Multilingual Nonce Dependency Treebanks: Understanding how Language Models *Represent* and *Process* Syntactic Structure

David Arps[1], **Laura Kallmeyer**[1], **Younes Samih**[2], **Hassan Sajjad**[3]
Heinrich-Heine-Universität Düsseldorf[1], IBM Research[2], Dalhousie University[3]
{david.arps,laura.kallmeyer}@hhu.de, younes.samih@ibm.com, hsajjad@dal.ca

Abstract

We introduce SPUD (Semantically Perturbed Universal Dependencies), a framework for creating nonce treebanks for the multilingual Universal Dependencies (UD) corpora. SPUD data satisfies syntactic argument structure, provides syntactic annotations, and ensures grammaticality via language-specific rules. We create nonce data in Arabic, English, French, German, and Russian, and demonstrate two use cases of SPUD treebanks. First, we investigate the effect of nonce data on word co-occurrence statistics, as measured by perplexity scores of autoregressive (ALM) and masked language models (MLM). We find that ALM scores are significantly more affected by nonce data than MLM scores. Second, we show how nonce data affects the performance of syntactic dependency probes. We replicate the findings of Müller-Eberstein et al. (2022) on nonce test data and show that the performance declines on both MLMs and ALMs wrt. original test data. However, a majority of the performance is kept, suggesting that the probe indeed learns syntax independently from semantics.[1]

1 Introduction

An ample amount of work in the last years has focused on making explicit the linguistic information encoded in language models (LMs). Frequently, the overarching question is to which extent the behavioral and representational properties of self-supervised LMs are cognitively plausible and assumed by linguistic theory (surveys: Linzen and Baroni, 2021; Mahowald et al., 2023; Chang and Bergen, 2023). A subset of this work aimed at understanding LMs ability to learn syntax (survey: Kulmizev and Nivre, 2022, examples: Hewitt and Manning, 2019; Manning et al., 2020; Tenney et al., 2019; Newman et al., 2021). A common approach is to rely on the performance of a probing classifier (Hupkes et al., 2018; Kunz and Kuhlmann, 2020, 2021) in predicting syntactic relations. However, this method has been criticized for various reasons (Belinkov, 2022), including that predicting a linguistic property from LM representations does not imply that the LM uses that property in its predictions (Lyu et al., 2024). Maudslay and Cotterell (2021) highlighted that results of syntactic probes can be misleading due to semantic information in the representation. In other words, a high probing accuracy may be fully or partially attributed to the use of present semantic knowledge.

The theoretical linguistic literature has discussed the separation of syntactic grammaticality and semantic information for decades, starting with the sentence *Colorless green ideas sleep furiously* in Chomsky (1957). A number of studies have used systematically perturbed data to probe LM representations and predictions on such grammatical but nonsensical sentences (Gulordava et al., 2018; Maudslay and Cotterell, 2021; Arps et al., 2022). However, none of the works ensures that the nonce data is grammatical, e.g. in their datasets, the valency of predicates is not controlled. Furthermore, only Gulordava et al. (2018) base their conclusions on languages beyond English.

In this paper, we aim to standardize the efforts in this direction by proposing a framework to create nonce data that satisfies syntactic argument structure, provides syntactic annotations, and ensures grammaticality via language-specific rules. The framework relies on Universal Dependencies (UD, de Marneffe et al., 2021) treebanks. For each sentence in the treebank, it replaces content words with other content words that are known to appear in the same syntactic context. This results in nonce sentences that preserve the syntactic structure of the original sentences (see Fig. 1). To ensure syntactic grammaticality, we define three types of language-specific rules and constraints that are applied during the process, which are: (i) the POS tag of the words

[1] Code at https://github.com/davidarps/spud

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 7822–7844
June 16-21, 2024 ©2024 Association for Computational Linguistics

to be replaced, (ii) constraints on word order and dependency relations of replaced words to ensure syntactic grammaticality, and (iii) word-level rules to ensure that morphosyntactic constraints are met. We refer to this algorithm and the resulting UD data as *Semantically Perturbed UD* (SPUD).

We create SPUD data for five widely spoken languages; Arabic, English, French, German, and Russian. We show via a human evlation that SPUD is preferrable in terms of grammaticality to the kind of nonce data that has been previously used in the literature to tackle similar research questions. We show the effectiveness of SPUD on two tasks to assess the robustness of autoregressive LMs (ALMs) and masked LMs (MLMs) to semantic perturbations. First, we study the effect of nonce data on LM scoring functions (perplexity and its adaptations for MLMs). Second, we investigate the robustness of syntactic dependency probes to semantic irregularities, and disentangle the effect of lexical semantic features on the findings of previous work. The contributions and main findings of our work are as follows:

- We introduce SPUD, a framework for creating nonce treebanks for UD corpora that ensures syntactic grammaticality, and provide nonce treebanks for 5 languages (Arabic, English, French, German, Russian).
- We show the effectiveness of the proposed data log-likelihood scoring of different language model architectures, and on the performance of syntactic dependency probes.
- We show that ALM perplexity is significantly more affected by nonce data than two formulations of MLM pseudo-perplexity, and that for MLMs, the availability of subword information affects pseudo-perplexity scores on both natural and SPUD data.
- In structural probing for dependency trees, we show that ALM performance decreases more than MLM performance on SPUD, and that this performance drop is more pronounced for edge attachment than relation labeling.

The paper is structured as follows. In the next section, we discuss related work. In Sec. 3, we describe the framework for creating nonce UD treebanks. Sec. 4 describes the experiment on scoring nonce data with perplexity. In Sec. 5, we describe the structural probing experiments. In Sec. 6, we discuss the results and conclude.

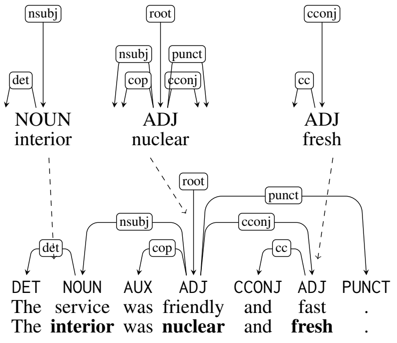

nsubj
root
cconj
det
nsubj
punct
cop
cconj
cc
NOUN
interior
ADJ
nuclear
ADJ
fresh
root
punct
nsubj
cconj
det
cop
cc
DET NOUN AUX ADJ CCONJ ADJ PUNCT
The service was friendly and fast .
The **interior** was **nuclear** and **fresh** .

Figure 1: SPUD data creation

2 Related Work

Automatically modified Dependency Trees The idea to replace words and subtrees based on dependency structure has been used for data augmentation in various areas of NLP, in particular for low-resource dependency parsing. Dehouck and Gómez-Rodríguez (2020) proposed an idea similar to ours, with the differences that (i) they swap subtrees instead of content words resulting in generated sentences with altered syntactic structure, and (ii) their constraints on possible replacements are language-agnostic. Şahin and Steedman (2018) cut subtrees based on their dependency relation, and modified treebanks by changing the order of annotated dependency subtrees. Vania et al. (2019) demonstrated the effectiveness of Şahin and Steedman (2018)'s algorithm as well as nonce treebanks for low-resource dependency parsing. However, their method lacked language-specific constraints and did not rely on dependency edges to dependents. Nagy et al. (2023) apply a similar method for low-resource machine translation. Wang and Eisner (2016) permuted dependents within a single tree to generate synthetic treebanks with lexical contents from one language and word order properties of another language.

Structural Probing DepProbe (Müller-Eberstein et al., 2022) decodes labeled and directed dependency trees from LM representations. With DepProbe, Müller-Eberstein et al. (2022) probed mBERT (Devlin et al., 2019) on 13 languages, and showed that the probe performance is predictive of parsing performance after finetuning the LM. Prior to that, similar probes have been applied to unlabeled (Kulmizev et al., 2020) as well as unlabeled and undirected dependency trees (Hewitt and Manning, 2019; Chi et al., 2020). Eisape et al. (2022) used Hewitt and Manning (2019)'s method to probe GPT-2 (Radford et al., 2019).

Syntactic and Semantic Information in LMs Gulordava et al. (2018) performed targeted syntactic evaluation (TSE; Marvin and Linzen, 2018) on ALMs in four languages and investigated the effect of nonce data that considers POS tags and morphological features for replacements. Lasri et al. (2022a) investigated the effect of nonce data for TSE in BERT and found that BERT correctly predicts number agreement for nonce sentences only in simple syntactic templates. Maudslay and Cotterell (2021) investigated the effect of *Jabberwocky words* (such as *provicated*) on syntactic probes. Ravfogel et al. (2020) found a transformation of LM representations that highlights structural properties. Arps et al. (2022) used a similar algorithm as ours to create nonce versions of the English PTB (Marcus et al., 1993), and probed the syntactic information in hidden representations of four MLMs. The main difference to SPUD is that they do not restrict replacements to content words, and do not apply language-specific processing steps. Therefore their nonce data is more likely to be ungrammatical. Sinha et al. (2021) tested to which extent MLMs rely on word order vs. higher-order cooccurence statistics. They found that word order is not needed to achieve high performance on many NLP tasks. Papadimitriou et al. (2022) probed the relevance of word order and semantic prototypicality for classifying grammatical roles with BERT. Lasri et al. (2022b) compared BERT and human judgments on the subject-verb agreement task for nonce data and found that, while the error patterns of both are similar, BERT has a generally higher performance drop for nonce data than humans. Kauf et al. (2023) explored the interaction of syntactic and semantic information with similarity between LMs and human fMRI signals, and show that lexical semantic content - not syntactic structure - is the main driver of similarity in both LM and human representations.

Scoring Functions for LMs The likelihood of sentences assigned by an LM is often used to investigate the model's preference for grammatical sentences (Kulmizev and Nivre, 2022; Warstadt et al., 2020). Perplexity (PPL) is a common metric to score the likelihood of a sentence with ALMs. For MLMs, estimating the likelihood of a given sentence in a fashion that is useful for applications is not as trivial. Salazar et al. (2020) formalized pseudo-perplexity ($PPPL$) as a scoring function for MLMs: Each token in a sentence is masked in turn, and $PPPL$ is computed as a function of the MLM's probability of the masked tokens. Kauf and Ivanova (2023) showed that $PPPL$ systematically inflates scores for multi-token words, and proposed $PPPL_{l2r}$, a subword-aware alternative that is more robust to this effect. Miaschi et al. (2021) trained classifiers on different linguistic features to predict the PPL of GPT-2 and $PPPL$ of BERT. They find that GPT-2 scores are generally better predictable by their set of features than BERT scores, and that lexical features are more important for GPT-2 than for BERT.

3 Nonce Treebanks for Five Languages

This section presents SPUD (*Semantically Perturbed Universal Dependencies*), our framework for nonce data creation. The input is a UD treebank, and the output is a treebank with the same syntactic structure as the input but nonce semantic content. In principle, the same algorithm is applied in any language, but each language requires a set of language-specific pre- and post-processing steps (Sec. 3.1). In this work, we create nonce data for five languages and treebanks: Arabic (PADT, Hajic et al., 2009), German (HDT, Borges Völker et al., 2019), English (EWT, Silveira et al., 2014), French (GSD, Guillaume et al., 2019), and Russian (SynTagRus, Droganova et al., 2018). Fig. 1 presents one example. More examples and information on the resources are in App. A. To apply the framework to a new language, an annotated UD treebank and access to a native speaker in the target language are required. Along with this paper, we publish a tutorial for creating SPUD treebanks in other languages.

3.1 Generating Nonce Data

Language-independent algorithm The procedure consists of iteratively replacing content words with other words that appear in the same syntactic context at another point in the treebank. We consider words with POS tags ADJ, ADV, NOUN, PROPN and VERB as content words. The syntactic context of a token t is defined as (i) the UPOS tag of t, (ii) the dependency relation of t to its head, and (iii) the dependency relations of t to its dependents. This syntactic context is collected for every lemma. Then, the nonce trees are created by replacing content words with other words where the lemma appeared in the same syntactic context. The morphological features of t, as annotated in

UD, are considered to determine the right form for a replacement. For this step, the morphological databases UDLexicon (Sagot, 2018) and Wiktextract (Ylonen, 2022) are used. Fig. 1 presents an example (without morphosyntactic features).

Language-specific modifications are necessary to ensure that the data meets the criterion of being morphosyntactically correct. For instance, if the first sound of the word following an English indefinite article changes from a vowel to a consonant or vice versa, the article is adjusted with the help of a phonological dictionary. E.g. when replacing *apple* in *an apple* with *bicyle*, the result is *a bicycle* and not **an bicycle*. We refer to App. A for details on the language-specific modifications.

Quality of the generated data To determine the sufficiency of the language specific rules, we asked linguistically trained native speakers of the respective languages to provide feedback on how well the nonce sentences match the desired criteria. The annotators received samples of at least 100 sentence pairs. Over 2-3 iterations, annotators pointed out problems in the generated data, which we then addressed by modifying the language-specific rules.

Human evaluation We conduct a human evaluation to estimate the benefits of using SPUD over the algorithm presented by Gulordava et al. (2018), which does not incorporate information about syntactic dependencies or language-specific rules. For this evaluation, human annotators were presented with sentence triplets, each consisting of an original sentence from the treebank, a nonce version of that sentence generated by SPUD, and a nonce version generated by the algorithm of Gulordava et al. (2018). The two nonce sentences were presented in random order, without indication of their source. Annotators rated for each sentence whether it was grammatical, and which of the two nonce sentences was syntactically closer to the original. One annotator rated 30 French sentence triplets, and two annotators rated 39 English and 153 German sentence triplets. All triplets where selected randomly from the corresponding treebanks. Annotators had the option to indicate that they were unsure, or that the two nonce sentences were equally close to the original. We find that SPUD is rated grammatical more frequently than the algorithm of Gulordava et al. (2018) in all three languages, and by all annotators (Tab. 1). Concretely, up to 87% of the SPUD sentences are rated grammatical, compared to only up to 38% of the sentences generated by the algorithm from Gulordava et al. (2018). Inter-annotator agreement on grammaticality judgments is moderate, with Cohen's Kappa of .51 (en) and 0.58 (de). SPUD is also preferred in the majority of cases, with some "equal" ratings (Tab. 2). We conclude that, while the scale of this experiment is limited, the results suggest that the quality of the sentences generated by SPUD is higher than that of the algorithm of Gulordava et al. (2018), which has been used previously to interpret the behavior of LMs.

4 Scoring SPUD with ALMs and MLMs

The SPUD treebanks are designed to be grammatical but highly improbable. In this section, we investigate how this property is reflected in the predictions of different LM architectures. On the one hand, this serves as a sanity check for SPUD resources: Do we in fact perturb co-occurrence statistics of words as intended? On the other hand, at a higher level, it investigates whether SPUD data makes the models perplexed and how different LM architectures, and uni- vs. bidirectional context, influence the effect that syntactic and semantic structure has on model predictions. Concretely, we answer the following questions: To what extent is SPUD data harder to predict than original data? Are ALMs and MLMs affected by SPUD data in different ways? Are MLM scoring functions affected differently by nonce data?

4.1 Scoring Functions for LMs

App. E.1 shows examples for all scoring functions.

ALMs: Perplexity (PPL) of a sentence $x = (x_1, \ldots, x_n)$ is commonly defined as the exponentiated average of the negative sum of log probabilities for all tokens x_i. The lower the perplexity, the higher the probability of x for the model.

$$PPL(x) = exp(-\frac{1}{n}\sum_{i=0}^{n} \log p(x_i|x_{<i})) \qquad (1)$$

MLMs: Pseudo-Perplexity ($PPPL$) is designed to capture the likelihood that an MLM assigns to a sequence (Salazar et al., 2020). It is calculated by processing the input n times, masking each token x_i exactly once. $PPPL(x)$ is defined via the sum of log probabilities that the LM assigns

		SPUD			Gulordava et al. (2018)			
	Sentences	yes	no	unsure	yes	no	unsure	\kappa
de	153	72.5	27.5	0	9.5	90.5	0	0.58
en	39	78.2	19.2	2.6	35.9	60.3	3.8	0.51
fr	30	86.7	13.3	0	0	100	0	

Table 1: Results for Grammaticality judgments. For German and English, the mean between the two annotators is presented. Grammaticality judgments were also made for the original sentences, which were almost always rated grammatical. They are not included in the figure and in the inter-annotator agreement calculations.

	Equal	SPUD	Gulordava et al. (2018)	unsure	κ
de	13.7	81.0	4.2	1.0	0.17
en	20.5	70.5	1.3	7.7	0.09
fr	13.3	86.7	0.0	0.0	

Table 2: Results for Preference judgments. For German and English, the mean between the two annotators is presented. While Cohen's κ is low, per item agreement is higher (.56 for en, .73 for de).

to the masked x_i:

$$PPPL(x) = exp(-\frac{1}{n}\sum_{i=0}^{n} \log p(x_i|x_{\setminus i})) \quad (2)$$

MLMs: $PPPL$ with subword generation Kauf and Ivanova (2023) proposed $PPPL_{l2r}$, a variant of $PPPL$ that is more aligned with PPL for ALMs. The idea is to consider the groupings of tokens to words, and predict subword tokens without conditioning on succeeding tokens. Let $\omega_r(x_i)$ denote the tokens that are in the same word as x_i, including and succeeding x_i. For example, when tokenizing *accordeon* as *accord, ##eon*, it holds that $\omega_r(\text{accord}) = \{\text{accord, \#\#eon}\}$. Then, tokens $\omega_r(x_i)$ from the right context within the same word are masked out when predicting the token x_i:

$$PPPL_{l2r}(x) = exp(-\frac{1}{n}\sum_{i=1}^{n} \log p(x_i|x_{\setminus \omega_r(x_i)})) \quad (3)$$

Evaluating Scoring Functions on SPUD When comparing scores across sentences, the variance is very high: Some sentences are far more likely than others, irrespective of the language or whether they are nonce. Thus, we use the ratio between sentence-level scores of a pair of original and corresponding SPUD sentence (s_{orig}, s_{nonce}). For a scoring function $f \in \{PPL, PPPL, PPPL_{l2r}\}$, we define the ratio as $r_f(s_{orig}, s_{nonce}) = \frac{f(s_{nonce})}{f(s_{orig})}$ and then investigate the distribution of r_f for all sentence pairs in a corpus.

4.2 Hypotheses

Hypothesis 1: *LMs assign a higher score to* SPUD *data than to original data* - the content words are chosen at random, and therefore should be much harder to predict than the original words.

Hypothesis 2: *ALMs and MLMs are impacted by* SPUD *data in different ways.* Bidirectional context determines the syntactic properties of a predicted token to a larger degree than unidirectional context. Therefore, the space of probable predictions is smaller for MLMs than for ALMs, and we expect that the impact of SPUD data on ALM perplexity should be higher than on MLM pseudo-perplexity.

Hypothesis 3: *For* SPUD*,* $r_{PPPL_{l2r}}$ *are higher than* r_{PPPL} because nonce words are hard to predict based on context from surrounding words, but easy to predict based on context from the same word.

4.3 Experimental Setup

We adapt Kauf and Ivanova (2023)'s implementation to retrieve token-level scores for all sentences in the SPUD and original data. We compare results for mBERT (Devlin et al., 2019) as an MLM and mGPT (Shliazhko et al., 2024) as an ALM. We report the scoring results of only one instance of SPUD data per sentence. We argue that this is sufficient to draw conclusions about the general behavior of the models, since we conduct the experiment on a large number of sentences, and different languages.

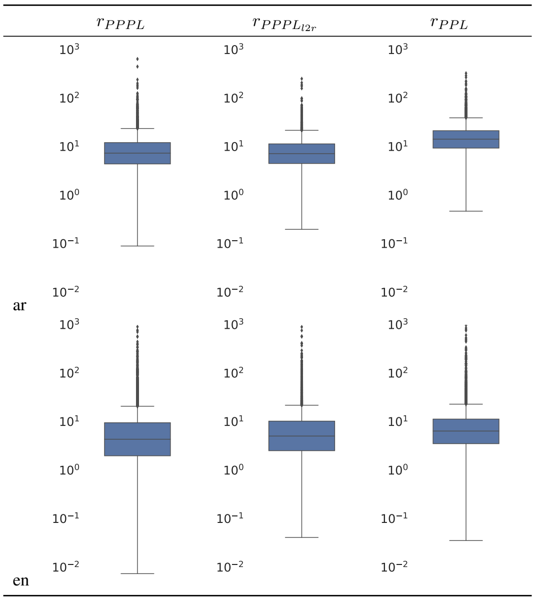


Figure 2: Intrinsic evaluation results for English and Arabic. Plots for other languages are in App. E.2.

4.4 Results

Is SPUD data harder to predict than original data? To test Hypothesis 1, we check the distribution of score ratios r_f for all scoring functions f. The results for mBERT and mGPT are shown in Fig. 2 and show that for all settings, the first quartile is larger than 1, i.e. the score of SPUD sentences is higher than that of original sentences for the vast majority of the data. This means that SPUD data receives lower likelihood than original data and is therefore harder to predict, as expected.

Are scoring functions affected differently by nonce data? To answer this question, we compare the distributions of score ratios for all scoring functions, as displayed in Fig. 2. This question is answered in two parts, and both parts hold irrespective of whether outliers with $r_f > 250$ are taken into account or not. First, we test Hypothesis 2 by comparing the distributions of score ratios for ALMs and MLMs. Here, the picture is clear: r_{PPL} are significantly higher than both $r_{PPPL_{l2r}}$ and r_{PPPL}, measured with a Wilcoxon signed-rank test. This result, and all following results, are significant at $p \ll 0.001$. Second, we test Hypothesis 3 by comparing the MLM scoring functions $r_{PPPL_{l2r}}$ and r_{PPPL}. Here, the picture is less clear: All ratios are significantly different from each other according to the same test, but $f_{PPPL_{l2r}}$ has slightly lower quartiles than r_{PPPL} for Arabic (in all other languages, the opposite is true). This means that for all languages except Arabic, $PPPL$ is less affected by SPUD data than $PPPL_{l2r}$.

Differences across languages and models The same trends were tested for monolingual Arabic and English models. The results are presented in App. E.2. In all monolingual settings, SPUD data is harder to predict than original data (all first quartiles of ratio distributions are larger than 1). For differences between ratio predictions, the picture is more mixed: Regarding the difference between ALMs and MLMs, Arabic AraGPT2 (Antoun et al., 2021) shows higher ratio quartiles for PPL than the MLM scoring functions of AraBERT (Antoun et al., 2020). For English, the MLM scoring functions of RoBERTa (Liu et al., 2019) are significantly higher than PPL of GPT-2 (Radford et al., 2019). Regarding the difference between $PPPL$ and $PPPL_{l2r}$, the picture is the same as for multilingual models: For English, $PPPL_{l2r}$ is significantly higher than $PPPL$, and for Arabic, the opposite is true.

4.5 Interim Summary and Discussion

There are three main takeaways from this experiment. First, we find (as expected) that SPUD is harder to predict than original data in all settings. Second, the ALM perplexity and two MLM scoring functions respond differently to SPUD data: mBERT is affected less than mGPT, and within mBERT, $PPPL_{l2r}$ is affected more than $PPPL$ for all languages except Arabic (because of the availability of subword context). For monolingual models, we see a more mixed picture where for English, ALMs are affected less by SPUD data than MLMs, and for Arabic, the trend that $PPPL_{l2r}$ is affected more than $PPPL$ is consistent. These results suggest on one hand a special role of the Arabic language, and on the other hand point to the possibility that these differences are not systematic properties of model architectures and scoring functions, but rather depend on the training data and tokenization. However, as a general trend, we can hypothesize from the comparison between $PPPL$ and $PPPL_{l2r}$ that the inflated scores for multi-token words that Kauf and Ivanova (2023) found for $PPPL$ are carried over to SPUD data: If $PPPL$ increases less than $PPPL_{l2r}$ for nonce data, this means that the increase in $PPPL$ is cushioned by the high predictability of multi-token words in $PPPL$. We measure the lexical diversity of SPUD in App. D. For all languages, Type-Token-Ratio (TTR) decreases.

This means that all our findings hold even though SPUD is less lexically diverse than natural data.

5 Nonce Dependency Probing

In this section, we measure the efficacy of SPUD in decoding the syntactic knowledge learned in LM representations. Concretely, we train DepProbe structural probes (Müller-Eberstein et al., 2022) and compare the performance on the standard (original) UD test sets with performance on SPUD test sets. We choose DepProbe because none of its alternatives produces both directed and labeled dependency trees. We investigate to which extent DepProbe performance is influenced by lexical semantics rather than its desired task of predicting purely syntactic properties. With the assumption from previous work (Sec. 2) that a certain drop in performance on nonce test data is expected, we answer the following research questions:

RQ1: *Is MLM and ALM performance affected differently when tested on* SPUD *data?*

RQ2: *Are the predictions of dependencies between tokens (edges) and dependency relations (edge labels) affected differently by* SPUD *data?*

We answer these questions on all five languages, and present additional experiments to improve the robustness of our findings.

5.1 Probing Model

Müller-Eberstein et al. (2022) presented DepProbe, a lightweight decoder for directed and labeled dependency trees. The model consists of two learnt linear transformations of the LM representations. Both transform d_h-dimensional word representations into vectors that highlight a feature of the dependency tree. L is a sequence labeling classifier that predicts for each word the label of the incoming dependency edge. $B \in \mathbb{R}^{d_h \times b}$ projects the LM representation in a *syntactic subspace* with $b < d_h$, where word vector distances mimic the distance between words in the tree. The dependency tree is constructed in a top-down fashion from both components, selecting the word as root to which L assigns the highest probability of being the root. Details are provided in App. F.1.

5.2 Experimental setup

Metrics We report the same metrics as Müller-Eberstein et al. (2022), all of which are commonly used in the dependency parsing and probing literature. RelAcc is the percentage of tokens for which the relation label of the incoming edge is correctly predicted, UAS (unlabeled attachment score) is the percentage of tokens for which the head token is correctly predicted, and LAS (labeled attachment score) is the conjunction of RelAcc and UAS.

Hyperparameters All probe hyperparameters were taken from Müller-Eberstein et al. (2022) in order to replicate their results. For all models, the dimensionality of the syntactic subspace is $b = 128$. We take the hidden representations of layers 6 (7) as input to DepProbe's distance (relation) component for all twelve-layered models. Müller-Eberstein et al. (2022) trained probes on English data for all layers of mBERT and chose the best-performing layer for each component. We repeated this experiment to determine the best-performing layer for the 24-layered mGPT (App. F.6), and found that layer 12 performs best for both components.

Baselines We use three baselines to ensure that the syntactic information we probe for is (i) contextual in nature rather than dependent on word types, and (ii) learnt during LM pretraining rather than by the probe. To estimate how much information about the probing task is contained in (context-insensitive) token embeddings, we train DepProbe on mBERT's embedding layer. To estimate how much information about the probing task is acquired during pretraining of a LM, we probe the internal representations of two MLMs that share the architecture of mBERT but randomize either all parameters, or all parameters except the embedding matrix (Belinkov, 2022). All trained probes outperform the baselines by large margins (minimal difference of 15.9 LAS for mGPT on SPUD data). All baseline results are presented in App. F.3.

5.3 Results

Tables 3, 4 present the results. Δ shows the performance drop between original and SPUD test sets. In general, performance depends largely on treebank size. This trend is also visible in Müller-Eberstein et al. (2022), where Kendall's τ between LAS and treebank size in 13 languages is 0.50 (p = 0.017).

mBERT vs. mGPT (RQ1) For all languages and settings except Arabic RelAcc (both test sets), and Russian (all metrics, original data), mBERT outperforms mGPT in terms of absolute performance. In addition, Δ on SPUD is larger for mGPT than for mBERT in almost all settings. The difference in model architectures is especially relevant for predicting the syntactic structure of nonce data,

	RelAcc		UAS		LAS	
	orig	Δ	orig	Δ	orig	Δ
ar	82.8	5.5	63.1	7.3	55.5	8.3
de	92.7	1.7	84.4	3.4	80.2	4.0
en	87.1	2.3	74.2	3.8	67.9	4.1
fr	89.3	2.0	76.0	4.4	70.4	4.3
ru	88.2	1.5	75.4	3.2	69.2	3.1

Table 3: DepProbe results for mBERT. Δ shows the performance drop when nonce data is used.

	RelAcc		UAS		LAS	
	orig	Δ	orig	Δ	orig	Δ
ar	85.2	7.7	59.5	12.6	53.4	13.5
de	92.6	3.6	84.2	6.8	79.9	8.1
en	82.8	5.2	58.9	5.5	52.2	6.5
fr	87.5	2.8	69.6	6.2	64.0	6.7
ru	88.7	3.0	75.6	5.4	69.6	6.0

Table 4: DepProbe results for mGPT. Δ shows the performance drop when nonce data is used.

where context is a more important cue than for original data. The only exception to this is Russian, where mGPT outperforms mBERT in absolute performance on original data but not on SPUD. This is likely due to the distribution of languages in the pretraining data: mGPT's training data put an emphasis on Russian and regionally related languages (Shliazhko et al., 2024). The relatively low performance of both models on English, however, is most likely explained by the relatively small size of the English treebank.

Relation labeling vs. attachment (RQ2) DepProbe performance is measured in terms of the two components of the model: relation labeling (RelAcc) and attachment (UAS). While the absolute performance varies, for both models and all languages, Δ is larger for the attachment component than for the relation labeling component. mGPT shows a larger Δ than mBERT for both components when directly comparing the absolute performance drop per language. In the most extreme case, the Δ in UAS for Arabic is 7.3 points for mBERT and 12.6 points for mGPT. This points in the same direction as the performance of the embedding layer baseline: Relations are more easily predictable from lexical cues. Attachment, on the other hand, requires more contextual information, especially when the head is in the future context of the dependent. This finding is particularly insightful because it shows that probing for labeled and directed dependency trees highlights differences between ALMs and MLMs that are not captured when ignoring relations and directionality (e.g. Eisape et al., 2022).

	RelAcc		UAS		LAS	
	orig	Δ	orig	Δ	orig	Δ
roberta-base	85.9	2.5	70.9	3.6	64.0	4.0
gpt2	82.4	4.5	53.1	3.6	47.1	4.6

Table 5: DepProbe monolingual results for English

Monolingual results (RQ1,RQ2) We test the above hypotheses on two English 12-layered models: GPT-2 (Radford et al., 2019) and RoBERTa (Liu et al., 2019). The trends (Tab. 5) are similar to the multilingual models: RoBERTa outperforms GPT-2 on all metrics, and the performance drop is larger for GPT-2 than for RoBERTa (except for UAS, where it is the same). The only difference is that for GPT-2, the separate probe components show a behavior inverse to the multilingual models: RelAcc shows a larger performance drop on SPUD data (4.5) than UAS (3.6). However, this can be explained by the absolute performances of the GPT-2 probe, which are much higher for RelAcc than for UAS. This leaves less room for performance detriments on SPUD data at attachment than at relation labeling.

5.4 Additional experiments

To test the robustness of our results, we conduct several additional experiments. First, we test the effect of different random seeds for the probe's initial weights and the selection of nonce words (App. F.4). We find that performance is stable across random seeds. Concretely, the largest observed standard deviation across random seeds was 0.35 points UAS for English SPUD data, and most of the other settings show much lower standard deviations. Second, we test the effect of ignoring dependency relations in nonce data creation, i.e. selecting replacements based only on POS tags and language-specific rules (App. F.5). The results show that this kind of nonce data produces worse probing results than SPUD data, with some cross-lingual variation. Finally, we train DepProbe on all layers of mBERT and mGPT and compare the performance on English original and SPUD data (App. F.6). Most importantly, this confirms that all other experiments were run on the best-performing layers. Additionally, this provides insights into the layerwise dynamics of both models.

5.5 Interim Summary

The probing experiments show an overall performance drop on SPUD data for both MLMs and ALMs. In general, probes for MLMs perform better than probes for ALMs on both original and SPUD, and the performance drop on SPUD is higher for ALMs than for MLMs. This is likely due to the fact that ALM probe predictions are based on a representation with one-sided contextual information.[2] All models show a larger peformance drop for attachment than for relation labeling, and the embedding layer baseline shows a relatively high performance on relation labeling. This points in the direction that relation labeling information is more easily predictable from lexical cues than attachment information and that, on the contrary, MLMs are better at predicting attachment information for nonce data than ALMs. Finally, when viewed in isolation, many SPUD test sets show a high absolute performance, suggesting that probes indeed base their predictions on syntactic information.

6 Discussion and Conclusion

Dataset We presented *Semantically Perturbed Universal Dependencies* (SPUD), a framework for creating parallel nonsensical UD treebanks, and apply it to generate grammatical but nonsensical data in five languages. We present two use cases for SPUD, which contribute to the literature that relates LMs to the notions of grammaticality, acceptability and probability (Gulordava et al., 2018; Sprouse et al., 2018; Lau et al., 2017, 2020). Beyond that, SPUD has several possible applications such as data augmentation for low-resource parsing or syntactic similarity benchmarks.

Scoring SPUD with ALMs and MLMs In analyzing how token-level predictions are affected by SPUD, we find that SPUD data consistently shows higher scores (i.e., lower likelihood) than original data on all models. This shows that, as desired, the lexical co-occurrence patterns of content words in SPUD are perturbed. Of all scoring functions, the increase in ALM perplexity is highest. With our experiments, we contribute to the open question of how best to compute scores from MLMs by comparing $PPPL$ with subword-aware $PPPL_{l2r}$ (Kauf and Ivanova, 2023). While we are mostly able to support (Kauf and Ivanova, 2023)'s argument that $PPPL_{l2r}$ is empirically more similar to PPL than $PPPL$ is, we find that the response of ALMs and MLMs to SPUD data is significantly different. This indicates a more fundamental difference between how ALMs and MLMs score sentences: Bidirectional context in MLMs makes it easier to identify the syntactic properties of a predicted word, even in nonce sentences.

Probing ALMs and MLMs for SPUD trees We probe different models for labeled and directed dependency trees on SPUD using the DepProbe framework (Müller-Eberstein et al., 2022). We find that probe performance drops (compared to original test sets) for all languages and both model architectures when using SPUD test sets. However, a majority of the probe performance is maintained. The ALM mGPT has a lower overall performance than mBERT, as well as a higher performance drop on SPUD data. For both architectures, attachment contributes more to the performance drop than relation labeling does. These findings confirm the results in Maudslay and Cotterell (2021) and Arps et al. (2022). On the contrary, Eisape et al. (2022) found that GPT-2 performs on par with MLMs on structural probing of unlabeled undirected dependency trees. This contrast highlights the importance of probe architecture and task design.

Linguistic Information and LMs Sinha et al. (2021) found that MLMs rely on higher-order cooccurrence statistics for many tasks, and can maintain high performance without word order information. We view the same problem from a different angle: Our scoring results show that, expectedly, LM predictions are optimized for co-occurrence information. Probing performance, however, can be maintained to a high-degree even if co-occurrence at the lexical level is disrupted. Our comparison of MLMs and ALMs suggests that both architectures process nonce data differently, and MLMs are better at identifying syntactic structure in SPUD data than ALMs.

Limitations

Model instances and architectures All our direct comparisons between model architectures are limited by the fact that we compare models with different hyperparameters, training data, and tokenizers. To quantify the effect of each of these components on the syntactic learning requires availability of models trained using various combination of these settings.

[2] In a small-scale qualitative analysis, we find that this can explain many errors made by ALMs but not MLMs.

Data While we put significant effort into assuring the quality of the generated data, we did not conduct a large-scale human evaluation, e.g. via crowd-sourced grammaticality judgments. This means that while we are confident that grammatical errors in SPUD are rare, rare errors might exist that have potentially severe effects on individual predictions.

Formulations of scoring functions The literature varies in the way in which scoring functions are applied. Assume that $\sigma = \sum_{i=0}^{n} \log p(x_i|context)$ is the sum of log likelihoods in a scoring function such as PPL. We follow Salazar et al. (2020) and divide σ by sequence length and exponentiate $(exp(-(1/n)\sigma))$, as defined in Eq. 1. Kauf and Ivanova (2023), on the contrary, base their experiments directly on σ. Due to the nonlinear differences between both versions, it is possible that not all of our findings on scoring functions are transferable to raw log likelihood scores. We do not expect this to affect our main findings, especially since our comparison of the exponentiated versions of $PPPL$ and $PPPL_{l2r}$ shows trends that are consistent with Kauf and Ivanova (2023)'s findings on English data, the language they focus on. However, we deem a more detailed comparison of scoring functions with and without exponentiation is necessary to draw more general conclusions, even though this is beyond the scope of this paper.

Structural Probing and Properties of Explanations In the following, we discuss limitations of our probing experiments in terms of the properties of explanations defined by Lyu et al. (2024). Several of their criteria are met: First, our experiments are *plausible* (the resulting dependency trees are intuitive to humans), Second, the critierion of *Input sensitivity* relates closely to the interaction of syntax and semantics in probing which we investigate. The definition is that "An explanation should be sensitive (resp. insensitive) to changes in the input that influence (resp. do not influence) the prediction" (Lyu et al., 2024, p. 6). This criterion is met because (i) we directly investigate the effect of changing the semantic content of the probe and model input, and (ii) because DepProbe is generally sensitive to changes in syntax. Third, the criterion of *Faithfulness* states that "an explanation should accurately represent the reasoning process behind the model's prediction" (Lyu et al., 2024, p. 1). Since we do not investigate model predictions but probe predictions, this criterion is not directly applicable. However, we can say that the probe predictions need not be faithful, because DepProbe learns to predict syntactic structure in a supervised fashion, and does not necessarily decode the syntactic generalizations that the model has learnt. Finally, *Completeness* and *Minimality* do not apply because they refer to explaining contributing factors for model predictions rather than probe predictions.

References

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2021. AraGPT2: Pre-trained transformer for Arabic language generation. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 196–207, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

David Arps, Younes Samih, Laura Kallmeyer, and Hassan Sajjad. 2022. Probing for constituency structure in neural language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6738–6757, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

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

Tyler A. Chang and Benjamin K. Bergen. 2023. Language Model Behavior: A Comprehensive Survey. *Computational Linguistics*.

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online. Association for Computational Linguistics.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton and Co., The Hague.

Kareem Darwish and Hamdy Mubarak. 2016. Farasa: A new fast and accurate Arabic word segmenter. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1070–1074, Portorož, Slovenia. European Language Resources Association (ELRA).

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Mathieu Dehouck and Carlos Gómez-Rodríguez. 2020. Data augmentation via subtree swapping for dependency parsing of low-resource languages. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3818–3830, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Kira Droganova, Olga Lyashevskaya, and Daniel Zeman. 2018. Data Conversion and Consistency of Monolingual Corpora: Russian UD Treebanks. In *Proceedings of the 17th international workshop on treebanks and linguistic theories (tlt 2018)*, 155, pages 53–66. Linköping University Electronic Press Linköping, Sweden.

Matthew S. Dryer and Martin Haspelmath, editors. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Tiwalayo Eisape, Vineet Gangireddy, Roger Levy, and Yoon Kim. 2022. Probing for incremental parse states in autoregressive language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2801–2813, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Bruno Guillaume, Marie-Catherine de Marneffe, and Guy Perrier. 2019. Conversion et améliorations de corpus du français annotés en Universal Dependencies [conversion and improvement of Universal Dependencies French corpora]. *Traitement Automatique des Langues*, 60(2):71–95.

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

Jan Hajic, Otakar Smrz, Petr Zemánek, Petr Pajas, Jan Šnaidauf, Emanuel Beška, Jakub Kráčmar, and Kamila Hassanová. 2009. Prague Arabic Dependency Treebank 1.0.

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, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. 2018. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research*, 61:907–926.

Carina Kauf and Anna Ivanova. 2023. A better way to do masked language model scoring. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 925–935, Toronto, Canada. Association for Computational Linguistics.

Carina Kauf, Greta Tuckute, Roger Levy, Jacob Andreas, and Evelina Fedorenko. 2023. Lexical-Semantic Content, Not Syntactic Structure, Is the Main Contributor to ANN-Brain Similarity of fMRI Responses in the Language Network. *Neurobiology of Language*.

Reinhard Köhler. 2003. *Zur Type-Token-Ratio syntaktischer Einheiten: Eine quantitativ-korpuslinguistische Studie*, pages 93–101. Deutscher Universitätsverlag, Wiesbaden.

Artur Kulmizev and Joakim Nivre. 2022. Schrödinger's tree—On syntax and neural language models. *Frontiers in Artificial Intelligence*, 5.

Artur Kulmizev, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre. 2020. Do neural language models show preferences for syntactic formalisms? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4077–4091, Online. Association for Computational Linguistics.

Jenny Kunz and Marco Kuhlmann. 2020. Classifier probes may just learn from linear context features. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5136–5146, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jenny Kunz and Marco Kuhlmann. 2021. Test harder than you train: Probing with extrapolation splits. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 15–25, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Karim Lasri, Alessandro Lenci, and Thierry Poibeau. 2022a. Does BERT really agree? fine-grained analysis of lexical dependence on a syntactic task. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2309–2315, Dublin, Ireland. Association for Computational Linguistics.

Karim Lasri, Olga Seminck, Alessandro Lenci, and Thierry Poibeau. 2022b. Subject verb agreement error patterns in meaningless sentences: Humans vs. BERT. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 37–43, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Jey Han Lau, Carlos Armendariz, Shalom Lappin, Matthew Purver, and Chang Shu. 2020. How furiously can colorless green ideas sleep? sentence acceptability in context. *Transactions of the Association for Computational Linguistics*, 8:296–310.

Jey Han Lau, Alexander Clark, and Shalom Lappin. 2017. Grammaticality, Acceptability, and Probability: A Probabilistic View of Linguistic Knowledge. *Cognitive Science*, 41(5):1202–1241.

Tal Linzen and Marco Baroni. 2021. Syntactic structure from deep learning. *Annual Review of Linguistics*, 7(1):195–212.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Qing Lyu, Marianna Apidianaki, and Chris Callison-Burch. 2024. Towards Faithful Model Explanation in NLP: A Survey. *Computational Linguistics*.

Kyle Mahowald, Anna A. Ivanova, Idan Asher Blank, Nancy Kanwisher, Joshua B. Tenenbaum, and Evelina Fedorenko. 2023. Dissociating language and thought in large language models: a cognitive perspective. *CoRR*, abs/2301.06627.

Christopher D. Manning, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences*, 117(48):30046–30054.

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

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

Rowan Hall Maudslay and Ryan Cotterell. 2021. Do syntactic probes probe syntax? experiments with jabberwocky probing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 124–131, Online. Association for Computational Linguistics.

Alessio Miaschi, Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi. 2021. What makes my model perplexed? a linguistic investigation on neural language models perplexity. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 40–47, Online. Association for Computational Linguistics.

Max Müller-Eberstein, Rob van der Goot, and Barbara Plank. 2022. Probing for labeled dependency trees. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7711–7726, Dublin, Ireland. Association for Computational Linguistics.

Attila Nagy, Dorina Lakatos, Botond Barta, and Judit Ács. 2023. TreeSwap: Data augmentation for machine translation via dependency subtree swapping. In *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*, pages 759–768, Varna, Bulgaria. INCOMA Ltd., Shoumen, Bulgaria.

Benjamin Newman, Kai-Siang Ang, Julia Gong, and John Hewitt. 2021. Refining targeted syntactic evaluation of language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3710–3723, Online. Association for Computational Linguistics.

Isabel Papadimitriou, Richard Futrell, and Kyle Mahowald. 2022. When classifying grammatical role, BERT doesn't care about word order... except when it matters. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 636–643, Dublin, Ireland. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Shauli Ravfogel, Yanai Elazar, Jacob Goldberger, and Yoav Goldberg. 2020. Unsupervised distillation of syntactic information from contextualized word representations. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 91–106, Online. Association for Computational Linguistics.

Benoît Sagot. 2018. A multilingual collection of CoNLL-U-compatible morphological lexicons. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC*

2018), Miyazaki, Japan. European Language Resources Association (ELRA).

Gözde Gül Şahin and Mark Steedman. 2018. Data augmentation via dependency tree morphing for low-resource languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5004–5009, Brussels, Belgium. Association for Computational Linguistics.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Oleh Shliazhko, Alena Fenogenova, Maria Tikhonova, Anastasia Kozlova, Vladislav Mikhailov, and Tatiana Shavrina. 2024. mGPT: Few-Shot Learners Go Multilingual. *Transactions of the Association for Computational Linguistics*, 12:58–79.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Chris Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2897–2904, Reykjavik, Iceland. European Language Resources Association (ELRA).

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jon Sprouse, Beracah Yankama, Sagar Indurkhya, Sandiway Fong, and Robert C. Berwick. 2018. Colorless green ideas do sleep furiously: gradient acceptability and the nature of the grammar. *The Linguistic Review*, 35(3):575–599.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Clara Vania, Yova Kementchedjhieva, Anders Søgaard, and Adam Lopez. 2019. A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1105–1116, Hong Kong, China. Association for Computational Linguistics.

Dingquan Wang and Jason Eisner. 2016. The galactic dependencies treebanks: Getting more data by synthesizing new languages. *Transactions of the Association for Computational Linguistics*, 4:491–505.

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

Tatu Ylonen. 2022. Wiktextract: Wiktionary as machine-readable structured data. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1317–1325, Marseille, France. European Language Resources Association.

Taha Zerrouki. 2010. Pyarabic, an arabic language library for python. `https://pypi.python.org/pypi/pyarabic`.

A Nonce treebanks

A.1 Languages and resources

Table 6 summarizes the languages for which SPUD treebanks are generated. In general, we use the UD test sets for evaluation, and also sample replacement candidates from the UD test sets to avoid data leakage. For all experiments, we used the UD release 2.10.

A.2 Language-specific rules

For consistency, we implement the general rule in languages written in Latin script that the replacement has to be capitalized if the replaced word was capitalized.

Arabic No language-specific rules are applied, except for the removal of diacritics to increase the number of possible replacements.

German Adjective suffixes are depending on several features, however, not all of these features are consistently annotated in the available resources. Concretely, the forms are inflected for Case, Number, Genus, Degree of comparison, and Determinacy, leading to large inflection paradigms. For this reason, we implement the rule that for adjectives ending in *-e, -em, -en, -er, -es*, the replaced adjective has to have the same ending.

English A word list is compiled from wiktionary using wiktextract (Ylonen, 2022) to determine words starting with consonants or vowels. This list is used to determine the correct indefinite article for English nouns: The choice of *a* or *an* depends on the following word, which can be a replaced content word (e.g., an adjective or noun).

	Family, Genus	Writing system	UD treebank	UD$_{train}$ tokens
Arabic	Afro-Asiatic, Semitic	Arabic	PADT, (Hajic et al., 2009)	224K
English	IE, Germanic	Latin	EWT, (Silveira et al., 2014)	205K
French	IE, Romance	Latin	GSD, (Guillaume et al., 2019)	355K
German	IE, Germanic	Latin	HDT, (Borges Völker et al., 2019)	2,754K
Russian	IE, Slavic	Cyrillic	SynTagRus, (Droganova et al., 2018)	1,206K

Table 6: Summary of the languages for which nonce data is generated. IE = Indo-European. Family and Genus according to Dryer and Haspelmath (2013).

The determiner is adjusted if the first sound of the following word changes from a vowel to a consonant or vice versa. For example, when replacing *apple* in *an apple* with *bicyle*, the result is *a bicycle* and not **an bicycle*.

French In French, we implement a similar rule as for English determiners. Concretely, we replace *le/la/de/* with *l'/d'* and vice versa if the following word starts with a vowel, consonant, or aspirated *h*, respectively. The pronunciation of the following word is determined from wiktionary using wiktextract (Ylonen, 2022). Furthermore, French adjectives fall into three classes: A fixed set of adjectives precedes the noun. The majority of adjectives follow the noun. A small set of adjectives can appear in both positions. For instance, *grande maison* – 'big house' and *voiture rouge* – 'red car' are correct, but **maison grande* and **rouge maison* are not. The syntactic context as defined in generating nonce data does not capture the adjective classes. Thus, we replace adjectives preceding (following) the head only with adjectives that also precede (follow) the head in the UD treebank.

Russian The rich case system in Russian poses a challenge for the algorithm as defined above. Concretely, the case of dependents is not part of the syntactic context. Contrary to the other languages in our sample, the case of an `object` is determined by the verb, and both accusative and genetive case are possible and frequent cases for these objects. This means that if a verb with an accusative object is replaced with a verb with a genitive object, a case mismatch is introduced. Since Russian nevertheless retains high probing performance (Sec. 5), we did not explicitly address this problem.

A.3 Examples

For illustration purposes, we present relatively short sentences with multiple nonce versions in Figures 3 (ar), 4 (de), 5 (en), 6 (fr), and 7 (ru).

	NOUN	PROPN	ADJ	ADV	VERB	total
ar	0.93	0.77	0.93	0.74	0.66	0.47
de	0.96	0.93	0.68	0.92	0.61	0.41
en	0.85	0.84	0.86	0.83	0.59	0.38
fr	0.83	0.69	0.88	0.92	0.50	0.36
ru	0.84	0.70	0.80	0.89	0.46	0.39

Table 7: Ratio of replaced words per POS in the test set. The column "total" shows the replacement ratio over all POS, including function words and punctuation.

A.4 Replacement statistics

Tab. 7 shows the ratio of replaced words per POS in the test set. Of the 25 ratios (5 languages, 5 POS tags), 6 are above 0.9 and 14 are above 0.8, indicating that the algorithm is able to find replacements for the vast majority of content words. The replacement ratio is lowest for verbs, for two reasons: First, verbs are generally less frequent than e.g. nouns. Second, verbs frequently have a relatively complex argument structure, which leads to a large number of syntactic contexts and an increased frequency of syntactic contexts that are not shared with other verbs.

B List of Models

mGPT (Shliazhko et al., 2024, huggingface id: `ai-forever/mGPT`) and mBERT (Devlin et al., 2019, `bert-base-multilingual-cased`) are used for all experiments. English scoring and probing experiments are additionally conducted with RoBERTa (Liu et al., 2019, `roberta-base`) and GPT-2 (Radford et al., 2019, `gpt2`). Arabic scoring experiments are additionally conducted with AraGPT2 (Antoun et al., 2021, `aubmindlab/aragpt2-base`) and AraBERT (Antoun et al., 2020, `aubmindlab/bert-base-arabertv2`).

C Data preprocessing

We apply the following preprocessing steps to the UD treebanks after creating SPUD splits. To allow a

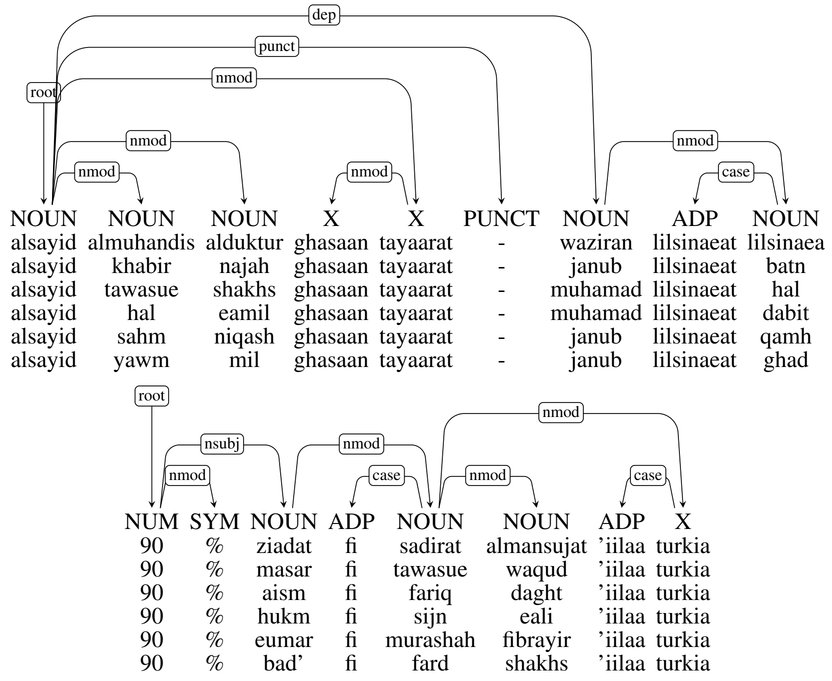

dep
punct
nmod
root
nmod
nmod
nmod
nmod
case
NOUN NOUN NOUN X X PUNCT NOUN ADP NOUN
alsayid almuhandis alduktur ghasaan tayaarat - waziran lilsinaeat lilsinaea
alsayid khabir najah ghasaan tayaarat - janub lilsinaeat batn
alsayid tawasue shakhs ghasaan tayaarat - muhamad lilsinaeat hal
alsayid hal eamil ghasaan tayaarat - muhamad lilsinaeat dabit
alsayid sahm niqash ghasaan tayaarat - janub lilsinaeat qamh
alsayid yawm mil ghasaan tayaarat - janub lilsinaeat ghad
root
nmod
nsubj
nmod
nmod
case
nmod
case
NUM SYM NOUN ADP NOUN NOUN ADP X
90 % ziadat fi sadirat almansujat 'iilaa turkia
90 % masar fi tawasue waqud 'iilaa turkia
90 % aism fi fariq daght 'iilaa turkia
90 % hukm fi sijn eali 'iilaa turkia
90 % eumar fi murashah fibrayir 'iilaa turkia
90 % bad' fi fard shakhs 'iilaa turkia

Figure 3: Arabic SPUD examples (in transliteration).

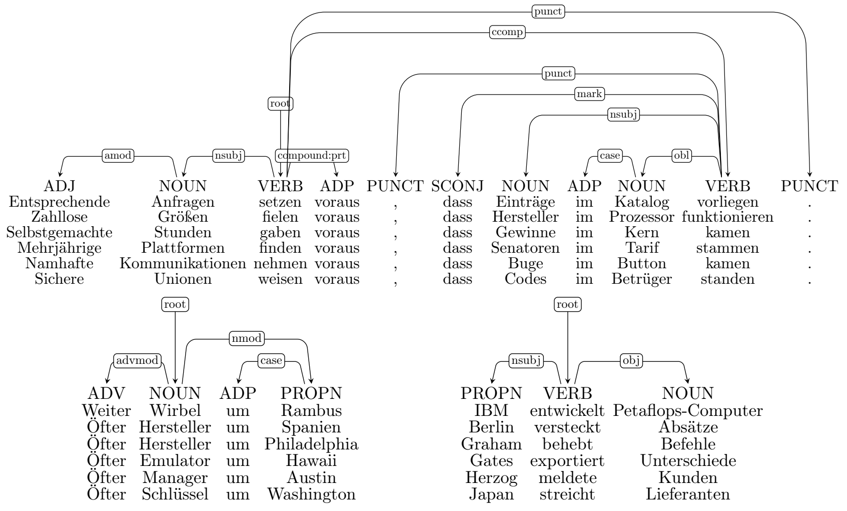

punct
ccomp
punct
mark
root
nsubj
amod
nsubj
compound:prt
case
obl
ADJ NOUN VERB ADP PUNCT SCONJ NOUN ADP NOUN VERB PUNCT
Entsprechende Anfragen setzen voraus , dass Einträge im Katalog vorliegen .
Zahllose Größen fielen voraus , dass Hersteller im Prozessor funktionieren .
Selbstgemachte Stunden gaben voraus , dass Gewinne im Kern kamen .
Mehrjährige Plattformen finden voraus , dass Senatoren im Tarif stammen .
Namhafte Kommunikationen nehmen voraus , dass Buge im Button kamen .
Sichere Unionen weisen voraus , dass Codes im Betrüger standen .
root
nmod
advmod
case
ADV NOUN ADP PROPN
Weiter Wirbel um Rambus
Öfter Hersteller um Spanien
Öfter Hersteller um Philadelphia
Öfter Emulator um Hawaii
Öfter Manager um Austin
Öfter Schlüssel um Washington
root
nsubj
obj
PROPN VERB NOUN
IBM entwickelt Petaflops-Computer
Berlin versteckt Absätze
Graham behebt Befehle
Gates exportiert Unterschiede
Herzog meldete Kunden
Japan streicht Lieferanten

Figure 4: German SPUD examples.

fair comparison between MLMs and ALMs, we filter out sentences that are shorter than 4 words from the data in all experiments. On short sentences, ALMs have a disadvantage because their per-token score increases over the sentence given increasing context length, while the basic masked language modeling task assumes access to the same amount of context for each token. For SPUD in Arabic, we remove diacritics with the PyArabic library (Zerrouki, 2010). The Arabic morphological lexicon is lemmatized with Farasa (Darwish and Mubarak, 2016).

D Lexical Diversity of Nonce data

	TTR	
Language	orig.	nonce
ar	.0094	.0077
de	.0054	.0043
en	.0571	.0463
fr	.0441	.0297
ru	.0052	.0044

Table 8: Type-Token Ratio of underlying UD treebanks and SPUD versions.

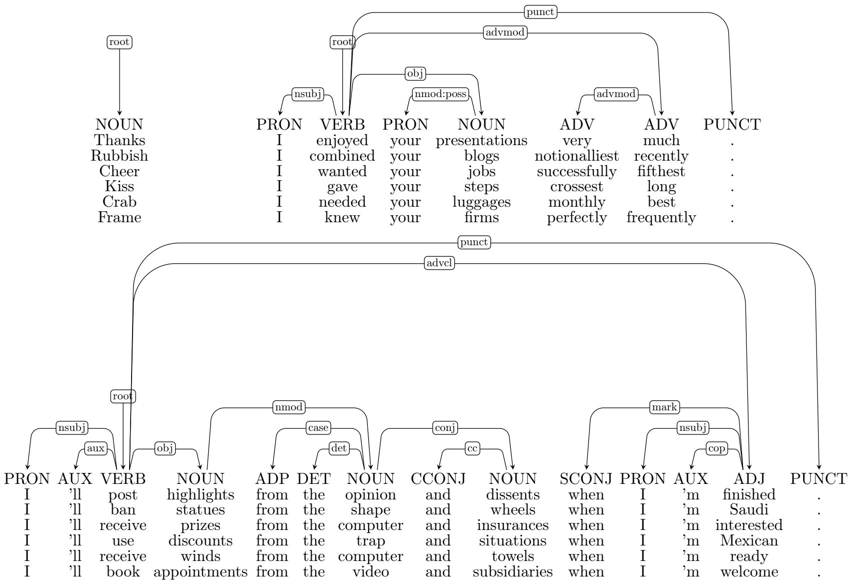

root
punct
advmod
root
obj
nsubj
nmod:poss
advmod
NOUN PRON VERB PRON NOUN ADV ADV PUNCT
Thanks I enjoyed your presentations very much .
Rubbish I combined your blogs notionalliest recently .
Cheer I wanted your jobs successfully fifthest .
Kiss I gave your steps crossest long .
Crab I needed your luggages monthly best .
Frame I knew your firms perfectly frequently .
punct
advcl
root
nmod
mark
nsubj
aux
obj
case
det
conj
cc
nsubj
cop
PRON AUX VERB NOUN ADP DET NOUN CCONJ NOUN SCONJ PRON AUX ADJ PUNCT
I ’ll post highlights from the opinion and dissents when I ’m finished .
I ’ll ban statues from the shape and wheels when I ’m Saudi .
I ’ll receive prizes from the computer and insurances when I ’m interested .
I ’ll use discounts from the trap and situations when I ’m Mexican .
I ’ll receive winds from the computer and towels when I ’m ready .
I ’ll book appointments from the video and subsidiaries when I ’m welcome .

Figure 5: English SPUD examples.

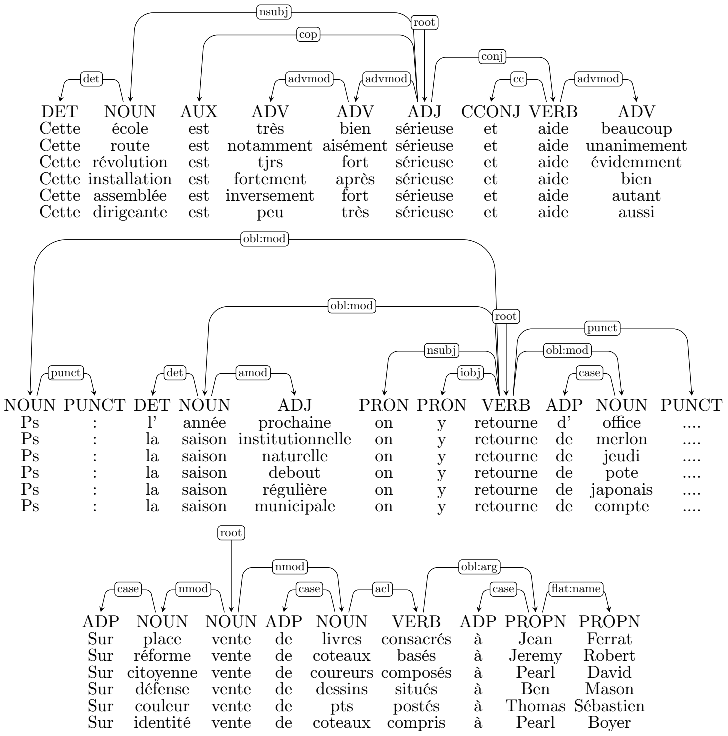

nsubj
root
cop
conj
det
advmod
advmod
cc
advmod
DET NOUN AUX ADV ADV ADJ CCONJ VERB ADV
Cette école est très bien sérieuse et aide beaucoup
Cette route est notamment aisément sérieuse et aide unanimement
Cette révolution est tjrs fort sérieuse et aide évidemment
Cette installation est fortement après sérieuse et aide bien
Cette assemblée est inversement fort sérieuse et aide autant
Cette dirigeante est peu très sérieuse et aide aussi
obl:mod
obl:mod
root
punct
nsubj
obl:mod
punct
det
amod
iobj
case
NOUN PUNCT DET NOUN ADJ PRON PRON VERB ADP NOUN PUNCT
Ps : l’ année prochaine on y retourne d’ office ....
Ps : la saison institutionnelle on y retourne de merlon ....
Ps : la saison naturelle on y retourne de jeudi ....
Ps : la saison debout on y retourne de pote ....
Ps : la saison régulière on y retourne de japonais ....
Ps : la saison municipale on y retourne de compte ....
root
nmod
obl:arg
case
nmod
case
acl
case
flat:name
ADP NOUN NOUN ADP NOUN VERB ADP PROPN PROPN
Sur place vente de livres consacrés à Jean Ferrat
Sur réforme vente de coteaux basés à Jeremy Robert
Sur citoyenne vente de coureurs composés à Pearl David
Sur défense vente de dessins situés à Ben Mason
Sur couleur vente de pts postés à Thomas Sébastien
Sur identité vente de coteaux compris à Pearl Boyer

Figure 6: French SPUD examples.

Model-Independent Comparison: Type Token Ratio (TTR) is a widely used measure of lexical diversity. where a larger value indicates that less tokens are used repeatedly (Köhler, 2003). Apart from the segmentation model (in our case the mBERT tokenizer), TTR focuses on capturing the distribution of the data relatively independent from large computational or methodological overhead. The TTR (when tokenized with mBERT) in all languages is shown in Tab. 8.

E Scoring Experiments

E.1 Example for Scoring Functions

An example for how all scoring functions are computed for the phrase *accordeon player* is displayed in Tab. 9.

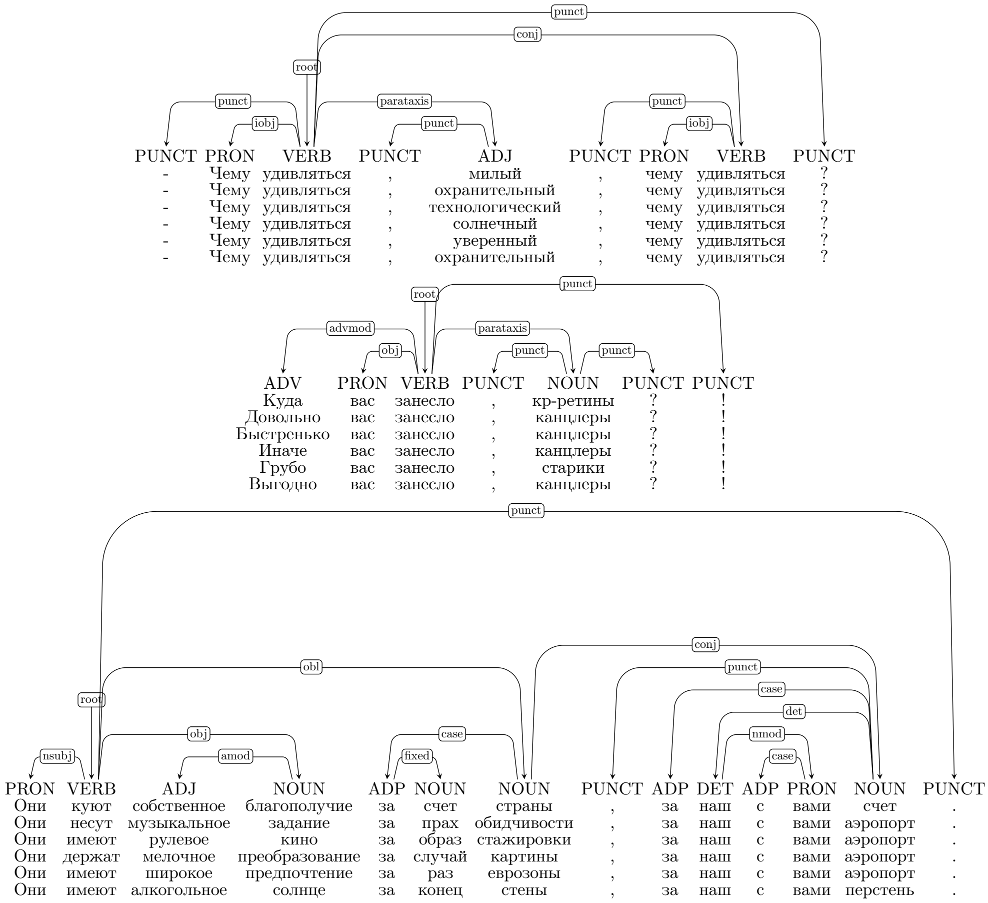

punct
conj
root
punct
iobj
parataxis
punct
punct
iobj
PUNCT PRON VERB PUNCT ADJ PUNCT PRON VERB PUNCT
- Чему удивляться , милый , чему удивляться ?
- Чему удивляться , охранительный , чему удивляться ?
- Чему удивляться , технологический , чему удивляться ?
- Чему удивляться , солнечный , чему удивляться ?
- Чему удивляться , уверенный , чему удивляться ?
- Чему удивляться , охранительный , чему удивляться ?
root
punct
advmod
parataxis
obj
punct
punct
ADV PRON VERB PUNCT NOUN PUNCT PUNCT
Куда вас занесло , кр-ретины ? !
Довольно вас занесло , канцлеры ? !
Быстренько вас занесло , канцлеры ? !
Иначе вас занесло , канцлеры ? !
Грубо вас занесло , старики ? !
Выгодно вас занесло , канцлеры ? !
punct
conj
obl
punct
case
det
root
obj
case
nmod
nsubj
amod
fixed
case
PRON VERB ADJ NOUN ADP NOUN NOUN PUNCT ADP DET ADP PRON NOUN PUNCT
Они куют собственное благополучие за счет страны , за наш с вами счет .
Они несут музыкальное задание за прах обидчивости , за наш с вами аэропорт .
Они имеют рулевое кино за образ стажировки , за наш с вами аэропорт .
Они держат мелочное преобразование за случай картины , за наш с вами аэропорт .
Они имеют широкое предпочтение за раз еврозоны , за наш с вами аэропорт .
Они имеют алкогольное солнце за конец стены , за наш с вами перстень .

Figure 7: Russian SPUD examples.

	accord	##eon	player
PPL	?	_	_
	accord	?	_
	accord	##eon	?
$PPPL$	?	##eon	player
	accord	?	player
	accord	##eon	?
$PPPL_{l2r}$	?	_	player
	accord	?	player
	accord	##eon	?

Table 9: The token for which a probability is recorded is marked with "?". Tokens that are masked out or not available from context are marked with "_".

E.2 Scoring Experiment

In Tab. 10, we show the ratios of scoring functions for multilingual models in German, French and Russian (complementing the English and Arabic results in Tab. 2). In Tab. 11, we show the ratios of scoring functions for monolingual models in Arabic and English.

E.3 Scoring outlier analysis

For English, we investigate the sentence pairs with the highest and lowest scoring function ratios. The sentences with the highest ratio are shown in Table 12. Sentences with very high ratios are all relatively short. The large difference in ratios between scores assigned by mBERT and mGPT indicate that these original sentences could be part of the pre-training data for mGPT but not mBERT - the former is overfitting to the original sentence. Also, these sentences contain uppercase sentences (which might affect tokenization), and almost all original sentences contain well-known named entities or common phrases that are disrupted in the nonce data (*U.S. Senate Committee*, *AS FAR AS POSSIBLE*, etc.)

The sentences with the lowest ratio are shown in Table 13. The examples include typos in the original data (*ling* instead of *long*), an example in which the original seems ungrammatical (6), and examples in which only single words are replaced by other forms that are presumably more frequent (*cover* - *go*, *Per* - *Google*, ...)

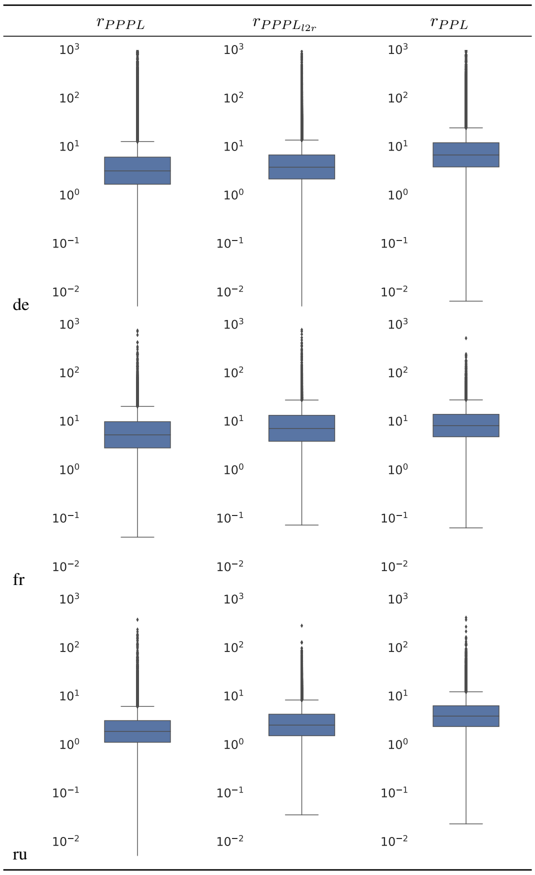


Table 10: Scoring function ratios for multilingual models.

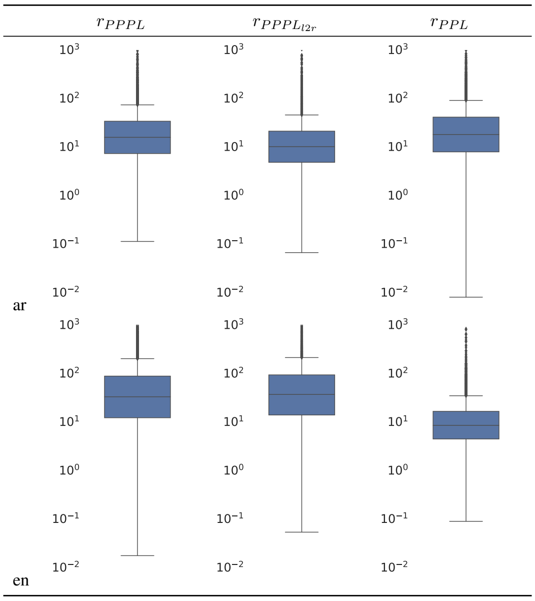


Table 11: Scoring function ratios for monolingual models (Arabic and English).

F Probing Experiments

F.1 DepProbe architecture

Müller-Eberstein et al. (2022) presented **DepProbe**, a lightweight decoder for directed and labeled dependency trees. The model consists of two matrices, L and B. Both transform d_h-dimensional word representations[3] into vectors that hightlights a syntactic property of that representation. Assume that the LM representation of a word w_i is $h_i \in \mathbb{R}^{d_h}$, and that the annotation has l dependency relations.

Predicting dependency relations The matrix $L \in \mathbb{R}^{d_h \times l}$ is a linear classifier that predicts for each word the label of this word's incoming dependency edge – the relation between the word and its head. Concretely, the "probability of a word's relation r_i being of class l_k is given by:

$$p(r_i = l_k|w_i) = softmax(Lh_i)_k \tag{4}$$

" (Müller-Eberstein et al., 2022, p. 7713). This model component is trained using cross-entropy loss.

Predicting word distances The matrix $B \in \mathbb{R}^{d_h \times b}$ predicts the dependency edges between words. B projects the LM representations in a vector space that has less dimensions than the LM layer ($b < d_h$). The target vector space is informally called the syntactic subspace: It reflects structural information such that vector distances between words mimic the distance between words in the dependency tree. Concretely, when h_i, h_j are the LM representations of words w_i, w_j, their distance $d_B(h_i, h_j)$ in the syntactic subspace is defined by Müller-Eberstein et al. (2022, Eq. 1):

$$d_B(h_i, h_j) = \sqrt{(Bh_i - Bh_j)^T(Bh_i - Bh_j)} \tag{5}$$

This model component is trained to predict the distance between all word pairs in the dependency tree. Assume that the distance between two words in the dependency tree is defined as the path length between the two words, $d_P(w_i, w_j)$. If s is a sentence of length $N + 1$, the loss for optimizing B is given by Müller-Eberstein et al. (2022, Eq. 2):

$$\mathcal{L}_B(s) = \frac{1}{N^2} \sum_{i=0}^{N} \sum_{j=0}^{N} |d_P(w_i, w_j) - d_B(h_i, h_j)| \tag{6}$$

Constructing the dependency tree The outputs of the relation and distance components are combined to construct the dependency tree starting from the root. The word with the highest probability of being the root (as assigned by L) is set

[3] For tokens consisting of multiple subwords, the word representation is the mean of the subword representations.

orig.	nonce	r_{PPPL}	$r_{PPPL_{l2r}}$	r_{PPL}
NORTH CAROLINA RELIGIOUS COALITION FOR MARRIAGE...	D' Vernon ugly Coalition For El Dong	388.97	176.08	3320.79
U.S. Senate Committee on Appropriations	Can Bay Committee on Williss'	35734.78	16130.36	1027.64
SHE KNOWS GREAT FOOD AND DINING EXPERIENCES.	She solicits Swedish election And rabbit outlets.	8.67	16.67	968.43
BEST CHINESE RESTAURANT EVER!!!	Truest corrupt Restaurant definitely!!!	4.57	7.44	855.44
THEY ARE VERY RUDE AND NASTY.	They Are allegedly happy And hidden.	80.31	38.98	797.19
my new OLYMPUS X940 DIGITAL CAMERA...?	My inhuman April Port wireless Camera...?	10.03	5.29	759.14
STAY AWAY AS FAR AS POSSIBLE.	Move wrongest currently low As satisfactory.	31.35	28.88	608.69
Super Pet Silent Spinner Exercise Wheel	Standard Fidelity alleged Alliance Prince Wheel	257.01	74.85	538.77
CHERNOBYL ACCIDENT: TEN YEARS ON	Greenfield diplomacy: Ten Years partly	10.86	17.24	511.29

Table 12: Selection of sentences with r_{PPL} above 500. Data is printed verbatim, including casing, punctuation, etc.

orig	nonce	r_{PPPL}	$r_{PPPL_{l2r}}$	r_{PPL}
New training Centre is excellent	Extensive vacation temperature is amazing	0.01	0.32	0.04
Maria Valdes superior $62,500	Todd Taylors' strong $62,500	0.47	0.24	0.11
Gold award parts excellence, metro.	Hoodie leader coffees tank, Michelle.	1.60	2.17	0.14
Talked to Craig and the Court today.	Talked to Emily and the viewer yesterday.	0.05	0.43	0.16
oh god is there an agenda.	Oh god is there an antibiotic.	0.22	0.20	0.18
Broke out the activities of 1179.	Checked out the spirits of 1179.	0.79	0.69	0.21
Shanna, I spoke with Per tonight about this.	Kim, I spoke with Google yesterday about this.	0.30	0.30	0.24
George Bush: Money manager	Roger's Dick's: garage therapist	0.04	0.13	0.25
Is it possible to shoot lazers out of your Wang?	Is it great to see e-mails out of your house?	0.08	0.06	0.28
if they hiss, they are not playing.	If they are, they are not running.	0.62	0.49	0.34
Even though you are expensive.	Certainly though you are expensive.	3.74	5.69	0.39
See you all there - this is ling overdue	See you all there - this is here deep	1.19	0.56	0.44
Can you cover for me today?	Can you go for me tomorrow?	0.18	0.43	0.46
Brazil we have current data already	Danny we have ambitious tongue feverishly	0.08	0.82	0.47
Your cat will adjust quickly.	Your health will starve inappropriateliest.	0.76	2.27	0.48

Table 13: Selection of sentences with $r_{PPL} < 0.5$. Data is printed verbatim, including casing, punctuation, etc. Data with such low ratios also included longer sentences, which are omitted for space reasons.

as the dependency tree root. Then, the words are iteratively added to the dependency tree, always choosing the word with the smallest distance d_B to a word that is already covered. The dependency labels are assigned based on L. A detailed example of this process is shown in App. F.2.

F.2 DepProbe Example

In Fig. 8, we present a step-by-step example for for reconstructing trees with DepProbe, following Alg. 1 in (Müller-Eberstein et al., 2022). For simplicity, the correct distance matrix (Fig. 8a) and correct relations are taken as input to the algorithm.

F.3 Baselines

mBERT embedding layer (M_0) In this baseline, we use the output of mBERT's embedding layer as input to DepProbe. M_0 estimates how much information about the probing task is contained in the (context-insensitive) token embeddings.

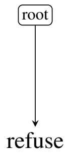
Random mBERT (M_R) We consider the representations of a transformer model with the same architecture as mBERT, but all parameters are randomly initialized. Performance of this baseline estimates how much information about dependency structures is contained in the architecture of an MLM and the DepProbe training.

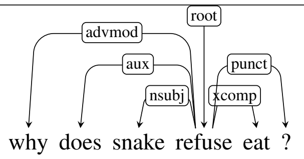
Random contextualization mBERT (M_{RC}) We consider a transformer model with the same architecture as mBERT, but all parameters except the embedding layer are randomly initialized. The performance of this baseline estimates how much information about dependency structures from linguistic context is acquired during pretraining of the MLM (Belinkov, 2022). We take the representations from layers 6 (7) as input to DepProbe's distance (relation) component.

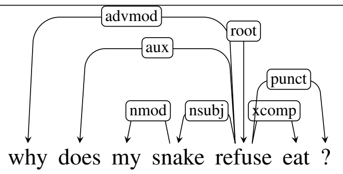
Results All probed models outperform the baselines by a large margin, both on original and nonce data (Tables 14, 15, 16). This shows that the probing task probes for contextual information (comparison to M_0), and that the probed information is learnt during pretraining (comparison to M_R and M_{RC}). M_0, the embedding layer baseline, shows decent performance on the relation labeling task (Tab. 14): Up to 74.6 RelAcc for German, and $\Delta < 5$ for all languages. For attachment, the ab-

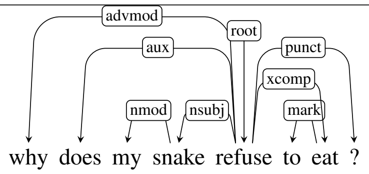
	why	does	my	snake	refuse	to	eat	?
why	0	2	3	2	1	3	2	2
does	2	0	3	2	1	3	2	2
my	3	3	0	1	2	4	3	3
snake	2	2	1	0	1	3	2	2
refuse	1	1	2	1	0	2	1	1
to	3	3	4	3	2	0	1	3
eat	2	2	3	2	1	1	0	2
?	2	2	3	2	1	3	2	0

(a) Gold distance matrix for the example sentence.

<table>
<tr><th></th><th>covered words</th><th>tree</th><th>comment</th></tr>
<tr><td>1</td><td>$\{refuse\}$</td><td>


</td><td>maximum $p(\texttt{root}|refuse)$</td></tr>
<tr><td>2-6</td><td>{<i>why, does, snake, refuse, eat, ?</i>}</td><td>


</td><td>all distance 1 to root <i>refuse</i></td></tr>
<tr><td>7</td><td>{<i>why, does, my, snake, refuse, eat, ?</i>}</td><td>


</td><td><i>snake</i> is only word with distance 1 to <i>my</i></td></tr>
<tr><td>8</td><td>{<i>why, does, my, snake, refuse, to, eat, ?</i>}</td><td>


</td><td><i>to</i> is only word with distance 1 to <i>eat</i></td></tr>
</table>

(b) Step by Step example.

Figure 8: DepProbe tree decoding example

solute scores are lower and the performance drop is larger. This is intuitive: The embedding layer provides a basis for predicting context-insensitive information such as the most frequent dependency relation of a word type, and this information is still useful for predicting the dependency relations in SPUD data. However, predicting attachment requires more contextual information, for SPUD data even more so than for original data. The absolute performance and Δ of M_{RC} is within 1 point to M_0 for all languages (Tab. 15). This shows that the random contextualization does not detriment the predictability of relation labeling based on the embedding layer. Attachment performance is lower in terms of absolute scores for M_{RC} than for M_0, which leads to less room for a performance drop but overall lower LAS on both test sets. On relation labeling, M_{RC} performs between 4.8 and 12.6 accuracy points better than M_R on original data. On attachment, M_R performs better than M_{RC} for all languages except Arabic. All baseline performance drops for nonce data are relatively small. The most intuitive explanation for this is that on the one hand, lower overall performance leaves less room for a performance drop, and on the other hand it confirms that nonce data is created in a way in which only the co-occurence information is changed, but not the syntactic contexts in which the words generally appear (e.g. the tendency of a word to appear as a subject is preserved).

F.4 Random seeds

The experimental setup introduced two random components: The probe's initial weights are randomly initialized, and the nonce data is created by randomly sampling words. To test the effect

	RelAcc		UAS		LAS	
	orig	Δ	orig	Δ	orig	Δ
ar	66.1	4.7	43.3	5.8	29.9	5.9
de	74.6	2.8	49.7	5.5	40.8	6.1
en	65.4	1.6	45.6	3.5	33.7	3.2
fr	70.6	2.1	48.2	3.0	36.9	3.2
ru	71.3	2.0	46.6	3.3	35.3	3.3

Table 14: M_0. Embedding layer of mBERT

	RelAcc		UAS		LAS	
	orig	Δ	orig	Δ	orig	Δ
ar	65.4	4.7	31.0	4.1	21.9	4.1
de	75.2	3.2	46.4	5.3	38.7	6.1
en	66.4	1.7	41.9	3.0	32.0	2.9
fr	70.9	2.5	40.6	2.3	31.7	2.4
ru	71.0	2.0	42.0	2.7	32.4	2.8

Table 15: M_{RC}. Mid layers of mBERT architecture with trained embedding matrix and random transformer layers

	RelAcc		UAS		LAS	
	orig	Δ	orig	Δ	orig	Δ
ar	60.6	2.6	38.9	3.1	25.6	2.6
de	64.6	6.0	39.6	5.2	29.5	6.2
en	58.4	3.0	37.8	2.9	25.7	3.3
fr	63.7	2.1	39.8	3.1	27.3	2.3
ru	58.4	2.0	36.1	2.3	22.9	2.0

Table 16: M_R. Mid layers of mBERT architecture with completely random parameters

	orig.	SPUD	Δ mean
ar	84.6 ± 0.13	78.9 ± 0.13	5.7
de	92.5 ± 0.04	90.8 ± 0.05	1.7
en	86.5 ± 0.10	84.8 ± 0.10	1.7
fr	89.9 ± 0.13	87.9 ± 0.25	2.0
ru	88.8 ± 0.00	87.3 ± 0.04	1.5

Table 17: Mean and standard deviation RelAcc of probing with different random seeds.

	orig	nonce	Δ mean
ar	63.5 ± 0.18	56.0 ± 0.18	7.5
de	83.3 ± 0.08	80.0 ± 0.09	3.3
en	73.0 ± 0.31	69.3 ± 0.35	3.6
fr	75.5 ± 0.17	71.3 ± 0.33	4.2
ru	74.8 ± 0.02	71.6 ± 0.15	3.2

Table 18: Mean and standard deviation UAS of probing with different random seeds.

	orig.	nonce	Δ means
ar	56.9 ± 0.13	48.0 ± 0.17	8.9
de	79.5 ± 0.06	75.6 ± 0.07	4.0
en	67.3 ± 0.29	63.6 ± 0.31	3.8
fr	70.8 ± 0.18	66.5 ± 0.32	4.3
ru	69.3 ± 0.01	66.2 ± 0.17	3.2

Table 19: Mean and standard deviation LAS of probing with different random seeds.

of these random components, we additionally run experiments with varying random seeds for each of these components in a pilot study. Per language, we initialize probes with 3 different random seeds, and each of these probes is evaluated on 5 different SPUD datasets. This means that per language, 3 predictions for original data and 15 predictions for nonce data are available. For all 3 evaluation metrics, the resulting scores across predictions are stable. Concretely, the largest observed standard deviation across predictions was 0.35 points UAS for English SPUD data. Most of the other settings show much lower standard deviations. We argue that these random components do not have a large effect on the results, and that it is therefore sufficient to run the above probing experiments as described.

F.5 Ignoring dependency relations in nonce data creation

To compare our approach against the common nonce data creation practice of using POS tags, we test on a version of English and German nonce data that ignores dependency-specific information and only modifies the data based on POS tags and language-specific rules. The results are shown in Table 20. The performance difference between the two versions of nonce data is larger for English than for German. This supports the fact that in English, dependency information is more easily extracted from positional information than in German: In the English data, the position of a word is a stronger cue for its dependency relation than in German.

F.6 Layer dymanics

We train DepProbe on English data on all layers of mBERT and mGPT. On the one hand, this serves the purpose of selecting the layer used in all other experiments. Müller-Eberstein et al. (2022) conducted the same experiment on mBERT to identify the layer that performs best on the probing task.

	RelAcc		UAS		LAS	
	SPUD	POS	SPUD	POS	SPUD	POS
de	91.0	90.1	81.0	79.7	76.2	75.0
en	84.8	81.7	70.4	66.0	63.8	59.2

Table 20: Results for English and German test sets with (SPUD) and without (only POS) dependency information in the nonce data creation.

	left	right
ar	0.67	0.31
de	0.32	0.63
en	0.37	0.56
fr	0.40	0.56
ru	0.42	0.53

Table 21: Fraction of edges to the left and right in the respective UD test sets. Both columns do not sum to 1 because the root is not counted.

The best layer for each task (6 for relation labeling, 7 for attachment) is then used in all other experiments. We repeat this process on mGPT to identify the highest performing layer for this model. In the case of mGPT, layer 12 performs best on both tasks. For selecting the best layer, we use the original test data (following existing work).

On the other hand, the layerwise probing results highlight how syntactic and semantic information influence probing performance across layers. For RelAcc, mGPT's performance on both test sets develops relatively stable, with a sharp increase in the lowest layers and a slight decrease in the higher layers. The Δ to the SPUD test set also increases during the lower layers and then stays quite stable. For mBERT, the pattern is similar, with a steeper decline in performance on the highest layer. For UAS (and LAS), mGPT shows smaller changes across layers than for RelAcc. While the pattern appears jumpy, no individual difference between consecutive layers is larger than 3.5 points and the Δ to the SPUD test set is stable. mBERT shows a clearer but steeper trend with best performance in the mid layers and worse performance in the higher and lower layers.

F.7 Probe Performance by edge direction

To further elaborate on the differences between decoded trees from ALMs and MLMs, we investigate the performance of DepProbe in our main experiment on edges in different directions. For all languages, edges to the left are defined as edges

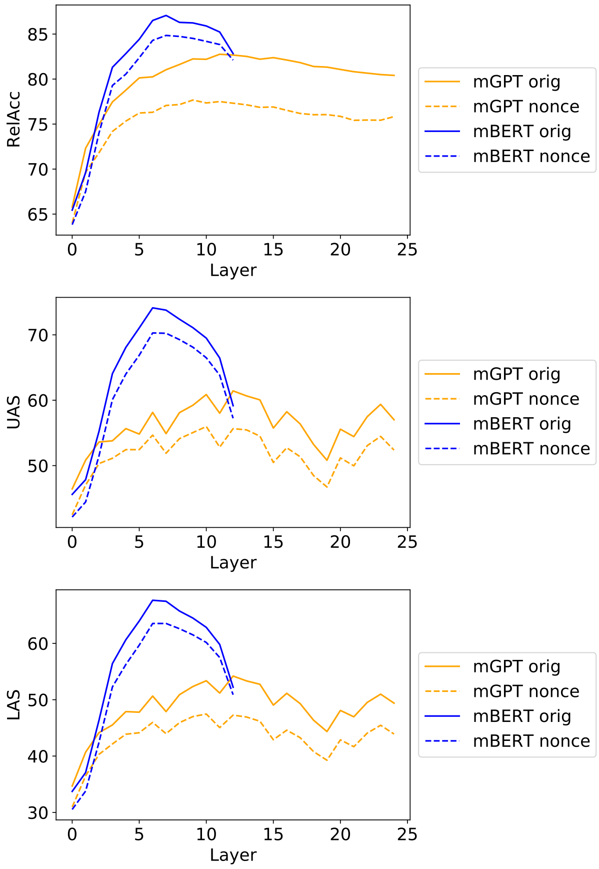


Figure 9: Layerwise DepProbe performance for mBERT and mGPT.

		RelAcc		UAS		LAS	
		l	r	l	r	l	r
mGPT, orig.	ar	81.8	91.9	60.0	56.0	52.0	53.5
	de	88.3	94.5	77.0	86.7	71.1	82.9
	en	75.4	87.4	54.3	58.7	44.5	53.3
	fr	81.1	92.1	66.5	70.5	58.2	66.4
	ru	84.9	91.5	71.7	77.0	64.1	71.8
mGPT, Δ	ar	10.3	2.3	15.6	6.6	16.9	6.7
	de	5.8	2.5	9.3	5.9	10.9	7.0
	en	7.0	3.9	5.5	5.4	6.6	6.4
	fr	2.7	2.9	7.4	5.6	7.0	6.9
	ru	4.2	2.1	7.1	4.1	7.7	4.8
mBERT, orig.	ar	77.8	92.7	63.1	60.9	52.7	58.8
	de	85.7	95.9	75.4	87.8	67.7	85.1
	en	75.4	94.5	65.8	78.0	52.6	75.3
	fr	79.3	96.2	69.6	79.3	58.6	77.2
	ru	81.4	93.3	70.5	77.7	60.3	74.0
mBERT, Δ	ar	7.6	1.1	8.8	4.3	10.4	4.3
	de	2.6	1.2	4.3	3.1	5.1	3.7
	en	2.9	1.8	3.0	4.5	3.1	4.9
	fr	2.8	1.2	4.4	4.3	3.9	4.5
	ru	1.9	1.1	3.6	3.0	3.2	3.0

Table 22: Performance for edges to the left and right

where the dependent has a lower index than the head (vice versa for edges to the right). The relative frequency of edges to the left and right is displayed in Tab. 21. For all Indo-European (IE) languages, the majority of edges are to the right (left for Arabic). The results (Tab. 22) show that for the IE languages, edges to the right show higher performance than edges to the left. For ALMs, this is especially intuitive: The edges to the right are the ones where both head and dependent are in the context of the token on which the prediction is made. When considering only original data and IE languages, the performance difference between ALMs and MLMs interestingly shows that MLMs are better at predicting edges to the right, but not necessarily better at predicting edges to the left. Arabic consitutes a special case because the distribution of edges to the left and right is swapped. For RelAcc and SPUD, edges to the left show lower performance than edges to the right, following the pattern in the other languages. For attachment, the more frequent edges to the left show higher performance than edges to the right (but only in original data). The performance drop on nonce data and RelAcc is larger for edges to the left in all languages and both models. For attachment however, the pattern is more mixed. Since the performance drop on nonce data is secondary to the overarching research question of this particular experiment, we do not investigate this further.

G Computational requirements

All experiments were run on a single NVIDIA GeForce RTX 3090 GPU with 24GB of memory, and powered by renewable energy sources. Collecting the (pseudo-)perplexity scores for one combination of language, model, and scoring function takes between 5 minutes and 2 hours. The duration depends mostly on the dataset size and model architecture. Computing both versions of pseudo-perplexity for a single sentence requires n forward passes for a sentence of length n, while PPL computation requires just one forward pass. Training a single DepProbe model takes between 10 minutes and 3 hours, depending on the size of the LM and treebank. Probe inference is faster. From these values, we estimate that the complete set of experiments reported in this paper can be conducted with our hardware in 25-30 hours. Creating SPUD treebanks does not require GPU access and takes several minutes on a CPU and 64GB RAM.