.82	0.83	0.78	0.85	0.79	0.85	0.83
list	0.31	0.28	0.32	0.32	0.38	0.33	0.28	0.23	0.34	0.33	0.41	0.36
mark	0.89	0.87	0.91	0.91	0.91	0.91	0.88	0.87	0.91	0.91	0.91	0.90
nmod	0.78	0.78	0.81	0.81	0.80	0.81	0.77	0.77	0.81	0.81	0.80	0.81
nsubj	0.79	0.80	0.82	0.83	0.82	0.83	0.77	0.78	0.82	0.83	0.81	0.82
nummod	0.87	0.90	0.88	0.91	0.88	0.91	0.87	0.89	0.88	0.92	0.88	0.92
obj	0.79	0.80	0.82	0.83	0.83	0.84	0.77	0.78	0.83	0.83	0.84	0.83
obl	0.74	0.75	0.77	0.77	0.77	0.78	0.73	0.73	0.77	0.77	0.77	0.78
orphan	0.29	0.11	0.31	0.12	0.35	0.11	0.21	0.10	0.35	0.13	0.38	0.15
parataxis	0.77	0.70	0.80	0.72	0.77	0.72	0.75	0.70	0.80	0.74	0.79	0.73
punct	0.82	0.82	0.85	0.85	0.84	0.84	0.81	0.81	0.86	0.86	0.84	0.84
root	0.90	0.90	0.92	0.92	0.92	0.92	0.90	0.88	0.92	0.91	0.92	0.90
vocative	0.43	0.28	0.43	0.32	0.65	0.43	0.53	0.35	0.58	0.38	0.52	0.38
xcomp	0.68	0.62	0.71	0.66	0.70	0.68	0.65	0.62	0.74	0.69	0.73	0.70

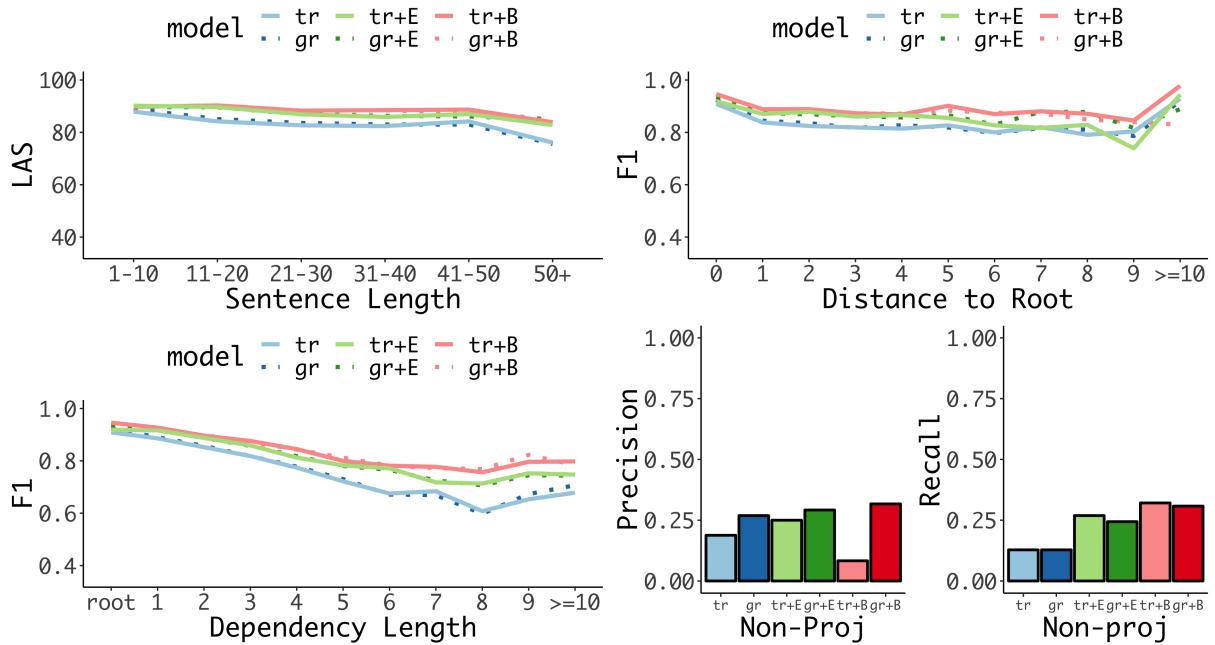
## B.6 Sentence Length

	GR	GR+B	GR+E	TR	TR+B	TR+E
1-10	83.50	83.40	84.90	81.80	83.40	84.80
11-20	83.50	85.80	86.20	82.10	85.70	86.00
21-30	83.00	86.20	85.50	82.00	86.10	85.10
31-40	81.90	85.20	84.30	81.00	84.90	84.20
41-50	81.70	84.40	83.80	80.70	84.10	83.60
50+	78.70	81.60	81.50	78.60	82.20	81.70

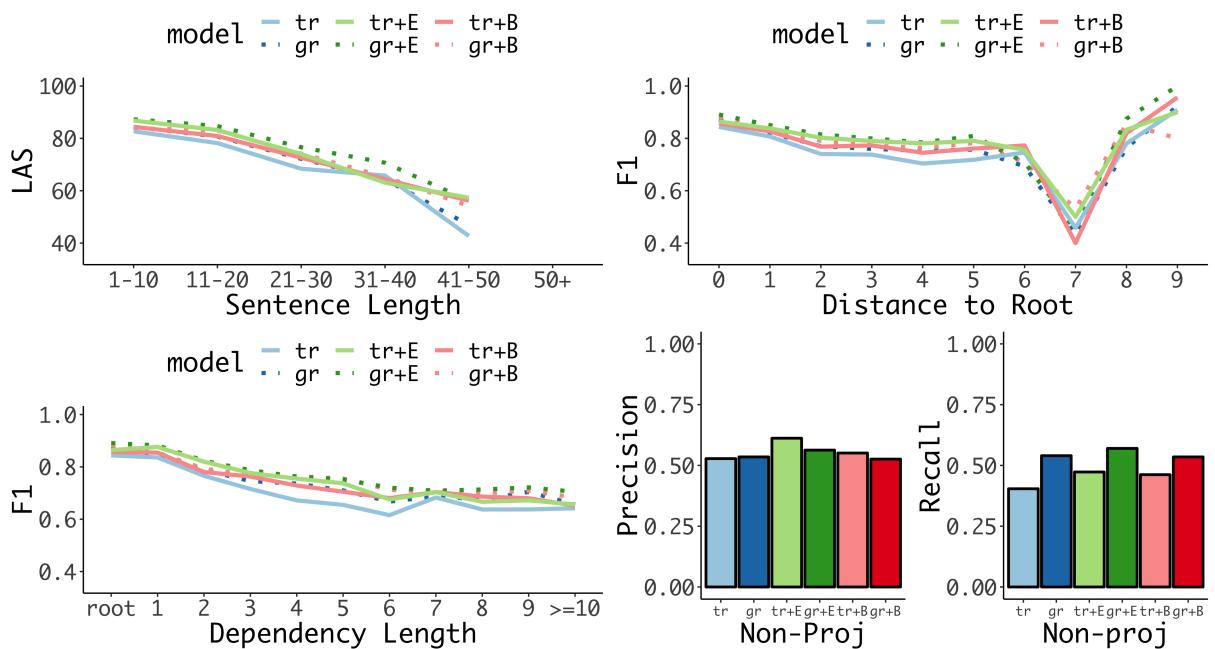
### C Per-Language Error Analysis



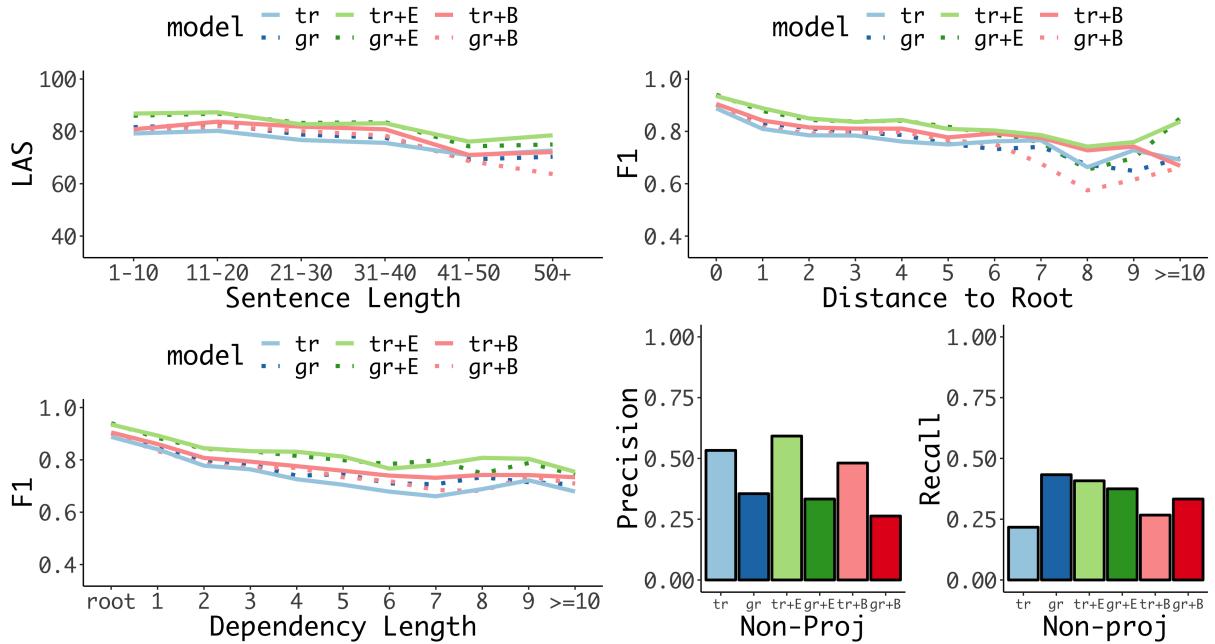
### en\_ewt



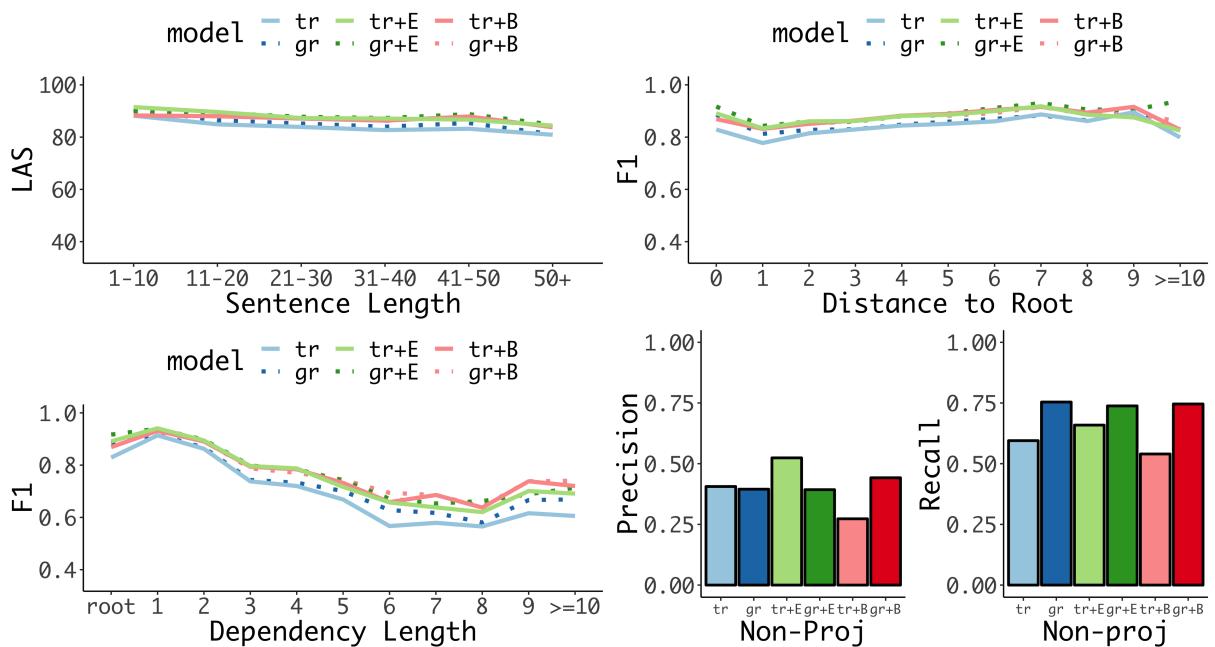
### eu\_bdt



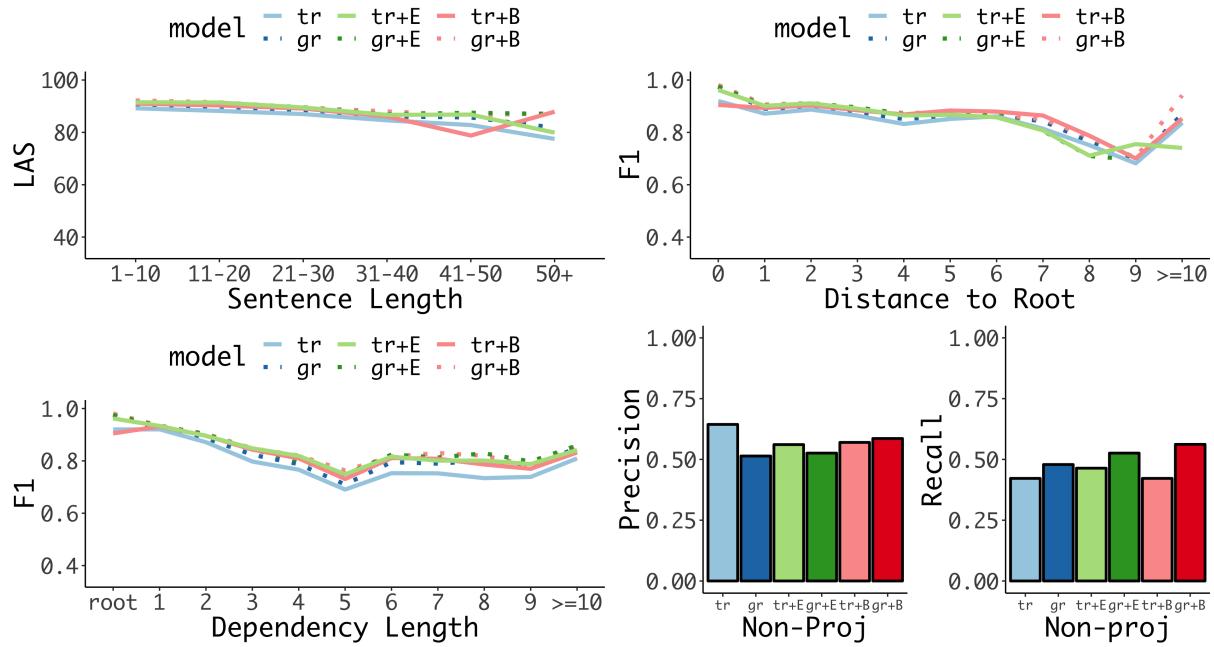
### fi\_tdt



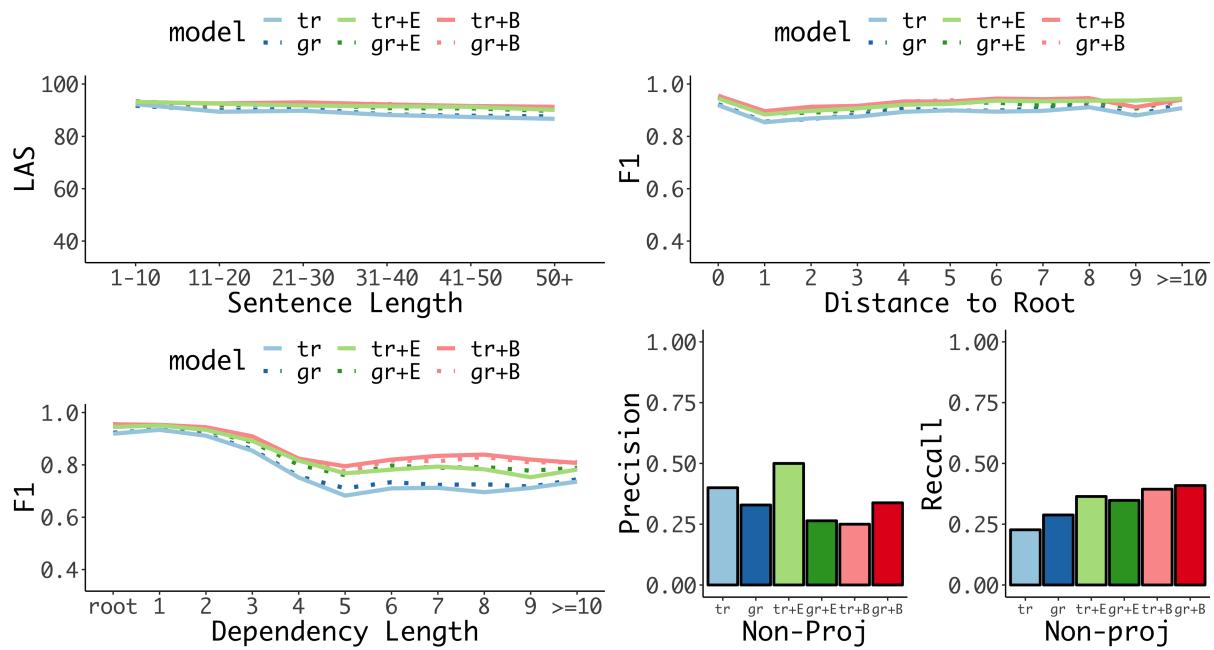
### he\_htb



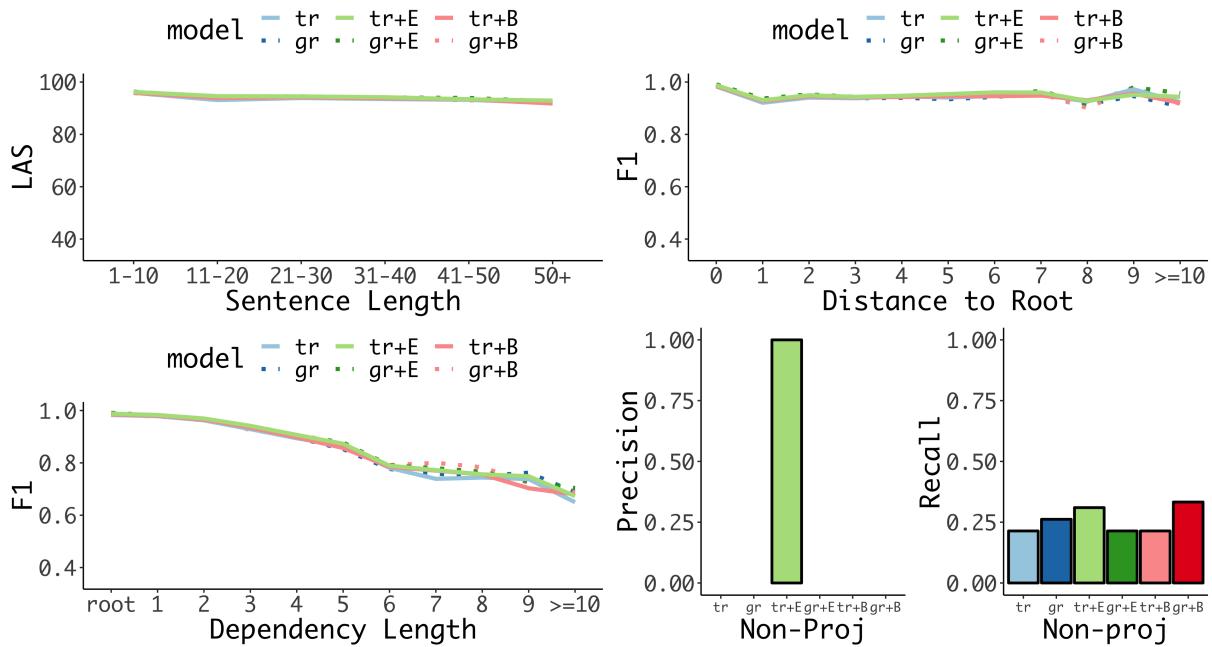
### hi\_hdtb



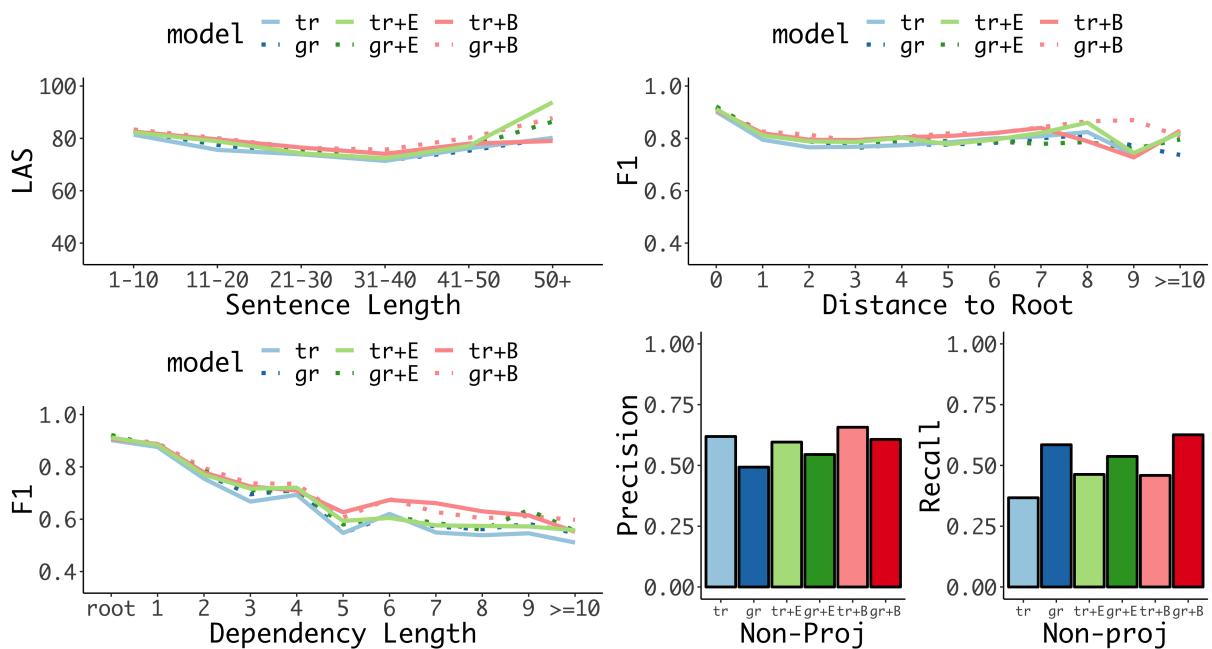
### it\_isdt



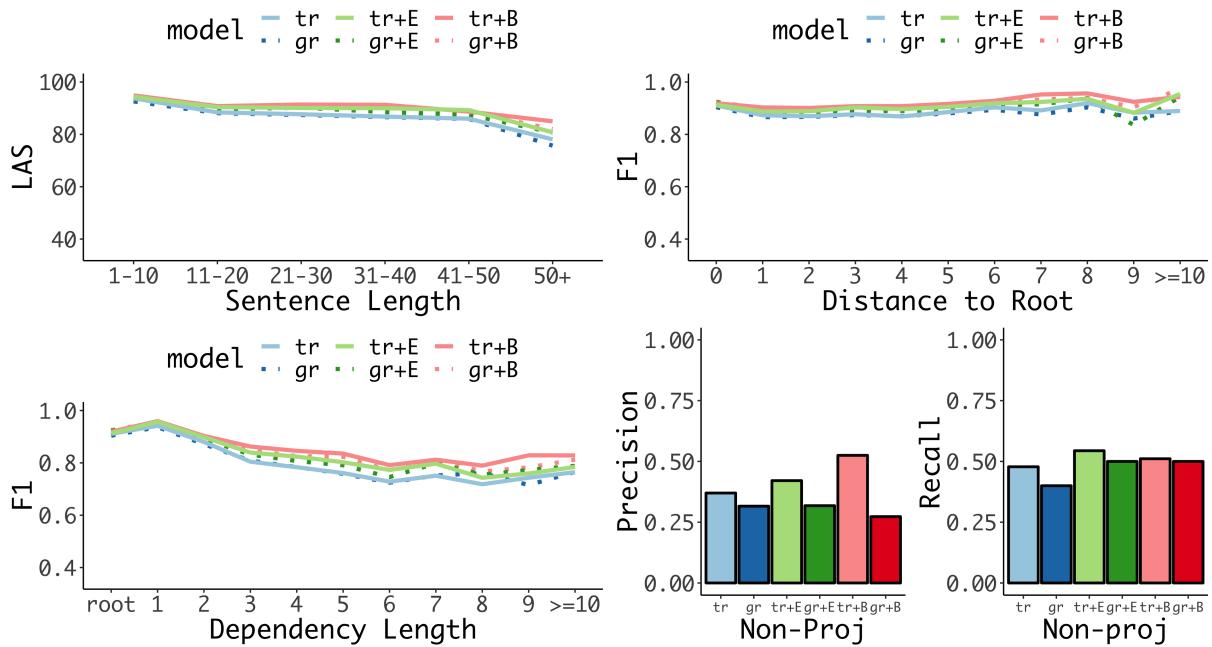
### ja\_gsd



### ko\_gsd



### ru\_syntagrus



### sv\_talbanken

