Is Supervised Syntactic Parsing Beneficial for Language Understanding Tasks? An Empirical Investigation

Goran Glavaš
University of Mannheim
Data and Web Science Group
goran@informatik.uni-mannheim.de

Ivan Vulić
University of Cambridge
Language Technology Lab
iv250@cam.ac.uk

Abstract

Traditional NLP has long held (supervised) syntactic parsing necessary for successful higher-level semantic language understanding (LU). The recent advent of end-to-end neural models, self-supervised via language modeling (LM), and their success on a wide range of LU tasks, however, questions this belief. In this work, we empirically investigate the usefulness of supervised parsing for semantic LU in the context of LM-pretrained transformer networks. Relying on the established fine-tuning paradigm, we first couple a pretrained transformer with a biaffine parsing head, aiming to infuse explicit syntactic knowledge from Universal Dependencies treebanks into the transformer. We then fine-tune the model for LU tasks and measure the effect of the intermediate parsing training (IPT) on downstream LU task performance. Results from both monolingual English and zero-shot language transfer experiments (with intermediate target-language parsing) show that explicit formalized syntax, injected into transformers through IPT, has very limited and inconsistent effect on downstream LU performance. Our results, coupled with our analysis of transformers' representation spaces before and after intermediate parsing, make a significant step towards providing answers to an essential question: how (un)availing is supervised parsing for high-level semantic natural language understanding in the era of large neural models?

1 Introduction

Structural analysis of sentences, based on a variety of syntactic formalisms (Charniak, 1996; Taylor et al., 2003; De Marneffe et al., 2006; Hockenmaier and Steedman, 2007; Nivre et al., 2016, 2020, *inter alia*), has been the beating heart of NLP pipelines for decades (Klein and Manning, 2003; Chen and Manning, 2014; Dozat and Manning, 2017; Kondratyuk and Straka, 2019), establishing rather strong common belief that high-level semantic language understanding (LU) crucially depends on explicit syntax. The unprecedented success of neural language learning models based on transformer networks (Vaswani et al., 2017), trained on unlabeled corpora via language modeling (LM) objectives (Devlin et al., 2019; Liu et al., 2019b; Clark et al., 2020, *inter alia*) on a wide variety of LU tasks (Wang et al., 2018; Hu et al., 2020), however, questions this widely accepted assumption.

The question of necessity of supervised parsing for LU and NLP in general has been raised before. More than a decade ago, Bod (2007) questioned the superiority of supervised parsing over unsupervised induction of syntactic structures in the context of statistical machine translation. Nonetheless, the NLP community has since still managed to find sufficient evidence for the usefulness of explicit syntax in higher-level LU tasks (Levy and Goldberg, 2014; Cheng and Kartsaklis, 2015; Bastings et al., 2017; Kasai et al., 2019; Zhang et al., 2019a, *inter alia*). However, we believe that the massive improvements brought about by the LM-pretrained transformers – unexposed to any explicit syntactic signal – warrant a renewed scrutiny of the utility of supervised parsing for high-level language understanding.[1,2] The research question we address in this work can be summarized as follows:

[1] **Disclaimer 1:** In this work, we make a clear distinction between Computational Linguistics (CL), i.e., the area of linguistics leveraging computational methods for analyses of human languages and NLP, the area of artificial intelligence tackling human language in order to perform intelligent tasks. This work scrutinizes the usefulness of supervised parsing and explicit syntax only for the latter. We find the usefulness of explicit syntax in CL to be self-evident.

[2] **Disclaimer 2:** The purpose of this work is definitely not to invalidate the admirable efforts on syntactic annotation and modeling, but rather to make an empirically driven step towards a deeper understanding of the relationship between LU and formalised syntactic knowledge, and the extent of its impact to modern semantic LU and applications.

(RQ) *Is explicit structural language information, provided in the form of a widely adopted syntactic formalism (Universal Dependencies, UD) (Nivre et al., 2016) and injected in a supervised manner into LM-pretrained transformers beneficial for transformers' downstream LU performance?*

While existing body of work (Lin et al., 2019; Tenney et al., 2019; Liu et al., 2019a; Kulmizev et al., 2020; Chi et al., 2020) probes transformers for structural phenomena, our work is more pragmatically motivated. We directly evaluate the effect of infusing structural language information from UD treebanks, via intermediate dependency parsing (DP) training, on transformers' performance in downstream LU. To this end, we couple a pretrained transformer with a biaffine parser similar to Dozat and Manning (2017), and train the model (i.e., fine-tune the transformer) for DP. Our parser on top of RoBERTa (Liu et al., 2019b) and XLM-R (Conneau et al., 2020) produces DP results which are comparable to state of the art. We then fine-tune the syntactically-informed transformers for three downstream LU tasks: natural language inference (NLI) (Williams et al., 2018; Conneau et al., 2018), paraphrase identification (Zhang et al., 2019b; Yang et al., 2019), and causal commonsense reasoning (Sap et al., 2019; Ponti et al., 2020). We quantify the contribution of explicit syntax by comparing LU performance of the transformer exposed to intermediate parsing training (IPT) and its counterpart directly fine-tuned for the downstream task. We investigate the effects of IPT (1) *monolingually*, by fine-tuning English transformers, BERT and RoBERTa, on an English UD treebank and for (2) downstream *zero-shot language transfer*, by fine-tuning massively multilingual transformers (MMTs) – mBERT and XLM-R (Conneau et al., 2020) – on treebanks of downstream target languages, before the downstream fine-tuning on source language (English) data.

While intermediate parsing training is obviously not the only way of bringing syntactic knowledge to downstream tasks (Kuncoro et al., 2019; Swayamdipta et al., 2019; Kuncoro et al., 2020), it is arguably the most straightforward way of injecting syntactic signal in the context of the predominant pretraining-fine-tuning paradigm that has, nonetheless, not been investigated up to this point. Other methods of bringing syntactic signal to downstream tasks such as knowledge distillation (Kuncoro et al., 2020) and pre-training on shallow trees instead of sequences (Swayamdipta et al., 2019) have failed to demonstrate significant gains on higher-level LU tasks.

Our results also render supervised UD parsing largely inconsequential to LU. We observe limited and inconsistent gains only in zero-shot downstream language transfer: further analyses reveal that (1) intermediate LM training yields comparable gains and (2) IPT only marginally changes representation spaces of transformers exposed to sufficient amount of language data in LM-pretraining. We hope that these empirical findings will shed new light on the relationship between supervised parsing (and manually labeled treebanks) and LU with transformer networks, and guide further similar investigations in future work, in order to fully understand the impact of formal syntactic knowledge on LU performance with modern neural architectures.

2 Related Work

Bringing Explicit Syntax to LMs. Previous work has attempted to enrich language models with explicit syntactic knowledge in ways other than intermediate parsing training. Swayamdipta et al. (2019) modify the pretraining objective of ELMo (Peters et al., 2018) to learn from shallowly parsed (i.e., chunked) corpora. They, however, report no notable improvements on downstream tasks. Kuncoro et al. (2019) propose to distil the knowledge from a Recurrent NN Grammar (RNNG) teacher trained on a small syntactically annotated corpus (by modeling the joint probability of surface sequence and phrase structure tree) into an LSTM-based student pretrained on a much larger corpus. They show that distillation helps the student in structured prediction tasks, but their downstream evaluation does not involve LU tasks. Their subsequent work (Kuncoro et al., 2020) replaces the RNN student with BERT (Devlin et al., 2019): syntactic distillation again helps structured prediction, but hurts (slightly) the performance on LU tasks from the GLUE benchmark (Wang et al., 2018).

Transformer-Based Dependency Parsing. Building on the success of preceding neural parsers (Chen and Manning, 2014; Kiperwasser and Goldberg, 2016), Dozat and Manning (2017) proposed a biaffine parsing head on top of a Bi-LSTM encoder: contextualized word vectors are fed to two feed-forward networks, producing dependent- and head-specific token representations, respectively. Arc and relation scores are produced via biaffine prod-

ucts between these dependent- and head-specific representation matrices. Finally, the Edmonds algorithm induces the optimal tree from pairwise arc predictions. Most recent DP work (Kondratyuk and Straka, 2019; Üstün et al., 2020) replaces the Bi-LSTM encoder with multilingual BERT's transformer, reporting state-of-the-art parsing performance. Kondratyuk and Straka (2019) fine-tune mBERTs parameters on the concatenation of all UD treebanks, whereas Üstün et al. (2020) freeze the original transformer's parameters and inject adapters (Houlsby et al., 2019) for parsing.

We propose and work with a simpler transformer-based biaffine parser: we apply biaffine attention directly on representations from transformer's output layer, eliminating the head- and dependendant-based feed-forward mapping. Despite this simplification, our biaffine parser produces DP results comparable to current state-of-the-art parsers.

Syntactic BERTology. The substantial body of syntactic probing work shows that BERT (Devlin et al., 2019) (a) encodes text in a hierarchical manner (i.e., it encodes some implicit underlying syntax) (Lin et al., 2019); and (b) captures specific shallow syntactic information (parts-of-speech and syntactic chunks) (Tenney et al., 2019; Liu et al., 2019a). Hewitt and Manning (2019) find that linear transformations, when applied on BERT's contextualized word vectors, reflect distances in dependency trees. This suggests that BERT encodes sufficient structural information to reconstruct dependency trees (though without arc directionality and relations). Chi et al. (2020) extend the analysis to multilingual BERT, finding that its representation subspaces may recover trees also for other languages. They also provide evidence that clusters of head–dependency pairs roughly correspond to UD relations. Similarly, Kulmizev et al. (2020) show that BERT's latent syntax corresponds more to UD trees than to shallower SUD (Gerdes et al., 2018) structures. Despite the evident similarity between BERT's latent syntax and formalisms such as UD, there is ample evidence that BERT insufficiently leverages syntax in downstream tasks: it often produces similar predictions for syntactically valid as well as for structurally corrupt sentences (e.g., with random word order) (Wallace et al., 2019; Ettinger, 2020; Zhao et al., 2020).

Intermediate Training. Sometimes called Supplementary Training on Intermediate Labeled-data Tasks (STILT) (Phang et al., 2018), intermediate

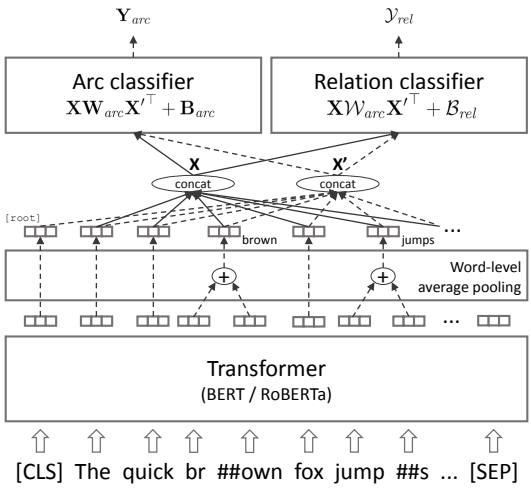

Figure 1: Architecture of our transformer-based biaffine dependency parser.

training is a transfer learning setup in which one trains an LM-pretrained transformer on one or more supervised tasks (ideally with large training sets) before final fine-tuning for the target task. Phang et al. (2018) show that intermediate NLI training of BERT on the Multi-NLI dataset (Williams et al., 2018) benefits several language understanding tasks. Subsequent work (Wang et al., 2019; Pruksachatkun et al., 2020) investigated many combinations of intermediate and target LU tasks, failing to identify any universally beneficial intermediate task. In this work we use DP as an intermediate training task (IPT) for LM-pretrained transformers.

3 Methodology

Biaffine Parser. Our parsing model, illustrated in Figure 1, consists of a biaffine attention layer applied directly on the transformer's output (BERT, RoBERTa, mBERT, or XLM-R). We first obtain word-level vectors by averaging transformed representations of their constituent subwords, produced by the transformer. Let $\mathbf{X} \in \mathbb{R}^{N \times H}$ denote the encoding of a sentence with N word-level tokens, consisting of N H-dimensional vectors (where H is the transformer's hidden size). We use the transformed representation of the sentence start token (e.g., `[CLS]` for BERT), $\mathbf{x}_{CLS} \in \mathbb{R}^{H}$, as the representation for the `root` node of the parse tree, and prepend it to $\mathbf{X}$, $\mathbf{X}' = [\mathbf{x}_{CLS}; \mathbf{X}] \in \mathbb{R}^{(N+1) \times H}$. We then use $\mathbf{X}$ as the representation of syntactic dependants and $\mathbf{X}'$ as the representation of dependency heads. We then directly compute the arc and relation scores as biaffine products of $\mathbf{X}$ and $\mathbf{X}'$:

$$\mathbf{Y}_{arc} = \mathbf{X}\mathbf{W}_{arc}\mathbf{X}'^{\top} + \mathbf{B}_{arc}; \;\; \mathbf{Y}_{rel} = \mathbf{X}\mathbf{W}_{rel}\mathbf{X}'^{\top} + \mathbf{B}_{rel}$$

where $\mathbf{W}_{arc} \in \mathbb{R}^{H \times H}$ and $\mathbf{W}_{rel} \in \mathbb{R}^{H \times H \times R}$ denote, respectively, the arc classification matrix and relation classification tensor (with R as the number of relations); $\mathbf{B}_{arc}$ and $\mathbf{B}_{rel}$ denote the corresponding bias parameters. We greedily select the dependency head for each word by finding the maximal score in each row of $\mathbf{Y}_{arc}$: while this is not guaranteed to produce a tree, Zhang et al. (2017) show that in most cases it does.[3] Our arc prediction loss is the cross-entropy loss with sentence words (plus the `root` node) as categorical labels: this implies a different number of labels for different sentences. We compute the relation prediction loss as a cross-entropy loss over gold arcs. Our final loss is the sum of the arc loss and relation loss.

Note that, in comparison with the original biaffine parser (Dozat and Manning, 2017) and its other transformer-based variants (Kondratyuk and Straka, 2019; Üstün et al., 2020), we feed word-level representations derived from the transformer's output directly to biaffine products, omitting the dependent- and head-specific MLP transformations. Deep task-specific architectures go against the fine-tuning idea: deep transformers have plenty of their own parameters that can be tuned for DP. We want to propagate as much of the explicit syntactic knowledge as possible into the transformer: a deep(er) DP-specific architecture on top of the transformer would impede the propagation of this knowledge to the transformer's parameters.

Downstream Models. After IPT, we fine-tune transformers for two *types* of LU tasks: (1) *sequence classification* (SEQC) tasks, where a sequence of text needs to be assigned a discrete label; and (2) *multiple choice classification* (MCC) tasks where we need to select the correct answer between two or more options for a given a premise and/or question. For SEQC, we simply apply a softmax classifier on the transformed representation of the sequence start token: $\mathbf{y} = softmax\left(\mathbf{x}_{CLS}\mathbf{W}_{sc} + \mathbf{b}_{sc}\right)$ (with $\mathbf{W}_{sc} \in \mathbb{R}^{H \times C}$ and $\mathbf{b}_{sc} \in \mathbb{R}^{C}$ as classifier's parameters and C as the number of task's labels).

For MCC tasks, we first concatenate each of the offered answer choices (independently of each other) to the premise and/or question, and encode it with the transformer. Since some of these tasks, e.g., COPA (Roemmele et al., 2011; Ponti et al., 2020), have very small training sets, we would like to support model transfer between different MCC tasks. Different multiple-choice classification tasks, however, may differ in the number of choices: a classifier with the number of parameters depending on the number of labels is thus not a good fit; instead, we follow Sap et al. (2019) and Ponti et al. (2020), and couple the transformer with a feed-forward network outputting a single scalar for each answer. Let $\mathbf{x}^i_{CLS} \in \mathbb{R}^H$ be the representation of the sequence start token (i.e., `[CLS]` or `<s>`) for the concatenation of the premise/question and the i-th answer. We obtain the score for the i-th answer as follows:

$$y_i = \mathbf{W}^o_{mcc} \tanh\left(\mathbf{W}^h_{mcc}\mathbf{x}^i_{CLS} + \mathbf{b}^h_{mcc}\right)$$

with $\mathbf{W}^h_{mcc} \in \mathbb{R}^{H \times H}$, $\mathbf{b}^h_{mcc} \in \mathbb{R}^H$ and $\mathbf{W}^o_{mcc} \in \mathbb{R}^{1 \times H}$ as parameters. We then apply a softmax function on the concatenation of y_i scores of all answers: $\mathbf{y} = \text{softmax}([y_1, \ldots, y_K])$, with K as the number of answers (i.e., labels) in the task. Finally, we compute the cross-entropy loss on $\mathbf{y}$.

4 Experimental Setup

We now detail experimental setup, where LU fine-tuning follows Intermediate Parsing Training (IPT).

4.1 Sequential Fine-Tuning

Our primary goal is to identify if injection of explicit syntax into transformers via supervised parsing training improves their downstream LU performance – this translates into sequential fine-tuning: (1) we first attach a biaffine parser from §3 on the transformer and train the whole model on a UD treebank; (2) we then couple the syntactically-informed transformer with the corresponding downstream classification head and perform final fine-tuning. We then compare the downstream performance of transformers with and without the IPT step.

Mono- vs. Cross-Lingual IPT Experiments. In the monolingual setup, we work with English (EN) transformers, BERT and RoBERTa, pretrained on EN corpora. In the zero-shot language transfer setup, where we work with multilingual models, mBERT and XLM-R (Conneau et al., 2020), we first train transformers via IPT on the UD treebank of the target language (i.e., a language with no downstream training data) before fine-tuning it on the EN training set of the LU task. We experiment with four target languages: German (DE),

[3] They also show the performance of greedy decoding to match that of decoding algorithms that produce optimal trees.

French (FR), Turkish (TR), and Chinese (ZH).[4]

Standard vs. Adapter-Based Fine-Tuning. Standard fine-tuning updates all transformer's parameters, which, for tasks with large training sets may have some drawbacks: (i) fine-tuning may last long and (ii) task-specific information may overwrite the useful distributional knowledge obtained during LM-pretraining. *Adapter-based fine-tuning* (Houlsby et al., 2019; Pfeiffer et al., 2020) remedies for these potential issues by keeping the original transformer's parameters frozen and inserting new *adapter parameters* in transformer layers. In fine-tuning, both sets of parameters are used to make predictions, but we only update adapters based on loss gradients. As the number of adapter parameters is only a fraction of the number of original parameters (3-8%), fine-tuning is also much faster.

Therefore, to account for the possibility of forgetting distributional knowledge in standard IPT fine-tuning, we also carry out adapter-based IPT. We follow Houlsby et al. (2019) and inject two *bottleneck adapters* into each transformer layer: first after the multi-head attention sublayer and another after the feed-forward sublayer. In downstream LU tasks, however, we unfreeze the original transformer parameters and fine-tune them together with adapters (now containing syntactic knowledge).

4.2 Language Understanding Tasks

We now outline the downstream LU tasks. For brevity, we report all the technical training and optimization details in the Supplementary Material.

NLI is a ternary sentence-pair classification task. We predict if the hypothesis is *entailed* by the premise, *contradicts* it, or neither. For monolingual EN experiments, we use Multi-NLI (Williams et al., 2018). In zero-shot transfer experiments, we train on EN Multi-NLI and evaluate on target language (DE, FR, TR, ZH) test portions of the multilingual XNLI dataset (Conneau et al., 2018). Models trained on the Multi-NLI datasets have been shown, however, to capture certain heuristics (e.g., lexical overlap) useful for many training instances rather than more complex and generalizable language inference (McCoy et al., 2020). Because of this, we additionally evaluate on the HANS dataset (McCoy et al., 2020), consisting of adversarial examples on which models that capture such heuristics fail.

Paraphrase Identification is a binary classification task where we predict if two sentences are mutual paraphrases. For EN, we train, validate, and test on respective portions of the PAWS dataset (Zhang et al., 2019b). In zero-shot language transfer, we evaluate on the test DE, FR, and ZH portions of the PAWS-X dataset (Yang et al., 2019).

Commonsense Reasoning. We evaluate on two multiple-choice classification (MCC) datasets. In monolingual evaluation, we use the SocialIQA (SIQA) dataset (Sap et al., 2019), testing models' ability to reason about social interactions. Each SIQA instance consists of a premise, a question, and three possible answers. For zero-shot language transfer experiments, we resort to the recently published XCOPA dataset (Ponti et al., 2020), obtained by translating test portions of the EN COPA (Choice of Plausible Alternatives) dataset (Roemmele et al., 2011) to 11 languages. As mentioned, (X)COPA is an MCC task, with each instance containing a premise, a question,[5] and two possible answers. Due to the very limited size of the EN COPA training set (mere 400 instances), we follow Ponti et al. (2020) and evaluate the models fine-tuned on SIQA (EN) on the XCOPA test portions (in TR and ZH).

4.3 Training and Optimization Details

All the transformer models with which we experiment – EN BERT, mBERT, EN RoBERTa, and XLM-R have $L = 12$ layers and hidden representations of size $H = 768$. We apply a dropout ($p = 0.1$) on the transformer outputs before forwarding them to the task-specific classification heads (i.e., biaffine parsing head in intermediate parsing training, and MCC or SEQC heads in downstream fine-tuning). We optimize the parameters using the Adam algorithm (Kingma and Ba, 2015): we found the initial learning rate of 10^{-5} to offer stable convergence in both intermediate parsing training and downstream fine-tuning for all LU tasks. We train for at most 30 epochs over the respective training set, with early stopping based on the development loss.[6] On UD treebanks and

[4] Selected languages vary in typological and etymological proximity to EN as the source language: DE is in the same (Germanic) branch of Indo-European languages, FR is from the different branch of the same family, whereas TR (Turkic) and ZH (Sino-Tibetan) belong to different language families.

[5] While SIQA has unconstrained questions, (X)COPA has only two question types: a) What is the CAUSE of this (premise)? and b) What is the RESULT of this (premise)?

[6] We measure the development loss every U update steps and stop the training if the loss does not decrease over 10 consecutive measurements. We set $U = 500$ in NLI training and $U = 250$ in all other training procedures.

Transformer	Fine-tune	EN (EWT)		DE (GSD)		FR (GSD)		TR (IMST)		ZH (GSD)	
		UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
BERT	Standard	91.9	89.3	–	–	–	–	–	–	–	–
	Adapter	90.1	87.3	–	–	–	–	–	–	–	–
RoBERTa	Standard	93.0	**90.5**	–	–	–	–	–	–	–	–
	Adapter	91.5	88.7	–	–	–	–	–	–	–	–
mBERT	Standard	91.5	88.9	76.3	72.0	94.1	91.3	75.5	67.5	87.0	**83.8**
	Adapter	89.6	86.8	75.1	70.1	92.8	89.7	66.4	57.8	81.0	77.4
XLM-R	Standard	**93.1**	**90.5**	**89.4**	**85.0**	**94.3**	**91.7**	**77.9**	**70.0**	79.0	75.6
	Adapter	91.4	88.6	88.3	83.8	93.1	90.3	72.1	64.1	73.8	70.3
Baseline: UDify (mBERT, Standard)		91.0	88.5	87.8	83.6	93.6	91.5	74.6	67.4	**87.9**	**83.8**

Table 1: Dependency parsing performance of our transformer-based biaffine parsers.

Transf.	Parsing FT	NLI	HANS	PAWS	SIQA
BERT	None	84.1	53.3	**92.4**	**60.7**
	Standard	**84.4**	**56.7**	91.9	58.8
	Adapter	84.1	53.3	**92.4**	58.3
RoBERTa	None	**88.4**	**67.4**	94.7	67.2
	Standard	87.7	64.5	**94.9**	66.5
	Adapter	87.9	66.3	94.7	**67.3**

Table 2: Downstream LU performance of monolingual EN transformers (BERT and RoBERTa). **None**: no IPT; **Standard**: IPT via standard fine-tuning; **Adapter**: IPT via adapter-based fine-tuning.

SIQA we train in batches of size 8, whereas on Multi-NLI and PAWS we train in batches of size 32. In Adapter-based IPT, we set the adapter size to 64 and use GELU (Hendrycks and Gimpel, 2016) as the activation function in adapter layers.

5 Evaluation

We first discuss parsing performance of our novel biaffine parser (see §3). We then show transformers' downstream LU performance after IPT, both in monolingual EN setting and in zero-shot transfer.

5.1 Results and Discussion

Parsing Performance. In order to judge the benefits of IPT in downstream LU, we must first verify parsing performance of our biaffine parser, i.e., that we successfully fine-tune transformers for DP. Table 1 shows that our biaffine parser gives state-of-the-art performance for all five languages in our study. Our (m)BERT-based parser outperforms UDify (Kondratyuk and Straka, 2019), also based on mBERT, for EN, FR, and TR, and performs comparably for ZH.[7] Our parser based on XLM-R additionally yields an improvement over UDify for DE as well. It is worth noting that UDify trains the mBERT-based parser (1) on the concatenation of all UD treebanks and that it (2) additionally exploits gold UPOS and lemma annotations. We train our parsers only on the training portion of the respective treebank without using any additional morpho-syntactic information.[8] Our mBERT-based parser outperforms our XLM-R-based parser only for ZH: this is likely due to a tokenization mismatch between XLM-R's subword tokenization for ZH and gold tokenization in the ZH-GSD treebank.[9]

Monolingual EN Results. Table 2 quantifies the effects of applying IPT with the EN-EWT UD treebank to BERT and RoBERTa. We report downstream LU performance on NLI, PAWS, and SIQA. The reported results do not favor supervised parsing (i.e., explicit syntax): compared to original transformers that have not been exposed to any explicit syntactic supervision, variants exposed to UD syntax via IPT (Standard, Adapter) fail to produce any significant gains for any of the downstream LU tasks. One cannot argue that the cause of this might be forgetting (i.e., overwriting) of the distributional knowledge obtained in LM pretraining during IPT: Adapter IPT variants, in which all distributional knowledge is preserved by design, also fail to yield any significant LU gains. IPT yields the largest gain (+3.4%) for BERT on HANS – the NLI dataset consisting of adversarial examples for which syntax deliberately affects the sentence meaning more directly. The same effect, however, is not there for RoBERTa, suggesting that the additional syntactic knowledge that BERT gets through IPT, RoBERTa seems to obtain through larger-scale pretraining.

Zero-Shot Language Transfer. We show the results obtained for zero-shot downstream language

[7] Our mBERT-based parser performs poorly for DE: the cause of it is unclear and this requires further investigation.

[8] Also, since absolute parsing performance is not the primary objective of this work, we did not perform extensive language-specific hyperparameter tuning. One could likely obtain better parsing scores than what we report in Table 1 with careful language-specific model selection.

[9] We explain this mismatch in the Appendix.

Transformer	Parse FT	XNLI				PAWS-X			XCOPA	
		DE	FR	TR	ZH	DE	FR	ZH	TR	ZH
mBERT	None	71.0	73.7	**63.0**	70.3	85.1	86.3	76.4	52.0	61.2
	Standard	71.4	72.9	61.5	**70.4**	85.4	86.9	**79.8**	**57.4**	**65.4**
	Adapter	**71.7**	**74.8**	62.5	70.2	**85.8**	**87.1**	78.7	50.4	61.6
XLM-R	None	77.1	**78.1**	73.4	73.8	**88.3**	**89.3**	**81.4**	**61.2**	66.4
	Standard	76.1	77.2	73.1	73.8	86.4	89.2	81.1	59.2	**67.4**
	Adapter	**77.8**	76.4	**73.9**	**74.7**	86.7	88.7	80.7	57.4	65.6

Table 3: Performance of multilingual transformers, mBERT and XLM-R, in zero-shot language transfer for downstream LU tasks, with and without prior intermediate dependency parsing training on target language treebanks.

transfer setup, for both mBERT and XLM-R, in Table 3. Again, these results do not particularly favor the intermediate injection of explicit syntactic information in general. However, in few cases we do observe gains from the intermediate target-language parsing training: e.g., 3% gain on PAWS-X for ZH as well as 4% and 5% gains on XCOPA for ZH and TR , respectively. Interestingly, all substantial improvements are obtained for mBERT; for XLM-R, the improvements are less consistent and less pronounced. This might be due to XLM-R's larger capacity which makes it less susceptible to the "curse of multilinguality" (Conneau et al., 2020): with the subword vocabulary twice as large as mBERT's, XLM-R is able to store more language-specific information. Also, XLM-R has seen substantially more target language data in LM-pretraining than mBERT for each language. This might mean that the larger IPT gains for mBERT come from mere exposure to additional target language text rather than from injection of explicit syntactic UD signal (see further analyses in §5.2).

5.2 Further Analysis and Discussion

We first compare the impact of IPT with the effect of additional LM training on the same raw data. We then quantify the topological modification that IPT makes in transformers' representation spaces.

Explicit Syntax or Just More Language Data? We scrutinize the IPT gains that we observe in some zero-shot language transfer experiments. We hypothesize that these gains may, at least in part, be credited to transformer simply seeing more target language data. To investigate this, we replace IPT with intermediate (masked) language modeling training (ILMT) on the same data (i.e., sentences from the respective treebank used in IPT) before final downstream LU fine-tuning. Because MLM is a self-supervised objective, we can credit all differences in downstream LU performance between ILMT and IPT variants of the same pretrained transformer to supervised parsing, i.e., to the injection of explicit UD knowledge.

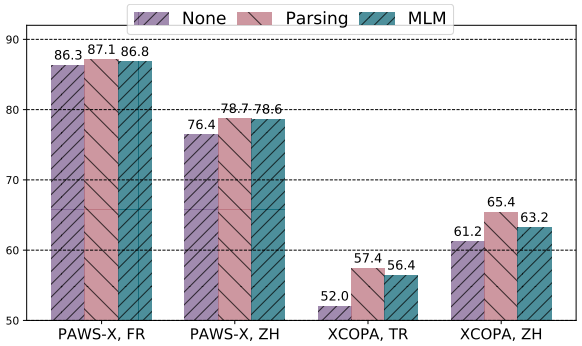

Figure 2: Comparison of IPT and ILMT in zero-shot language transfer experiments with mBERT on PAWS-X (FR and ZH) and XCOPA (TR and ZH). **None**: no intermediate training; **Parsing**: intermediate parsing training; **MLM**: intermediate masked LM training.

ILMT Details. We mask 15% of subword tokens in each sentence and predict them with a linear classifier applied on transformed representations of `[MASK]` tokens. We compute the cross-entropy loss and use the same hyperparameter configuration as described in §4.3. The development set, used for early stopping, is subdued to fixed masking, whereas we mask the training sentences dynamically, before feeding them to the transformer.

Results. We run this analysis for setups in which we observe substantial gains from IPT: PAWS-X for mBERT (*Adapter* fine-tuning, for FR and ZH) and XCOPA for mBERT (*Standard* fine-tuning, TR and ZH). The comparison between IPT and ILMT for these setups is provided in Figure 2. Like IPT, ILMT on mBERT generates downstream gains over direct downstream fine-tuning (i.e., no intermediate training) in all four setups. The gains from ILMT (with the exception of XCOPA for ZH) are almost as large as gains from IPT. This suggests that most of the gain with IPT comes from seeing more target language text, and prevents us from concluding that the explicit syntactic annotation is

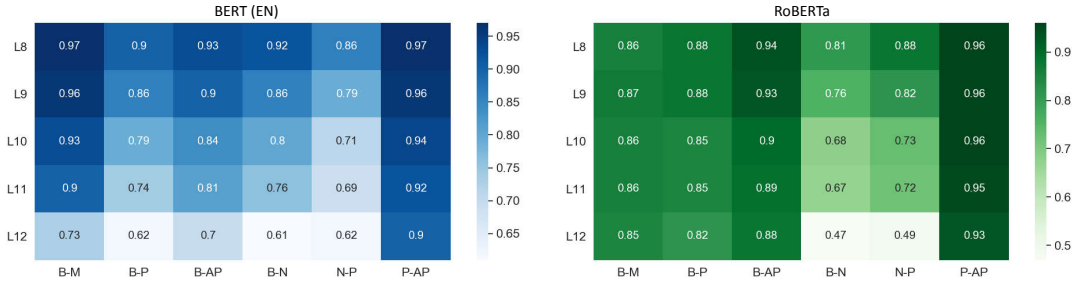

Figure 3: Topological similarity (l-CKA) for pairs of BERT and RoBERTa variants, before and after different fine-tuning steps (B, M, P, AP, and N). **Rows**: transformer layers; **Columns**: pairs of transformer variants in comparison.

responsible for the LU improvements in zero-shot downstream transfer. This interpretation is corroborated by the fact that IPT gains roughly correlate with the amount of language-specific data seen in LM-pretraining: the gains are more prominent for mBERT than for XLM-R and for TR and ZH than for FR and DE (see Table 3).

Changes in Representation Spaces. Finally, we analyze how fine-tuning transformers on different tasks modifies the topology of their representation spaces. We encode the set of sentences S from the test portions of treebanks used in IPT[10] with different variants: (a) Base (B): original LM-pretrained transformer, no further training; (b) MLM (M): after ILMT; (c) Parsing (P): after Standard IPT; and (d) Adapter-Parsing (AP): after Adapter-based IPT; for monolingual transformers (BERT and RoBERTa), also with (e) NLI (N): after NLI fine-tuning (without any intermediate training). We analyze the representations in each transformer layer separately: we represent each sentence $s \in S$ with the average of subword vectors from that layer (excluding sequence start and end tokens). Let $\mathbf{X}_1$ and $\mathbf{X}_2 \in \mathbb{R}^{|S| \times H}$ contain corresponding representations of sentences from S from the i-th layer of two transformer variants (e.g., B and P). We measure the topological similarity of the i-th layers of the two transformers with the linear centered kernel alignment (l-CKA) (Kornblith et al., 2019):[11]

$$l\text{-}CKA(\mathbf{X}_1, \mathbf{X}_2) = \frac{\left\|\mathbf{X}_2^\top \mathbf{X}_1\right\|_F^2}{\left(\left\|\mathbf{X}_1^\top \mathbf{X}_1\right\|_F\right)\left(\left\|\mathbf{X}_2^\top \mathbf{X}_2\right\|_F\right)}.$$

Although not invariant to all linear transformations, l-CKA is invariant to orthogonal projection and isotropic scaling, which suffices for our purposes. We base our analysis on the following assumption: the extent of change in transformers' representation space topology (reflected by l-CKA), is proportional to the novelty of knowledge injected in fine-tuning. Put differently, injection of new (i.e., missing) knowledge should substantially change the topology of the space (low l-CKA score).

Figure 3 shows the heatmap of l-CKA scores for pairs of BERT and RoBERTa variants, for layers L8-L12.[12] Comparing B-P and B-N reveals that IPT changes the topology of BERT's higher layers roughly as much as NLI fine-tuning does, implying that both the English UD treebank (EN -EWT) and Multi-NLI data contain a non-negligible amount of novel knowledge for BERT. However, the direct N-P comparison shows that IPT and NLI enrich BERT (also RoBERTa) with different type of knowledge, i.e., they change the representation spaces of its layers in different ways. This suggests that the transformers cannot acquire the missing knowledge needed for NLI from IPT (i.e., from EN -EWT), and explains why IPT is not effective for NLI.

IPT (comparison B-P) injects more new information than ILMT (comparison B-M), and this is more pronounced for BERT than for RoBERTa. IPT and ILMT change RoBERTa's parameters much less than BERT's (see B-M and B-P l-CKA scores for L11/L12), which we interpret as additional evidence, besides RoBERTa consistently outscoring BERT, that RoBERTa encodes richer language representations, due to its larger-scale and longer training. It also agrees with suggestions that BERT is "undertrained" for its capacity (Liu et al., 2019b).

Very high B-P (and B-AP) l-CKA scores in lower layers suggest that the explicit syntactic knowledge from human-curated treebanks is redundant w.r.t. the structural language knowledge transformers obtain through LM pretraining. This is consistent with concurrent observations (Chi et al., 2020; Kul-

[10] IPT itself only consumes train and development portions of UD treebanks. We can thus safely use sentences from test portions in this analysis, without risking information leakage.

[11] $\mathbf{X}_1$ and $\mathbf{X}_2$ must first be column-wise mean-centered.

[12] Most l-CKA scores in layers L1-L7 are very high (≥ 0.9) and provide little insight. See the Supplementary Material.

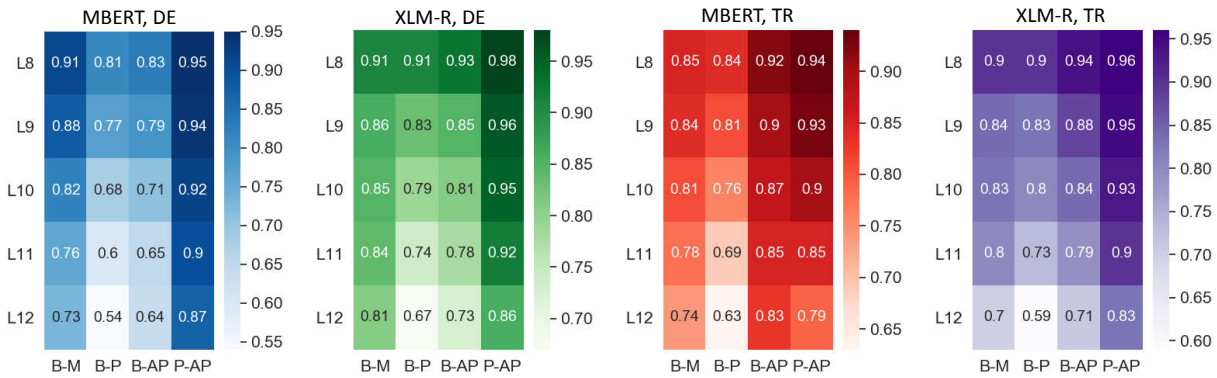

Figure 4: Analysis of topological similarity (l-CKA) for variants of mBERT and XLM-R before and after IPT and ILMT (B, M, P, AP) in zero-shot transfer experiments. Results shown for intermediate parsing on DE and TR data.

mizev et al., 2020) showing (some) correspondence between structural knowledge of (m)BERT and UD syntax. Finally, we observe highest l-CKA scores in the P-AP column, suggesting that Standard and Adapter IPT inject roughly the same syntactic information, despite different fine-tuning mechanisms.

Figure 4 illustrates the results of the same analysis for language transfer experiments, for DE and TR (scores for FR and ZH are in the Appendix). The effects of ILMT and IPT (B-M, B-P/B-AP) for DE and TR with mBERT and XLM-R resemble those for EN with BERT and RoBERTa: ILMT changes transformers less than IPT. The amount of new syntactic knowledge IPT injects is larger (l-CKA scores are lower) than for EN , especially for XLM-R (vs. RoBERTa for EN). We believe that it reflects the relative under-representation of the target language in the model's multilingual pretraining corpus (e.g., for TR): this leads to poorer representations of target language structure by mBERT and XLM-R compared to BERT's and RoBERTa's representation of EN structure. This gives us two seemingly conflicting empirical findings: (a) IPT appears to inject a fair amount of target-language UD syntax, but (b) this translates to (mostly) insignificant and inconsistent gains in language transfer in LU tasks (especially so for XLM-R, cf. Table 3). A plausible reconciling hypothesis is that there is a substantial mismatch between the type of structural information we obtain through supervised (UD) parsing and the type of structural knowledge beneficial for LU tasks. If true, this hypothesis would render supervised parsing rather unavailing for high-level language understanding, at least in the context of LM-pretrained transformers, the current state of the art in NLP. This warrants further investigation, and we hope that our work will inspire further discussion and additional studies.

6 Conclusion

We thoroughly examined the effects of leveraging formalized syntactic structures (UD) in state-of-the-art neural language models (e.g., RoBERTa, XLM-R) for downstream language understanding (LU) tasks, both in monolingual and language transfer settings. The key results, obtained through intermediate parsing training (IPT) based on a state-of-the-art-level dependency parser, indicate that explicit syntax, at least in our extensive experiments, provides negligible impact on LU tasks.

Besides offering extensive empirical evidence of the mismatch between explicit syntax and improved LU performance with state-of-the-art transformers, this study sheds new light on some fundamental questions such as the one in the title. Similar to word embeddings (Mikolov et al., 2013) removing sparse lexical features from the NLP horizon, will transformers make supervised parsing obsolete for LU applications or not? More dramatically, in the words of Rens Bod (2007): "Is the end of supervised parsing in sight" for semantic LU tasks?[13]

Acknowledgments

We thank the anonymous reviewers (especially R2!) for the exceptionally meaningful and helpful comments. Goran Glavaš is supported by the Baden Württemberg Stiftung (Eliteprogramm, AGREE grant). The work of Ivan Vulić is supported by the ERC Consolidator Grant LEXICAL: Lexical Acquisition Across Languages (no. 648909).

[13]The answer is 'Probably no': formalized syntactic structures will still be an important source of inductive bias, especially in setups without sufficient text data for large-scale pretraining; our experiments, however, validate that state-of-the-art transformer models can implicitly capture that inductive bias in high-resource setups and for major languages.

References

Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proc. of EMNLP*, pages 1957–1967.

Rens Bod. 2007. Is the end of supervised parsing in sight? In *Proc. of ACL*, pages 400–407.

Eugene Charniak. 1996. Tree-bank grammars. In *Proc. of NCAI*, pages 1031–1036.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proc. of EMNLP*, pages 740–750.

Jianpeng Cheng and Dimitri Kartsaklis. 2015. Syntax-aware multi-sense word embeddings for deep compositional models of meaning. In *Proc. of EMNLP*, pages 1531–1542.

Ethan A Chi, John Hewitt, and Christopher D Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proc. of ACL*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proc. of ICLR*.

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 *Proc. of ACL*, pages 8440–8451.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proc. of EMNLP*, pages 2475–2485.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. of LREC*, pages 449–454.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*, pages 4171–4186.

Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proc. of ICLR*.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *TACL*, 8:34–48.

Kim Gerdes, Bruno Guillaume, Sylvain Kahane, and Guy Perrier. 2018. Sud or surface-syntactic universal dependencies: An annotation scheme near-isomorphic to UD. In *Proc. of UDW*, pages 66–74.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (GELUs). *CoRR*, abs/1606.08415.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proc. of NAACL-HLT*, pages 4129–4138.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the penn treebank. *Computational Linguistics*, 33(3):355–396.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proc. of ICML*, pages 2790–2799.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. In *Proc. of ICML*.

Jungo Kasai, Dan Friedman, Robert Frank, Dragomir Radev, and Owen Rambow. 2019. Syntax-aware neural semantic role labeling with supertags. In *Proc. of NAACL-HLT*, pages 701–709.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proc. of ICLR*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *TACL*, 4:313–327.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proc. of ACL*, pages 423–430.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proc. of EMNLP*, pages 2779–2795.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *Proc. of ICML*, pages 3519–3529.

Artur Kulmizev, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre. 2020. Do neural language models show preferences for syntactic formalisms? In *Proc. of ACL*, pages 4077–4091.

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.

Adhiguna Kuncoro, Lingpeng Kong, Daniel Fried, Dani Yogatama, Laura Rimell, Chris Dyer, and Phil

Blunsom. 2020. Syntactic structure distillation pretraining for bidirectional encoders. *Transactions of the Association for Computational Linguistics*, 8:776–794.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proc. of ACL*, pages 302–308.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside bert's linguistic knowledge. In *Proc. of BlackboxNLP*, pages 241–253.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proc. of NAACL-HLT*, pages 1073–1094.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv:1907.11692*.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2020. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *57th Annual Meeting of the Association for Computational Linguistics, ACL 2019*, pages 3428–3448. Association for Computational Linguistics (ACL).

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proc. of NeurIPS*, pages 3111–3119.

Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proc. of LREC*, pages 1659–1666.

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. *arXiv preprint arXiv:2004.10643*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL-HLT*, pages 2227–2237.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting Transformers. In *Proc. of EMNLP: System Demonstrations*.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Edoardo Maria Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. XCOPA: A multilingual dataset for causal commonsense reasoning. In *Proc. of EMNLP*.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R Bowman. 2020. Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? In *Proc. of ACL*, pages 5231–5247.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *Proc. of AAAI SSS*.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proc. of EMNLP-IJCNLP*, pages 4463–4473.

Swabha Swayamdipta, Matthew Peters, Brendan Roof, Chris Dyer, and Noah A Smith. 2019. Shallow syntax in deep water. *arXiv preprint arXiv:1908.11047*.

Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. The Penn treebank: An overview. In *Treebanks*, pages 5–22. Springer.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *Proc. of ICLR*.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly universal dependency parsing. In *Proc. of EMNLP*.

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 *Proc. of NeurIPS*, pages 5998–6008.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proc. of EMNLP-IJCNLP*, pages 2153–2162.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, et al. 2019. Can you tell me how to get past Sesame street? sentence-level pretraining beyond language modeling. In *Proc. of ACL*, pages 4465–4476.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proc. of BlacboxNLP*, pages 353–355.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proc. of NAACL-HLT*, pages 1112–1122.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proc. of EMNLP-IJCNLP*, pages 3678–3683.

Meishan Zhang, Zhenghua Li, Guohong Fu, and Min Zhang. 2019a. Syntax-enhanced neural machine translation with syntax-aware word representations. In *Proc. of NAACL-HLT*, pages 1151–1161.

Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. 2017. Dependency parsing as head selection. In *Proc. of EACL*, pages 665–676.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. Paws: Paraphrase adversaries from word scrambling. In *Proc. of NAACL-HLT*, pages 1298–1308.

Wei Zhao, Goran Glavaš, Maxime Peyrard, Yang Gao, Robert West, and Steffen Eger. 2020. On the limitations of cross-lingual encoders as exposed by reference-free machine translation evaluation. In *Proc. of ACL*, pages 1656–1671.

A Reproducibility

We first provide details on where to obtain datasets and code used in this work.

A.1 Datasets

Table 5 lists the sizes (in number of sentences) of Universal Dependencies treebanks that we use for our intermediate parsing training and evaluation of our biaffine dependency parsers. The UD treebanks v.2.5, which we used in this work, are available at: http://hdl.handle.net/11234/1-3105. In Table 6 we provide links to language understanding datasets used in our study.

A.2 Code and Dependencies

We make our code available at: https://github.com/codogogo/parse_stilt. Our code is built on top of the HuggingFace Transformers framework: https://github.com/huggingface/transformers (v. 2.7). Table 4 details the LM-pretrained transformer models from this framework which we exploited in this work. Besides the Transformers library, our code only relies on standard Python's scientific computing libraries (e.g., `numpy`).

B ZH Tokenization: XLM-R vs. GSD

A word-level token from the parse tree normally corresponds to one or more transformer's subword tokens: we thus average subword vectors to obtain word vectors for biaffine parsing. For XLM-R and the ZH GSD treebank, however, a single XLM-R's subword token often corresponds to two treebank tokens. E.g., the sequence "只是二選一做決擇" with treebank tokenization ['只', '是', '二', '選', '一', '做', '決擇'] is tokenized as ['只是', '二', '選', '一', '做', '決', '擇'] by XLM-R. Two treebank tokens, '只' and '是', are captured with a single XLM-R "subword" token, '只是'. To ensure that each XLM-R subword token corresponds to exactly one treebank token, we inject spaces between treebank tokens before XLM-R tokenization: we then obtain the subword tokenization ['只', '是', '二', '選', '一', '做', '決, '擇']. However, this is suboptimal for XLM-R: its representations of tokens '只' and '是' are probably less reliable than that of the '只是' token. We believe this is why mBERT (without tokenization mismatches for ZH) outperforms XLM-R in ZH parsing.

C Complete Topology Analysis Results

Finally, we show the complete results (for all layers, all transformers, and all languages covered in our experiments) of our topological analysis of transformers' representations before and after different fine-tuning steps. Figure 5 shows the analysis results for monolingual EN transformers, BERT and RoBERTa. Figure 6 and Figure 7 show the results for multilingual transformers, mBERT and XLM-R, respectively, for all four target languages included in our experiments: DE , FR , TR , and ZH .

Name	Lang	Vocab	Params	URL
BERT	EN	29K	110M	https://huggingface.co/bert-base-cased
RoBERTa	EN	50K	110M	https://huggingface.co/roberta-base
mBERT	Multiling.	119K	125M	https://huggingface.co/bert-base-multilingual-cased
XLM-R	Multiling.	250K	125M	https://huggingface.co/xlm-roberta-base

Table 4: LM-pretrained transformer models used in our study.

Lang	Treebank	Train	Dev	Test
EN	EWT	12,538	2,002	2,077
DE	GSD	13,810	799	977
FR	GSD	14,440	1,475	416
TR	IMST	3,664	988	983
ZH	GSD	3,996	500	500

Table 5: Universal Dependencies treebanks used in our study. We display sizes of train, development, and test portions in terms of number of sentences.

Task	Dataset	URL
Natural Language Inference	Multi-NLI	https://cims.nyu.edu/~sbowman/multinli
Natural Language Inference	XNLI	https://github.com/facebookresearch/XNLI
Paraphrase identification	PAWS(-X)	https://github.com/google-research-datasets/paws
Commonsense social reasoning	SIQA	https://maartensap.github.io/social-iqa
Commonsense causal reasoning	COPA	https://people.ict.usc.edu/~gordon/copa.html
Commonsense causal reasoning	XCOPA	https://github.com/cambridgeltl/xcopa

Table 6: Links to downstream language understanding datasets used in our work.

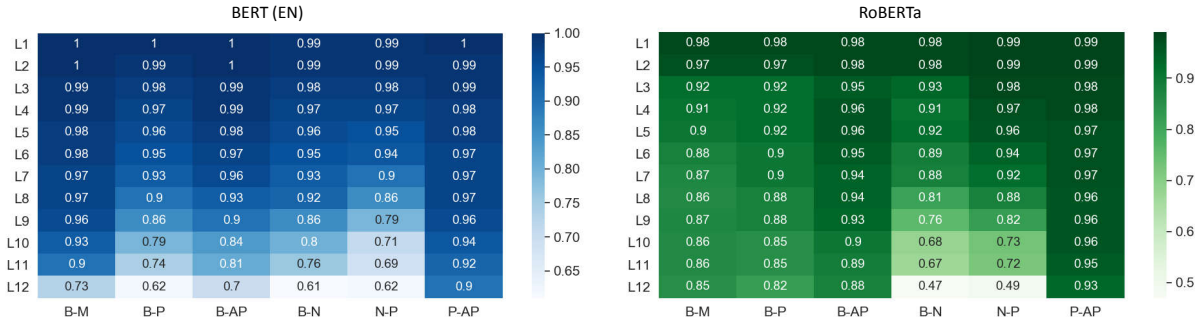

Figure 5: Full results of the topological similarity analysis (l-CKA) for pairs of BERT and RoBERTa variants, before and after different fine-tuning steps (B, M, P, AP, and N). **Rows**: transformer layers; **Columns**: pairs of transformer variants in comparison.

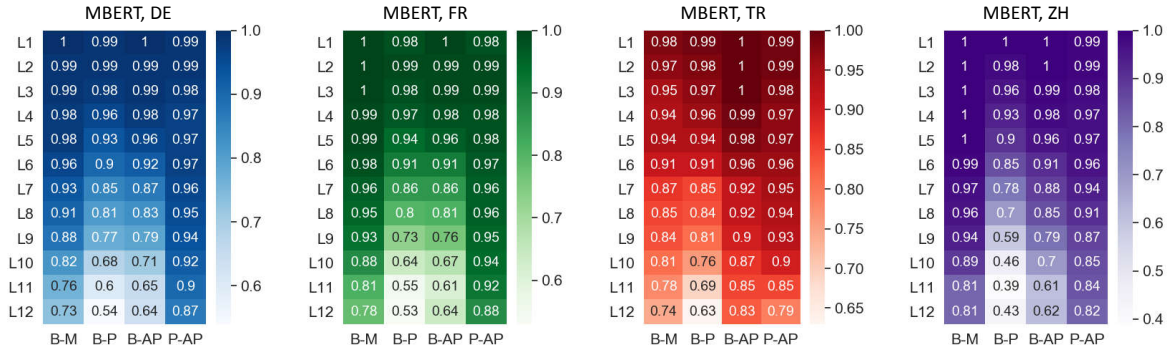

Figure 6: Full results of the topological similarity analysis (l-CKA) for variants of mBERT before and after IPT and ILMT (B, M, P, AP) for the following target languages (left to right): DE , FR , TR , and ZH .

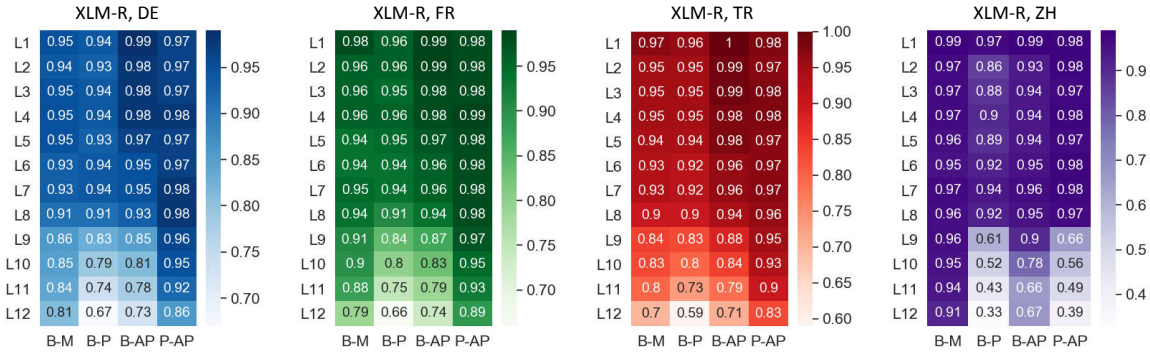

Figure 7: Full results of the topological similarity analysis (l-CKA) for variants of XLM-R before and after IPT and ILMT (B, M, P, AP) for the following target languages (left to right): DE , FR , TR , and ZH .