Masked Language Modeling and the Distributional Hypothesis: Order Word Matters Pre-training for Little

Koustuv Sinha[†‡] **Robin Jia**[†] **Dieuwke Hupkes**[†] **Joelle Pineau**[†‡]

Adina Williams[†] **Douwe Kiela**[†]

[†] Facebook AI Research; [‡] McGill University / Mila - Quebec AI

{koustuvs,adinawilliams,dkiela}@fb.com

Abstract

A possible explanation for the impressive performance of masked language model (MLM) pre-training is that such models have learned to represent the syntactic structures prevalent in classical NLP pipelines. In this paper, we propose a different explanation: MLMs succeed on downstream tasks mostly due to their ability to model higher-order word co-occurrence statistics. To demonstrate this, we pre-train MLMs on sentences with randomly shuffled word order, and show that these models still achieve high accuracy after fine-tuning on many downstream tasks - including tasks specifically designed to be challenging for models that ignore word order. Our models also perform surprisingly well according to some parametric syntactic probes, indicating possible deficiencies in how we test representations for syntactic information. Overall, our results show that purely distributional information largely explains the success of pre-training, and they underscore the importance of curating challenging evaluation datasets that require deeper linguistic knowledge.

1 Introduction

The field of natural language processing (NLP) has become dominated by the pretrain-and-finetune paradigm, where we first obtain a good parametric *prior* in order to subsequently model downstream tasks accurately. In particular, masked language model (MLM) pre-training, as epitomized by BERT (Devlin et al., 2019), has proven wildly successful, although the precise reason for this success has remained unclear. On one hand, we can view BERT as the newest in a long line of NLP techniques (Deerwester et al., 1990; Landauer and Dumais, 1997; Collobert and Weston, 2008; Mikolov et al., 2013; Peters et al., 2018) that exploit the well-known distributional hypothesis (Harris, 1954).[1] On the other hand, it has been claimed that BERT "rediscovers the classical NLP pipeline" (Tenney et al., 2019), suggesting that it has learned "the types of syntactic and semantic abstractions traditionally believed necessary for language processing" rather than "simply modeling complex co-occurrence statistics" (ibid. p.1).

In this work, we aim to uncover how much of MLM's success comes from learning simple distributional information, as opposed to grammatical abstractions (Tenney et al., 2019; Manning et al., 2020). We disentangle these two hypotheses by measuring the effect of removing word order information during pre-training: any sophisticated (English) NLP pipeline would presumably depend on the syntactic information conveyed by the order of words. Surprisingly, we find that most of MLM's high performance can in fact be explained by the "distributional prior" rather than its ability to replicate the classical NLP pipeline.

Concretely, we pre-train MLMs (RoBERTa, Liu et al. 2019) on various corpora with permuted word order while preserving some degree of distributional information, and examine their downstream performance. We also experiment with training MLMs without positional embeddings, making them entirely order agnostic, and with training on a corpus sampled from the source corpus's unigram distribution. We then evaluate these "permuted" models in a wide range of settings and compare with regularly-pre-trained models.

We demonstrate that pre-training on permuted data has surprisingly little effect on downstream task performance after fine-tuning (on non-shuffled training data). It has recently been found that MLMs are quite robust to permuting downstream test data (Sinha et al., 2020; Pham et al., 2020; Gupta et al., 2021) and even do quite well using permuted "unnatural" downstream train data (Sinha et al., 2020; Gupta et al., 2021). Here, we show that downstream performance for "unnatural language pre-training" is much closer to standard MLM pre-training than one might expect.

In an effort to shed light on these findings, we

[1] One might even argue that BERT is not actually all that different from earlier distributional models like word2vec (Mikolov et al., 2013), see Appendix A.

experiment with various probing tasks. We verify via non-parametric probes that the permutations do in fact make the model worse at syntax-dependent tasks. However, just like on the downstream fine-tuning tasks, permuted models perform well on parametric syntactic probes, in some cases almost matching the unpermuted model's performance, which is quite surprising given how important word order is crosslinguistically (Greenberg 1963; Dryer 1992; Cinque 1999, i.a.).

Our results can be interpreted in different ways. One could argue that our downstream and probing tasks are flawed, and that we need to examine models with examples that truly test strong generalization and compositionality. Alternatively, one could argue that prior works have overstated the dependence of human language understanding on word order, and that human language understanding depends less on the structure of the sentence and more on the structure of the *world*, which can be inferred to a large extent from distributional information. This work is meant to deepen our understanding of MLM pre-training and, through this, move us closer to finding out what is actually required for adequately modelling natural language.

2 Related Work

Sensitivity to word order in NLU. Information order has been a topic of research in computational linguistics since Barzilay and Lee (2004) introduced the task of ranking sentence orders as an evaluation for language generation quality, an approach which was subsequently also used to evaluate readability and dialogue coherence (Barzilay and Lapata, 2008; Laban et al., 2021).

More recently, several research groups have investigated information order for words rather than sentences as an evaluation of model humanlikeness. Sinha et al. (2020) investigate the task of natural language inference (NLI) and find high accuracy on permuted examples for different Transformer and pre-Transformer era models, across English and Chinese datasets (Hu et al., 2020). Gupta et al. (2021) use targeted permutations on RoBERTa-based models and show word order insensitivity across natural language inference (MNLI), paraphrase detection (QQP) and sentiment analysis tasks (SST-2). Pham et al. (2020) show insensitivity on a larger set of tasks, including the entire GLUE benchmark, and find that certain tasks in GLUE, such as CoLA and RTE are more sensitive to permutations than others. Ettinger (2020) recently observed that BERT accuracy decreases for some word order perturbed examples, but not for others. In all these prior works, models were given access to normal word order at (pre-)training time, but not at fine-tuning or test time. It was not clear whether the model acquires enough information about word order during the fine-tuning step, or whether it is ingrained in the pre-trained model. In this work, we take these investigations a step further: we show that the word order information needed for downstream tasks does not need to be provided to the model during pre-training. Since models can learn whatever word order information they do need largely from fine-tuning alone, this likely suggests that our downstream tasks don't actually require much complex word order information in the first place (cf., Glavaš and Vulić 2021).

Randomization ablations. Random controls have been explored in a variety of prior work. Wieting and Kiela (2019) show that random sentence encoders are surprisingly powerful baselines. Gauthier and Levy (2019) use random sentence reordering to label some tasks as "syntax-light" making them more easily decodeable from images of the brain. Shen et al. (2021) show that entire layers of MLM transformers can be randomly initialized and kept frozen throughout training without detrimental effect and that those layers perform better on some probing tasks than their frozen counterparts. Models have been found to be surprisingly robust to randomizing or cutting syntactic tree structures they were hoped to rely on (Scheible and Schütze, 2013; Williams et al., 2018a), and randomly permuting attention weights often induces only minimal changes in output (Jain and Wallace, 2019). In computer vision, it is well known that certain architectures constitute good "deep image priors" for fine-tuning (Ulyanov et al., 2018) or pruning (Frankle et al., 2020), and that even randomly wired networks can perform well at image recognition (Xie et al., 2019). Here, we explore randomizing the data, rather than the model, to assess whether certain claims about which phenomena the model has learned are established in fact.

Synthetic pre-training. Kataoka et al. (2020) found that pre-training on synthetically generated fractals for image classification is a very strong prior for subsequent fine-tuning on real image data. In language modeling, Papadimitriou and Jurafsky (2020) train LSTMs (Hochreiter and Schmidhuber,

1997) on non-linguistic data with latent structure such as MIDI music or Java code provides better test performance on downstream tasks than a randomly initialized model. They observe that even when there is no vocabulary overlap among source and target languages, LSTM language models leverage the latent hierarchical structure of the input to obtain better performance than a random, Zipfian corpus of the same vocabulary.

On the utility of probing tasks. Many recent papers provide compelling evidence that BERT contains a surprising amount of syntax, semantics, and world knowledge (Giulianelli et al., 2018; Rogers et al., 2020; Lakretz et al., 2019). Many of these works involve diagnostic classifiers (Hupkes et al., 2018) or *parametric* probes, i.e. a function atop learned representations that is optimized to find linguistic information. How well the probe learns a given signal can be seen as a proxy for linguistic knowledge encoded in the representations. However, the community is divided on many aspects of probing (Belinkov, 2021) including how complex probes should be. Many prefer *simple* linear probes over the complex ones (Alain and Bengio, 2017; Hewitt and Manning, 2019; Hall Maudslay et al., 2020). However, complex probes with strong representational capacity are able to extract the most information from representations (Voita and Titov, 2020; Pimentel et al., 2020b; Hall Maudslay et al., 2020). Here, we follow Pimentel et al. (2020a) and use *both* simple (linear) and complex (non-linear) models, as well as "complex" tasks (dependency parsing). As an alternative to parametric probes, stimulus-based *non-parametric* probing (Linzen et al., 2016; Jumelet and Hupkes, 2018; Marvin and Linzen, 2018; Gulordava et al., 2018a; Warstadt et al., 2019a, 2020a,b; Ettinger, 2020; Lakretz et al., 2021) has been used to show that even without a learned probe, BERT can predict syntactic properties with high confidence (Goldberg, 2019; Wolf, 2019). We use this class of non-parametric probes to investigate RoBERTa's ability to learn word order during pre-training.

3 Approach

We first describe the data generation and evaluation methodology used in this paper. We use the RoBERTa (base) (Liu et al., 2019) MLM architecture, due to its relative computational efficiency and good downstream task performance. We expect that other variants of MLMs would provide similar insights, given their similar characteristics.

3.1 Models

In all of our experiments, we use the original 16GB BookWiki corpus (the Toronto Books Corpus, Zhu et al. 2015, plus English Wikipedia) from Liu et al. (2019).[2] We denote the model trained on the original, un-modified BookWiki corpus as $\mathcal{M}_{\text{N}}$ (for "natural"). We use two types of word order randomization methods: permuting words at the sentence level, and resampling words at the corpus level.

Sentence word order permutation. To investigate to what extent the performance of MLM pretraining is a consequence of distributional information, we construct a training corpus devoid of natural word order but preserving local distributional information. We construct word order-randomized versions of the BookWiki corpus, following the setup of Sinha et al. (2020). Concretely, given a sentence S containing N words, we permute the sentence using a seeded random function $\mathcal{F}_1$ such that no word can remain in its original position. In total, there exist $(N-1)!$ possible permutations of a given sentence. We randomly sample a single permutation per sentence, to keep the total dataset size similar to the original.

We extend the permutation function $\mathcal{F}_1$ to a function $\mathcal{F}_n$ that preserves n-gram information. Specifically, given a sentence S of length N and n-gram value n, we sample a starting position i for possible contiguous n-grams $\in \{0, N-n\}$ and convert the span $S[i, i+n]$ to a single token, to form $\hat{S}$, of length $\hat{N} = N-(n+1)$. We continue this process repeatedly (without using the previously created n-grams) until there exists no starting position for selecting a contiguous n-gram in $\hat{S}$. For example, given a sentence of length $N = 6$, $\mathcal{F}_4$ will first convert one span of 4 tokens into a word, to have $\hat{S}$ consisting of three tokens (one conjoined token of 4 contiguous words, and two leftover words). Then, the resulting sentence $\hat{S}$ is permuted using $\mathcal{F}_1$. We train RoBERTa models on four permutation variants of BookWiki corpus, $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, $\mathcal{M}_4$ for each n-gram value $\in \{1, 2, 3, 4\}$. More details on the process, along with the pseudo code and sample quality, are provided in Appendix B.

Corpus word order bootstrap resample. The above permutations preserve higher order distri-

[2]We release the pre-trained RoBERTa models used in our experiments through the FairSeq repository: https://github.com/pytorch/fairseq/tree/master/examples /shuffled_word_order.

butional information by keeping words from the same sentence together. However, we need a baseline to understand how a model would perform without such co-occurrence information. We construct a baseline, $\mathcal{M}_{\text{UG}}$, that captures word/subword information, without access to co-occurrence statistics. To construct $\mathcal{M}_{\text{UG}}$, we sample unigrams from BookWiki according to their frequencies, while also treating named entities as unigrams. We leverage Spacy (Honnibal et al., 2020)[3] to extract unigrams and named entities from the corpus, and construct $\mathcal{M}_{\text{UG}}$ by drawing words from this set according to their frequency. This allows us to construct $\mathcal{M}_{\text{UG}}$ such that it has exactly the same size as BookWiki but without any distributional (i.e. co-occurrence) information beyond the unigram frequency distribution. Our hypothesis is that any model pre-trained on this data will perform poorly, but it should provide a baseline for the limits on learning language of the inductive bias of the model in isolation.

Further baselines. To investigate what happens if a model has absolutely no notion of word order, we also experiment with pre-training RoBERTa on the original corpus without positional embeddings. Concretely, we modify the RoBERTa architecture to remove the positional embeddings from the computation graph, and then proceed to pre-train on the natural order BookWiki corpus. We denote this model $\mathcal{M}_{\text{NP}}$. Finally, we consider a randomly initialized RoBERTa model $\mathcal{M}_{\text{RI}}$ to observe the extent we can learn from each task with only the model's base inductive bias.

Pre-training details. Each model $\in \{\mathcal{M}_{\text{N}}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_{\text{UG}}, \mathcal{M}_{\text{NP}}\}$ is a RoBERTa-base model (12 layers, hidden size of 768, 12 attention heads, 125M parameters), trained for 100k updates using 8k batch-size, 20k warmup steps, and 0.0006 peak learning rate. These are identical hyperparameters to Liu et al. (2019), except for the number of warmup steps which we changed to 20k for improved training stability. Each model was trained using 64 GPUs for up to 72 hours each. We train three seeds for each data configuration. We validate all models on the public Wiki-103 validation set (see Appendix C). We use FairSeq (Ott et al., 2019) for the pre-training and fine-tuning experiments.

[3]https://spacy.io/

3.2 Fine-tuning tasks

We evaluate downstream performance using the General Language Understanding and Evaluation (GLUE) benchmark, the Paraphrase Adversaries from Word Scrambling (PAWS) dataset, and various parametric and non-parametric tasks (see §5).

GLUE. The GLUE (Wang et al., 2018) benchmark is a collection of 9 datasets for evaluating natural language understanding systems, of which we use Corpus of Linguistic Acceptability (CoLA, Warstadt et al., 2019b), Stanford Sentiment Treebank (SST, Socher et al., 2013), Microsoft Research Paragraph Corpus (MRPC, Dolan and Brockett, 2005), Quora Question Pairs (QQP)[4], Multi-Genre NLI (MNLI, Williams et al., 2018b), Question NLI (QNLI, Rajpurkar et al., 2016; Demszky et al., 2018), Recognizing Textual Entailment (RTE, Dagan et al., 2005; Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009). Pham et al. (2020) show the word order insensitivity of several GLUE tasks (QQP, SST-2), evaluated on public regularly pre-trained checkpoints.

PAWS. The PAWS task (Zhang et al., 2019) consists of predicting whether a given pair of sentences are paraphrases. This dataset contains both paraphrase and non-paraphrase pairs with high lexical overlap, which are generated by controlled word swapping and back translation. Since even a small word swap and perturbation can drastically modify the meaning of the sentence, we hypothesize the randomized pre-trained models will struggle to attain a high performance on PAWS.

Fine-tuning details. We use the same fine-tuning methodology used by Liu et al. (2019), where we run hyperparameter search over the learning rates $\{1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}\}$ and batch sizes $\{16, 32\}$ for each model. For the best hyperparam configurations of each model, we fine-tune with 5 different seeds and report the mean and standard deviation for each setting. $\mathcal{M}_{\text{NP}}$ is fine-tuned without positional embeddings, matching the way it was pre-trained.

4 Downstream task results

In this section, we present the downstream task performance of the models defined in §3. For evaluation, we report Matthews correlation for CoLA and accuracy for all other tasks.

[4]http://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

Model	QNLI	RTE	QQP	SST-2	MRPC	PAWS	MNLI-m/mm	CoLA
$\mathcal{M}_{\texttt{N}}$	92.45 +/- 0.2	73.62 +/- 3.1	91.25 +/- 0.1	93.75 +/- 0.4	89.09 +/- 0.9	94.49 +/- 0.2	86.08 +/- 0.2 / 85.4 +/- 0.2	52.45 +/- 21.2
$\mathcal{M}_4$	91.65 +/- 0.1	70.94 +/- 1.2	91.39 +/- 0.1	92.46 +/- 0.3	86.90 +/- 0.3	94.26 +/- 0.2	83.79 +/- 0.2 / 83.94 +/- 0.3	35.25 +/- 32.2
$\mathcal{M}_3$	91.56 +/- 0.4	69.75 +/- 2.8	91.22 +/- 0.1	91.97 +/- 0.5	86.22 +/- 0.8	94.03 +/- 0.1	83.83 +/- 0.2 / 83.71 +/- 0.1	40.78 +/- 23.0
$\mathcal{M}_2$	90.51 +/- 0.1	70.00 +/- 2.5	91.33 +/- 0.0	91.78 +/- 0.3	85.90 +/- 1.2	93.53 +/- 0.3	83.45 +/- 0.3 / 83.54 +/- 0.3	50.83 +/- 5.8
$\mathcal{M}_1$	89.05 +/- 0.2	68.48 +/- 2.5	91.01 +/- 0.0	90.41 +/- 0.4	86.06 +/- 0.8	89.69 +/- 0.6	82.64 +/- 0.1 / 82.67 +/- 0.2	31.08 +/- 10.0
$\mathcal{M}_{\texttt{NP}}$	77.59 +/- 0.3	54.78 +/- 2.2	87.78 +/- 0.4	83.21 +/- 0.6	72.78 +/- 1.6	57.22 +/- 1.2	63.35 +/- 0.4 / 63.63 +/- 0.2	2.37 +/- 3.2
$\mathcal{M}_{\texttt{UG}}$	66.94 +/- 9.2	53.70 +/- 1.0	85.57 +/- 0.1	83.17 +/- 1.5	70.57 +/- 0.7	58.59 +/- 0.3	71.93 +/- 0.2 / 71.33 +/- 0.5	0.92 +/- 2.1
$\mathcal{M}_{\texttt{RI}}$	62.17 +/- 0.4	52.97 +/- 0.2	81.53 +/- 0.2	82.0 +/- 0.7	70.32 +/- 1.5	56.62 +/- 0.0	65.70 +/- 0.2 / 65.75 +/- 0.3	8.06 +/- 1.6

Table 1: GLUE and PAWS-Wiki dev set results on different RoBERTa (base) models trained on variants of the BookWiki corpus (with mean and std). The top row is the original model, the middle half contains our primary models under investigation, and the bottom half contains the baselines.

4.1 Word order permuted pre-training

In our first set of experiments, we finetune the pre-trained models on the GLUE and PAWS tasks. We report the results in Table 1.[5] First, we observe that the model without access to distributional or word order information, $\mathcal{M}_{\texttt{UG}}$ (unigram) performs much worse than $\mathcal{M}_{\texttt{N}}$ overall: $\mathcal{M}_{\texttt{UG}}$ is 18 points worse than $\mathcal{M}_{\texttt{N}}$ on average across the accuracy-based tasks in Table 1 and has essentially no correlation with human judgments on CoLA. $\mathcal{M}_{\texttt{UG}}$ $\mathcal{M}_{\texttt{NP}}$ and $\mathcal{M}_{\texttt{RI}}$ perform comparably on most of the tasks, while achieving surprisingly high scores in QQP and SST-2. However, all three models perform significantly worse on GLUE and PAWS, compared to $\mathcal{M}_{\texttt{N}}$ (Table 1, bottom half). $\mathcal{M}_{\texttt{UG}}$ reaches up to 71.9 on MNLI - possibly due to the fact that $\mathcal{M}_{\texttt{UG}}$ has access to (bags of) words and some phrases (from NER) is beneficial for MNLI. For the majority of tasks, the difference between $\mathcal{M}_{\texttt{NP}}$ and $\mathcal{M}_{\texttt{RI}}$ is small - a pure bag of words model performs comparably to a randomly initialized model.

Next, we observe a significant improvement on all tasks when we give models access to sentence-level distributional information during pre-training. $\mathcal{M}_1$, the model pre-trained on completely shuffled sentences, is on average only 3.3 points lower than $\mathcal{M}_{\texttt{N}}$ on the accuracy-based tasks, and within 0.3 points of $\mathcal{M}_{\texttt{N}}$ on QQP. Even on PAWS, which was designed to require knowledge of word order, $\mathcal{M}_1$ is within 5 points of $\mathcal{M}_{\texttt{N}}$. Randomizing n-grams instead of words during pre-training results in a (mostly) smooth increase on these tasks: $\mathcal{M}_4$, the model pre-trained on shuffled 4-grams, trails $\mathcal{M}_{\texttt{N}}$ by only 1.3 points on average, and even comes within 0.2 points of $\mathcal{M}_{\texttt{N}}$ on PAWS. We observe a somewhat different pattern on CoLA, where $\mathcal{M}_2$ does almost as well as $\mathcal{M}_{\texttt{N}}$ and outperforms $\mathcal{M}_3$ and $\mathcal{M}_4$, though we also observe very high variance across random seeds for this task. Crucially, we observe that $\mathcal{M}_1$ outperforms $\mathcal{M}_{\texttt{NP}}$ by a large margin. This shows that positional embeddings are critical for learning, even when the word orders themselves are not natural.[6] Overall, these results confirm our hypothesis that RoBERTa's strong performance on downstream tasks can be explained for a large part by the distributional prior.

4.2 Word order permuted fine-tuning

There are two possible explanations for the results in §4.1: either the tasks do not need word order information to be solved, or any necessary word order information can be acquired during fine-tuning. To examine this question, we permute the word order during fine-tuning as well. Concretely, for each task, we construct a unigram order-randomized version of each example in the fine-tuning training set using $\mathcal{F}_1$. We then fine-tune our pre-trained models on this shuffled data and evaluate task performance. For all experiments, we evaluate and perform early stopping on the original, natural word order dev set, in order to conduct a fair evaluation on the exact same optimization setup for all models.

Our results in Figure 1 provide some evidence for both hypotheses. On QQP and QNLI, accuracy decreases only slightly for models fine-tuned on shuffled data. Models can also achieve above 80% accuracy on MNLI, SST-2, and MRPC when

[5]The $\mathcal{M}_{\texttt{N}}$ results are not directly comparable with that of publicly released `roberta-base` model by Liu et al. (2019), as that uses the significantly larger 160GB corpus, and is trained for 500K updates. For computational reasons, we restrict our experiments to the 16GB BookWiki corpus and 100K updates, mirroring the RoBERTa ablations.

[6]Recall, $\mathcal{M}_{\texttt{NP}}$ is fed natural sentences as $\mathcal{M}_{\texttt{N}}$ while not having the ability to learn positional embeddings. To further quantify the effect of positional embeddings, we also investigated the effect of shuffling the entire context window, to keep the co-occurrence information same as $\mathcal{M}_{\texttt{NP}}$ in Appendix D. We observed this model to be worse than $\mathcal{M}_1$ but significantly better than $\mathcal{M}_{\texttt{NP}}$ to support the claim about the importance of positional embeddings while training.

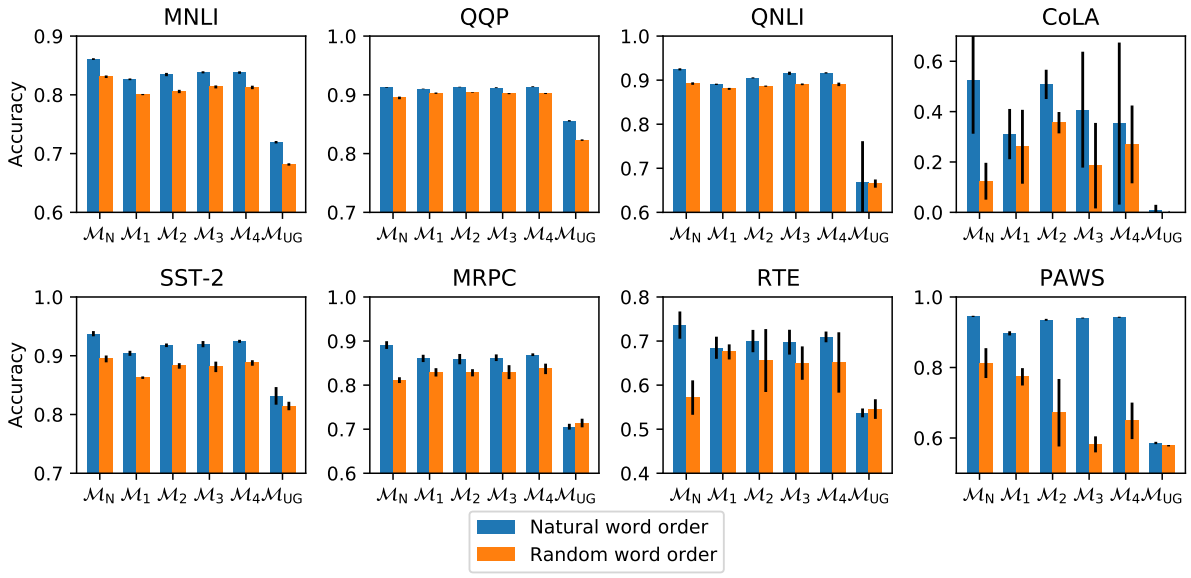

Figure 1: GLUE & PAWS task dev performance when finetuned on naturally (blue) and randomly ordered (orange) text, respectively, using pre-trained RoBERTa (base) models trained on different versions of BookWiki corpus.

fine-tuned on shuffled data, suggesting that purely lexical information is quite useful on its own.[7]

On the other hand, for all datasets besides QQP and QNLI, we see noticeable drops in accuracy when fine-tuning on shuffled data and testing on normal order, both for $\mathcal{M}_{\text{N}}$ and for shuffled models $\mathcal{M}_1$ through $\mathcal{M}_4$. This suggests both that word order information is useful for these tasks, and that shuffled models must be learning to use word order information during fine-tuning.[8] Having word order during fine-tuning is especially important for achieving high accuracy on CoLA, RTE (cf. Pham et al. 2020), as well as PAWS, suggesting that these tasks are the most word order reliant. Recent research (Yu and Ettinger, 2021) raised some questions about potential artefacts inflating performance on PAWS: their swapping-distance cue of appears consistent both with our finding of high PAWS performance for n-gram shuffled models in Table 1, and with our PAWS results in Figure 1, which suggests that PAWS performance does in fact rely to some extent on natural word order at the fine-tuning stage.

Finally, for CoLA, MRPC, and RTE, performance is higher after fine-tuning on shuffled data for $\mathcal{M}_1$ than $\mathcal{M}_{\text{N}}$. We hypothesize that $\mathcal{M}_{\text{N}}$ represents shuffled and non-shuffled sentences very differently, resulting in a domain mismatch problem when fine-tuning on shuffled data but evaluating on non-shuffled data.[9] Since $\mathcal{M}_1$ never learns to be sensitive to word order during pre-training or fine-tuning, it does not suffer from that issue. Our results in this section also highlights the issues with these datasets, concurrent to the findings that many GLUE tasks does not need sophisticated linguistic knowledge to solve, as models typically tend to exploit the statistical artefacts and spurious correlations during fine-tuning (cf. Gururangan et al. 2018; Poliak et al. 2018; Tsuchiya 2018; McCoy et al. 2019). However, our results overwhelmingly support the fact that word order does not matter during pre-training, if the model has the opportunity to learn the necessary information about word order during fine-tuning.

5 Probing results

To investigate how much syntactic information is contained in the MLM representations, we evaluate several probing tasks on our trained models. We consider two classes of probes: *parametric* probes, which make use of learnable parameters, and *non-parametric* probes, which directly exam-

[7] This finding is compatible with the observation of Gupta et al. (2021) and Sinha et al. (2020) who train on a randomized training corpus for MRPC, QQP, SST-2 and MNLI.

[8] We perform additional experiments on how the model representations change during fine-tuning for shuffled training using Risannen Data Analysis in Appendix I.

[9] We further study the domain mismatch problem by evaluating on shuffled data *after* fine-tuning on the shuffled data for models in Appendix F. We observe that models improves their scores on evaluation on shuffled data when the training data source is changed from natural to shuffled - highlighting domain match effect.

ine the language model's predictions.

5.1 Parametric Probing

To probe our models for syntactic, semantic and other linguistic properties, we investigate dependency parsing using Pareto probing (Pimentel et al., 2020a) and the probing tasks from Conneau et al. (2018) in SentEval (Conneau and Kiela, 2018).

5.1.1 Syntactic Probing

Pimentel et al. (2020a) proposed a framework based on Pareto optimality to probe for syntactic information in contextual representations. They suggest that an optimal probe should balance optimal performance on the probing task with the complexity of the probe. Following their setup, we use the "difficult" probe: dependency parsing (DEP). We also investigate the "easy" probes, dependency arc labeling (DAL) and POS tag prediction (POS), results are reported in Appendix K. We probe with Linear and MLP probes, and inspect the task accuracy in terms of Unlabeled Attachment Score (UAS). The dependency parsing probe used in Pimentel et al. (2020a) builds on the Biaffine Dependency Parser (Dozat and Manning, 2017), but with simple MLPs on top of the Transformer representations.[10]

Training setup. Similar to the setup by Pimentel et al. (2020a), we run 50 random hyperparameter searches on both MLP and Linear probes by uniformly sampling from the number of layers (0-5), dropout (0-0.5), log-uniform hidden size $[2^5, 2^{10}]$. We triple this experiment size by evaluating on three pre-trained models of different seeds for each model configuration. We consider Pimentel et al.'s English dataset, derived from Universal Dependencies EWT (UD EWT) (Bies et al., 2012; Silveira et al., 2014) which contains 12,543 training sentences. Additionally, we experiment on the Penn Treebank dataset (PTB), which contains 39,832 training sentences.[11] We report the mean test accuracy over three seeds for the best dev set accuracy for each task.[12]

[10] We experimented with a much stronger, state-of-the-art Second order Tree CRF Neural Dependency Parser (Zhang et al., 2020), but did not observe any difference in UAS with different pre-trained models (see Appendix G)

[11] PTB data (Kitaev et al., 2019) is used from github.com/nikitakit/self-attentive-parser/tree/master/data.

[12] Pimentel et al. (2020a) propose computing the *Pareto Hypervolume* over all hyperparameters in each task. We did not observe a significant difference in the hypervolumes for the models, as reported in Appendix K.

Model	UD EWT		PTB	
	MLP	Linear	MLP	Linear
$\mathcal{M}_{\text{N}}$	80.41 +/- 0.85	66.26 +/- 1.59	86.99 +/- 1.49	66.47 +/- 2.77
$\mathcal{M}_4$	78.04 +/- 2.06	65.61 +/- 1.99	85.62 +/- 1.09	66.49 +/- 2.02
$\mathcal{M}_3$	77.80 +/- 3.09	64.89 +/- 2.63	85.89 +/- 1.01	66.11 +/- 1.68
$\mathcal{M}_2$	78.22 +/- 0.88	64.96 +/- 2.32	84.72 +/- 0.55	64.69 +/- 2.50
$\mathcal{M}_1$	69.26 +/- 6.00	56.24 +/- 5.05	79.43 +/- 0.96	57.20 +/- 2.76
$\mathcal{M}_{\text{UG}}$	74.15 +/- 0.93	65.69 +/- 7.35	80.07 +/- 0.79	57.28 +/- 1.42

Table 2: Unlabeled Attachment Score (UAS) (mean and std) on the dependency parsing task (DEP) on two datasets, UD EWT and PTB, using the Pareto Probing framework (Pimentel et al., 2020a).

Results. We observe that the UAS scores follow a similar linear trend as the fine-tuning results in that $\mathcal{M}_1 \approx \mathcal{M}_{\text{UG}} < \mathcal{M}_2 < \mathcal{M}_3 < \mathcal{M}_4 < \mathcal{M}_{\text{N}}$ (Table 2). Surprisingly, $\mathcal{M}_{\text{UG}}$ probing scores seem to be somewhat better than $\mathcal{M}_1$ (though with large overlap in their standard deviations), even though $\mathcal{M}_{\text{UG}}$ cannot learn information related to either word order or co-occurrence patterns. The performance gap appears to be task- and probe specific. We observe a low performance gap in several scenarios, the lowest being between $\mathcal{M}_{\text{N}}$ vs. $\mathcal{M}_3/\mathcal{M}_4$, for PTB using the both MLP and Linear probes.

5.1.2 SentEval Probes

We also investigate the suite of 10 probing tasks (Conneau et al., 2018) available in the SentEval toolkit (Conneau and Kiela, 2018). This suite contains a range of semantic, syntactic and surface level tasks. Jawahar et al. (2019) utilize this set of probing tasks to arrive at the conclusion that "*BERT embeds a rich hierarchy of linguistic signals: surface information at the bottom, syntactic information in the middle, semantic information at the top*". We re-examine this hypothesis by using the same probing method and comparing against models trained with random word order.

Training setup. We run the probes on the final layer of each of our pre-trained models for three seeds, while keeping the encoder frozen. SentEval trains probes on top of fixed representations individually for each task. We follow the recommended setup and run grid search over the following hyperparams: number of hidden layer dimensions ($[0, 50, 100, 200]$), dropout ($[0, 0.1, 0.2]$), 4 epochs, 64 batch size. We select the best performance based on the dev set, and report the test set accuracy.

Results. We provide the results in Table 3. The $\mathcal{M}_{\text{N}}$ pre-trained model scores better than the unnatural word order models for only one out of five semantic tasks and in none of the lexical tasks.

Model	Length (Surface)	WordContent (Surface)	TreeDepth (Syntactic)	TopConstituents (Syntactic)	BigramShift (Syntactic)	Tense (Semantic)	SubjNumber (Semantic)	ObjNumber (Semantic)	OddManOut (Semantic)	CoordInversion (Semantic)
$\mathcal{M}_{\text{N}}$	78.92 +/- 1.91	31.83 +/- 1.75	35.97 +/- 1.38	**78.26** +/- 4.08	**81.82** +/- 0.55	87.83 +/- 0.51	85.05 +/- 1.23	75.94 +/- 0.68	58.40 +/- 0.33	**70.87** +/- 2.46
$\mathcal{M}_4$	92.88 +/- 0.15	57.78 +/- 0.36	40.05 +/- 0.29	72.50 +/- 0.51	76.12 +/- 0.29	88.32 +/- 0.13	**85.65** +/- 0.13	82.95 +/- 0.05	**58.89** +/- 0.30	61.31 +/- 0.19
$\mathcal{M}_3$	91.52 +/- 0.16	48.81 +/- 0.26	38.63 +/- 0.61	70.29 +/- 0.31	77.36 +/- 0.12	86.74 +/- 0.12	83.83 +/- 0.38	80.99 +/- 0.26	57.01 +/- 0.21	60.00 +/- 0.26
$\mathcal{M}_2$	**93.54** +/- 0.29	62.52 +/- 0.21	**41.40** +/- 0.32	74.31 +/- 0.29	75.44 +/- 0.14	**87.91** +/- 0.35	84.88 +/- 0.11	83.98 +/- 0.14	57.60 +/- 0.36	59.46 +/- 0.37
$\mathcal{M}_1$	88.33 +/- 0.14	**64.03** +/- 0.34	40.24 +/- 0.20	70.94 +/- 0.38	58.37 +/- 0.40	87.88 +/- 0.08	83.49 +/- 0.12	**83.44** +/- 0.06	56.51 +/- 0.26	56.98 +/- 0.50
$\mathcal{M}_{\text{UG}}$	86.69 +/- 0.33	36.60 +/- 0.33	32.53 +/- 0.76	61.54 +/- 0.60	57.42 +/- 0.04	68.45 +/- 0.23	71.25 +/- 0.12	66.63 +/- 0.21	50.06 +/- 0.40	56.26 +/- 0.17

Table 3: SentEval Probing (Conneau et al., 2018; Conneau and Kiela, 2018) results (with mean and std) on different model variants.

However, $\mathcal{M}_{\text{N}}$ does score higher for two out of three syntactic tasks. Even for these two syntactic tasks, the gap among $\mathcal{M}_{\text{UG}}$ and $\mathcal{M}_{\text{N}}$ is much higher than $\mathcal{M}_1$ and $\mathcal{M}_{\text{N}}$. These results show that while natural word order is useful for at least some probing tasks, the distributional prior of randomized models alone is enough to achieve a reasonably high accuracy on syntax sensitive probing.

5.2 Non-Parametric Probing

How to probe effectively with parametric probes is a matter of much recent debate (Hall Maudslay et al., 2020; Belinkov, 2021). From our results so far, it is unclear whether parametric probing meaningfully distinguishes models trained with corrupted word order from those trained with normal orders. Thus, we also investigate non-parametric probes (Linzen et al., 2016; Marvin and Linzen, 2018; Gulordava et al., 2018b) using the formulation of Goldberg (2019) and Wolf (2019).

We consider a set of non-parametric probes that use a range of sentences varying in their linguistic properties. For each, the objective is for a pre-trained model to provide higher probability to a grammatically correct word than to an incorrect one. Since both the correct and incorrect options occupy the same sentential position, we call them "focus words". Linzen et al. (2016) use sentences from Wikipedia containing present-tense verbs, and compare the probability assigned by the encoder to plural vs. singular forms of the verb; they focus on sentences containing at least one noun between the verb and its subject, known as "agreement attractors." Gulordava et al. (2018b) instead replace focus words with random substitutes from the same part-of-speech and inflection. Finally, Marvin and Linzen (2018) construct minimal pairs of grammatical and ungrammatical sentences, and compare the model's probability for the words that differ.

Setup. In our experiments, we mask the focus words in the stimuli and compute the probability of the correct and incorrect token respectively. To handle Byte-Pair Encoding (BPE), we use the WordPiece (Wu et al., 2016) tokens prepended with a space. We observe that the Linzen et al. (2016) and Gulordava et al. (2018b) datasets are skewed towards singular focus words, which could disproportionately help weaker models that just happen to assign more probability mass to singular focus words. To counter this, we balance these datasets to have an equal number of singular and plural focus words by upsampling, and report the aggregated and balanced results in Table 4 (see Appendix L for more detailed results). We verify our experiments by using three pre-trained models with different seeds for each model configuration.

Model	Linzen et al. (2016) *	Gulordava et al. (2018b) *	Marvin and Linzen (2018)
$\mathcal{M}_{\text{N}}$	91.17 +/- 2.6	68.66 +/- 11.6	88.05 +/- 6.5
$\mathcal{M}_4$	66.93 +/- 3.2	69.47 +/- 4.9	70.66 +/- 12.5
$\mathcal{M}_3$	64.60 +/- 2.7	66.10 +/- 5.9	73.82 +/- 15.7
$\mathcal{M}_2$	61.27 +/- 3.1	60.20 +/- 7.6	73.95 +/- 14.3
$\mathcal{M}_1$	58.96 +/- 1.8	68.10 +/- 14.4	70.69 +/- 11.6
$\mathcal{M}_{\text{UG}}$	65.36 +/- 7.1	60.88 +/- 24.3	50.10 +/- 0.2

Table 4: Mean (and std) non-parametric probing accuracy on different datasets. * indicates rebalanced datasets, see Appendix L for more details.

Results. We observe for the Linzen et al. (2016) and Marvin and Linzen (2018) datasets that the gap between the $\mathcal{M}_{\text{N}}$ and randomization models is relatively large. The Gulordava et al. (2018b) dataset shows a smaller gap between $\mathcal{M}_{\text{N}}$ and the randomization models. While some randomization models (e.g., $\mathcal{M}_2$, $\mathcal{M}_3$, and $\mathcal{M}_4$) performed quite similarly to $\mathcal{M}_{\text{N}}$ according to the parametric probes, they all are markedly worse than $\mathcal{M}_{\text{N}}$ according to the non-parametric ones. This suggests that non-parametric probes identify certain syntax-related modeling failures that parametric ones do not.

6 Discussion

The assumption that word order information is crucial for any classical NLP pipeline (especially for English) is deeply ingrained in our understanding of syntax itself (Chomsky, 1957): without order, many linguistic constructs are undefined. Our fine-

tuning results in §4.1 and parametric probing results in §5.1, however, suggests that MLMs do not need to rely much on word order to achieve high accuracy, bringing into question previous claims that they learn a "classical NLP pipeline."

One might ask, though, whether an NLP pipeline would really need natural word order at all: can transformers not simply learn what the correct word order is from unordered text? First, the lower non-parametric probing accuracies of the randomized models indicate that they are not able to accurately reconstruct the original word order (see also Appendix D). But even if models were able to "unshuffle" the words under our unnatural pre-training set up, they would only be doing so based on distributional information. Models would then abductively learn only the most likely word order. While models might infer a distribution over possible orders and use that information to structure their representations (Papadimitriou et al., 2021), syntax is not about *possible* or even *the most likely* orders: it is about the *actual* order. That is, even if one concludes in the end that Transformers are able to perform word order reconstruction based on distributional information, and recover almost all downstream performance based solely on that, we ought to be a lot more careful when making claims about what our evaluation datasets are telling us.

Thus, our results seem to suggest that we may need to revisit what we mean by "linguistic structure," and perhaps subsequently acknowledge that we may not need human-like linguistic abilities for most NLP tasks. Or, our results can be interpreted as evidence that we need to develop more challenging and more comprehensive evaluations, if we genuinely want to measure linguistic abilities, however those are defined, in NLP models.

There are many interesting and potentially exciting avenues for future work that we could not explore due to limitation of space. An interesting question revolves around whether this phenomenon is more pronounced for English than for other languages. It is natural to wonder whether more word-order flexible or morphologically-rich languages would suffer from the same problem. Using the methods discussed in this work, we could imagine devising a way to determine the degree of order-dependence for tasks across languages. Another possible extension pertains to other tasks, including extractive question answering (QA) or sequence tagging, for which we can also to determine whether word order information is acquired downstream or during pre-training.

The sensitivity of generative models to word order permuted input could also be investigated further. Recent work by Parthasarathi et al. (2021) begins this discussion, by showing that a Machine Translation (MT) model can often arrive at the gold source translation when provided with input sentences that have had their words permuted using parse trees. Relatedly, Alleman et al. (2021) also investigates targeted parse-tree-based perturbations as a means of evaluating model robustness. O'Connor and Andreas (2021) also demonstrate the insensitivity of Transformers towards syntax manipulations while achieving low perplexity in language modeling tasks. Exploring model sensitivity to word order permutations for approaches that unify generation and classification (e.g., multi-tasking) could also be interesting future work.

7 Conclusion

In this work, we revisited the hypothesis that masked language modelling's impressive performance can be explained in part by its ability to learn classical NLP pipelines. We investigated targeted pre-training on sentences with various degrees of randomization in their word order, and observed overwhelmingly that MLM's success is most likely not due to its ability to discover syntactic and semantic mechanisms necessary for a traditional language processing pipeline during pre-training. Instead, our experiments suggest that MLM's success can largely be explained by it having learned higher-order distributional statistics that make for a useful prior for subsequent fine-tuning. These results should hopefully encourage the development of better, more challenging tasks that require sophisticated reasoning, and harder probes to narrow down what exact linguistic information is present in the representations learned by our models.

Acknowledgements

We thank Tiago Pimentel, Shruti Bhosale, Naman Goyal, Shagun Sodhani, Sylke Gosen, Prasanna Parasarathi, Kyunghyun Cho, Mona Diab, Brenden Lake, Myle Ott, Ethan Perez, and Mike Lewis for their help in resolving technical doubts during experimentation and/or feedback on an earlier draft. We also thank the anonymous reviewers for their constructive feedback during the reviewing phase, which helped polish the paper to its current state.

References

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *ICLR 2017, Workshop Track Proceedings*. OpenReview.net.

Matteo Alleman, Jonathan Mamou, Miguel A Del Rio, Hanlin Tang, Yoon Kim, and SueYeon Chung. 2021. Syntactic perturbations reveal representational correlates of hierarchical phrase structure in pretrained language models. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 263–276, Online. Association for Computational Linguistics.

Trapit Bansal, Rishikesh Jha, and Andrew McCallum. 2020. Learning to few-shot learn across diverse natural language classification tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5108–5123, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 113–120, Boston, Massachusetts, USA. Association for Computational Linguistics.

Yonatan Belinkov. 2021. Probing classifiers: Promises, shortcomings, and alternatives. *arXiv preprint arXiv:2102.12452*.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *TAC*.

Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English web treebank. *Linguistic Data Consortium, Philadelphia, PA*.

Noam Chomsky. 1957. *Syntactic structures*. Walter de Gruyter.

Guglielmo Cinque. 1999. *Adverbs and functional heads: A cross-linguistic perspective*. Oxford University Press on Demand.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *arXiv preprint arXiv:1809.02922*.

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.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

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

Matthew S Dryer. 1992. The Greenbergian word order correlations. *Language*, pages 81–138.

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

Jonathan Frankle, David J Schwab, and Ari S Morcos. 2020. Training batchnorm and only batchnorm: On the expressive power of random features in cnns. *arXiv preprint arXiv:2003.00152*.

Jon Gauthier and Roger Levy. 2019. Linking artificial and human neural representations of language. 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 529–539, Hong Kong, China. Association for Computational Linguistics.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9.

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. 2018. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 240–248, Brussels, Belgium. Association for Computational Linguistics.

Goran Glavaš and Ivan Vulić. 2021. Is supervised syntactic parsing beneficial for language understanding tasks? an empirical investigation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3090–3104, Online. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing BERT's Syntactic Abilities. *CoRR*, page 4.

Joseph Greenberg. 1963. Some universals of grammar with particular reference to the order of meaningful elements. *In J. Greenberg, ed., Universals of Language. 73-113. Cambridge, MA.*

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018a. 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.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018b. Colorless green recurrent networks dream hierarchically. *arXiv:1803.11138 [cs]*.

Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. 2021. Bert & family eat word salad: Experiments with text understanding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12946–12954.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

R Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.

Rowan Hall Maudslay, Josef Valvoda, Tiago Pimentel, Adina Williams, and Ryan Cotterell. 2020. A tale of a probe and a parser. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7389–7395, Online. Association for Computational Linguistics.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

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.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kübler, and Lawrence Moss. 2020. OCNLI: Original Chinese Natural Language Inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3512–3526, Online. 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.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. 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 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What Does BERT Learn about the Structure of Language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Jaap Jumelet and Dieuwke Hupkes. 2018. Do language models understand anything? on the ability of LSTMs to understand negative polarity items. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 222–231, Brussels, Belgium. Association for Computational Linguistics.

Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Eisuke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh. 2020. Pre-training without Natural Images. In *Proceedings of the Asian Conference on Computer Vision*.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

Philippe Laban, Luke Dai, Lucas Bandarkar, and Marti A. Hearst. 2021. Can transformer models measure coherence in text: Re-thinking the shuffle test. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1058–1064, Online. Association for Computational Linguistics.

Yair Lakretz, Dieuwke Hupkes, Alessandra Vergallito, Marco Marelli, Marco Baroni, and Stanislas Dehaene. 2021. Mechanisms for handling nested dependencies in neural-network language models and humans. *Cognition*, page 104699.

Yair Lakretz, German Kruszewski, Theo Desbordes, Dieuwke Hupkes, Stanislas Dehaene, and Marco Baroni. 2019. The emergence of number and syntax units in LSTM language models. 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 11–20, Minneapolis, Minnesota. Association for Computational Linguistics.

Thomas K Landauer and Susan T Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

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

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. *arXiv:1907.11692 [cs]*.

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.

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.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Joe O'Connor and Jacob Andreas. 2021. What context features can transformer language models use?

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Isabel Papadimitriou, Ethan A. Chi, Richard Futrell, and Kyle Mahowald. 2021. Deep subjecthood: Higher-order grammatical features in multilingual BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2522–2532, Online. Association for Computational Linguistics.

Isabel Papadimitriou and Dan Jurafsky. 2020. Learning Music Helps You Read: Using transfer to study linguistic structure in language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6829–6839, Online. Association for Computational Linguistics.

Prasanna Parthasarathi, Koustuv Sinha, Joelle Pineau, and Adina Williams. 2021. Sometimes we want ungrammatical translations. In *Findings of the Association for Computational Linguistics: Empirical Methods in Natural Language Processing (EMNLP)*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. Rissanen Data Analysis: Examining Dataset Characteristics via Description Length. In *Proceedings of the Thirty-eighth International Conference on Machine Learning (ICML)*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Thang M. Pham, Trung Bui, Long Mai, and Anh Nguyen. 2020. Out of Order: How important is the sequential order of words in a sentence in Natural Language Understanding tasks? *arXiv:2012.15180 [cs]*.

Tiago Pimentel, Naomi Saphra, Adina Williams, and Ryan Cotterell. 2020a. Pareto probing: Trading off accuracy for complexity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3138–3153, Online. Association for Computational Linguistics.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020b. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Jorma Rissanen. 1984. Universal coding, information, prediction, and estimation. *IEEE Transactions on Information theory*, 30(4):629–636.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

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.

Christian Scheible and Hinrich Schütze. 2013. Cutting recursive autoencoder trees. In *1st International Conference on Learning Representations (ICLR) Scottsdale, Arizona, USA, May 2-4, 2013, Conference Track Proceedings*.

Sheng Shen, Alexei Baevski, Ari Morcos, Kurt Keutzer, Michael Auli, and Douwe Kiela. 2021. Reservoir transformers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4294–4309, Online. Association for Computational Linguistics.

Natalia Silveira, Timothy Dozat, Marie-Catherine De Marneffe, Samuel R Bowman, Miriam Connor, John Bauer, and Christopher D Manning. 2014. A gold standard dependency corpus for english. In *LREC*, pages 2897–2904. Citeseer.

Koustuv Sinha, Prasanna Parthasarathi, Joelle Pineau, and Adina Williams. 2020. Unnatural Language Inference. *arXiv:2101.00010 [cs]*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

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.

Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. 2018. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018.

GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, Sheng-Fu Wang, Jason Phang, Anhad Mohananey, Phu Mon Htut, Paloma Jeretic, and Samuel R. Bowman. 2019a. Investigating BERT's knowledge of language: Five analysis methods with NPIs. 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 2877–2887, Hong Kong, China. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020a. 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.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019b. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020b. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics.

John Wieting and Douwe Kiela. 2019. No training required: Exploring random encoders for sentence classification. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Adina Williams, Andrew Drozdov, and Samuel R Bowman. 2018a. Do latent tree learning models identify meaningful structure in sentences? *Transactions of the Association for Computational Linguistics*, 6:253–267.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018b. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf. 2019. Some additional experiments extending the tech report "Assessing BERT's Syntactic Abilities" by Yoav Goldberg. page 7.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Saining Xie, Alexander Kirillov, Ross Girshick, and Kaiming He. 2019. Exploring randomly wired neural networks for image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1284–1293.

Lang Yu and Allyson Ettinger. 2021. On the interplay between fine-tuning and composition in transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2279–2293, Online. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021. Revisiting few-sample BERT fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yu Zhang, Zhenghua Li, and Min Zhang. 2020. Efficient second-order TreeCRF for neural dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3295–3305, Online. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. 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 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A From Word2vec to BERT in 4 steps

Take the basic parameterization of skipgram word2vec (Mikolov et al., 2013):

$$p(t \mid w; \theta) = \frac{e^{f(t,w)}}{\sum_{t' \in V} e^{f(t',w)}} \quad (1)$$

where t is the target, w is a word in the context, V is the set of all possible context words and f is simply the dot product.

In actual word2vec, we would use negative sampling within a given window size and optimize $\log \sigma(w \cdot t) + k \cdot \mathbb{E}_{t' \in P} \log \sigma(-w \cdot t')$ computed over context $C(w) = \{w_{i-k}, ..., w_{i-1}, w_{i+1}, w_{i+k}\}$ for word index i, window size $2k$ and unigram probability distribution P. It has been shown that optimizing this objective is close to learning the shifted PPMI distribution (Levy et al., 2015).

Step 1: BPE One reason for not computing the full softmax is that it becomes a prohibitively expensive matrix multiplication with large vocabulary V. A solution is to tokenize based on subword units, e.g. BPE, to ensure a smaller total vocabulary U in the softmax denominator. Doing so makes the matrix multiplication feasible, at least on GPU. It also ensures we have sufficient coverage over the words in our vocabulary.

Step 2: Defenestration Next, replace the local context window with the entire sentence, while masking out the target word, i.e., $C(t) = \{w \in S : w \neq t\}$ where S is the sentence containing w.

Step 3: Non-linearity Replace the pairwise word-level dot product $f(w, t)$ with a fancy non-linear function, say a sequence of multi-head self attention layers, $g(t, C(t))$, that takes as input the entire sentence-with-mask, and you get:

$$p(t \mid C(t); \theta) = \frac{e^{g(t,C(t))}}{\sum_{t' \in U} e^{g(t',C(t))}}$$

Step 4: Sprinkle data and compute You have BERT. Now all you need is enough data and compute, and perhaps some optimization tricks. Make sure to update the parameters in your model g when fine-tuning, rather than keeping them fixed, for optimal performance on downstream tasks.

This correspondence is probably (hopefully) trivial to most NLP researchers, but worth pointing out, lest we forget.

	BLEU-2	BLEU-3	BLEU-4
$\mathcal{M}_1$	0.493 +/- 0.12	0.177 +/- 0.16	0.040 +/- 0.11
$\mathcal{M}_2$	0.754 +/- 0.07	0.432 +/- 0.18	0.226 +/- 0.19
$\mathcal{M}_3$	0.824 +/- 0.06	0.650 +/- 0.09	0.405 +/- 0.20
$\mathcal{M}_4$	0.811 +/- 0.08	0.671 +/- 0.11	0.553 +/- 0.12

Table 5: BLEU-2,3,4 scores (mean and std dev) on a sample of 1M sentences drawn from the corpus used to train $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$and $\mathcal{M}_4$ compared to $\mathcal{M}_N$.

B Data generation

We provide pseudo-code for $\mathcal{F}_i$ in Algorithm 1. Following Sinha et al. (2020), we do not explicitly control whether the permuted words maintain any of their original neighbors. Thus, a certain amount of extra n-grams are expected to co-occur, purely as a product of random shuffling. We quantify the amount of such shuffling on a sample of 1 million sentences drawn from the BookWiki random corpus, and present the BLEU-2, BLEU-3 and BLEU-4 scores in Table 5. We provide a sample snapshot of the generated data in Table 18.

Algorithm 1 SentenceRandomizer

procedure $\mathcal{F}(S, t, n)$ ▷ Randomize a sentence S with seed t and n grams n
  W = tokenize the words in S
  Set the seed to t
  if $n > 1$ **then**
    while True **do**
      K = Sample all possible starting points from $[0, |W| - n]$
      Ignore the starting points in K which overlap with conjoined tokens ▷ Conjoined tokens consists of joined unigrams
      if $|K| \geq 1$ **then**
        Sample one position $p \in K$
        g = Extract the n-gram $W[p : p + n]$
        Delete $W[p + 1 : p + n]$
        $W[p]$ = Convert g to a conjoined token
      else
        Break from While loop
  while True **do**
    $\hat{W}$ = randomly shuffle tokens in W
    $r = \sum(\hat{W}[i] = W[i])$ ▷ Count number of positions where the token remains in its original position
    if $r = 0$ **then** Break out of While loop
  $\hat{S}$ = join the tokens in $\hat{W}$
  Return $\hat{S}$

C Pre-training details

We use the Fairseq (Ott et al., 2019) toolkit to pretrain RoBERTa (base) models on the different variants of the BookWiki corpus. We follow the default parameters as reported in Liu et al. (2019), with the following adjustments: max steps 100k, warmup

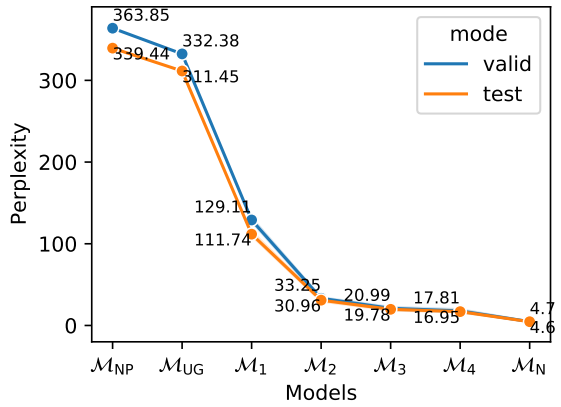

Figure 2: Perplexity of various models on Wiki 103 valid and test sets.

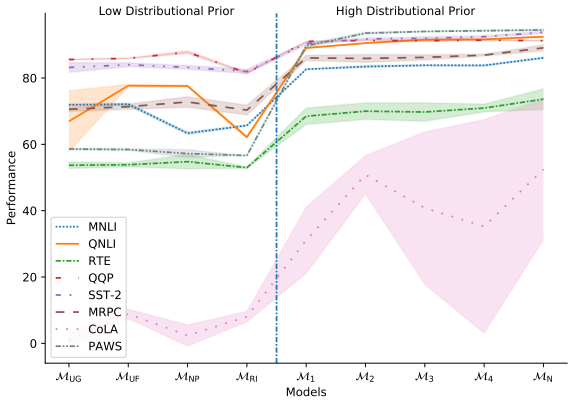

Figure 3: GLUE results on various model ablations using BookWiki corpus.

steps: 20k. We use the Wiki 103 validation and test set to validate and test the array of pre-trained models, as validation on this small dataset is quick, effective, and reproducible for comparison among publicly available datasets (Figure 2). We observe that perplexity monotonically increases from $\mathcal{M}_{\text{N}}$, through $\mathcal{M}_4$–$\mathcal{M}_1$, to $\mathcal{M}_{\text{UG}}$, and finally $\mathcal{M}_{\text{NP}}$.

D Word-order pre-training ablations

We also train further model ablations with low to high distributional priors. Following the construction of the corpus bootstrap resample, we train a model where words are drawn uniformly from BookWiki corpus, thus destroying the natural frequency distribution ($\mathcal{M}_{\text{UF}}$). We further study an ablation for a high distributional prior, $\mathcal{M}_{512}$, where we shuffle words (unigram) in a buffer created with joining multiple sentences such that maximum token length of the buffer is 512. This ablation—which is similar to the paragraph word shuffle condition in Gauthier and Levy (2019)—will allow us to study the effect of unigram shuffling in a window larger than the one for $\mathcal{M}_1$. Buffer size is chosen to be 512 because BERT/RoBERTa is typically trained with that maximum sequence length.

We observe dev set results on the GLUE benchmark of these ablations, along with baselines $\mathcal{M}_{\text{UG}}$, $\mathcal{M}_{\text{RI}}$ and $\mathcal{M}_{\text{NP}}$ and random shuffles in Table 6 and Figure 3. We observe that $\mathcal{M}_{512}$ exhibits worse overall scores than $\mathcal{M}_1$, however it is still significantly better than $\mathcal{M}_{\text{NP}}$ or $\mathcal{M}_{\text{UG}}$ baselines. We observe that destroying the natural frequency distribution of words ($\mathcal{M}_{\text{UF}}$) yields comparable or slightly better results compared to random corpus model $\mathcal{M}_{\text{UG}}$. This result shows that merely replicating the natural distribution of words without any context is not useful for the model to learn. These results indicate that at least some form of distributional prior is required for MLM-based models to learn a good downstream representation.

One might argue that the superior results displayed by the unnatural models is due to the ability of RoBERTa to "reconstruct" the natural word order from shuffled sentences. The data generation algorithm, $\mathcal{F}_i$ requires a seed t for every sentence. In our experiments, we had set the same seed for every sentence in the corpus to ensure reproducibility. However, it could be problematic if the sentences of the same length are permuted with the same seed, which could be easier for the model to "reconstruct" the natural word order to learn the necessary syntax. We tested this hypothesis by constructing a new corpus with different seeds for every sentence in every shard in the corpus (1/5th of BookWiki corpus is typically referred to as a *shard* for computational purposes), to build the model $\mathcal{M}_1{}^*$. We observe that there is minimal difference in the raw numbers among $\mathcal{M}_1$ and $\mathcal{M}_1{}^*$ for most of the tasks (Table 7) (with the exception of CoLA which performs similar to $\mathcal{M}_2$ possibly due to a difference in initialization). This result consequently proves that even with same seed, it is difficult for the model to just reconstruct the unnatural sentences during pre-training.

E Measuring Relative difference

In this section, we further measure the difference in downstream task performance reported in §4.1 using as a metric the *relative difference*. Let us denote the downstream task performance as $\mathcal{A}(\mathcal{T}|D)$, where $\mathcal{T}$ is the task and D is the pre-trained model. We primarily aim to evaluate the relative perfor-

Model	QNLI	RTE	QQP	SST-2	MRPC	PAWS	MNLI-m/mm	CoLA
$\mathcal{M}_{\mathtt{N}}$	92.45 +/- 0.2	73.62 +/- 3.1	91.25 +/- 0.1	93.75 +/- 0.4	89.09 +/- 0.9	94.49 +/- 0.2	86.08 +/- 0.2 / 85.4 +/- 0.2	52.45 +/- 21.2
$\mathcal{M}_4$	91.65 +/- 0.1	70.94 +/- 1.2	91.39 +/- 0.1	92.46 +/- 0.3	86.90 +/- 0.3	94.26 +/- 0.2	83.79 +/- 0.2 / 83.94 +/- 0.3	35.25 +/- 32.2
$\mathcal{M}_3$	91.56 +/- 0.4	69.75 +/- 2.8	91.22 +/- 0.1	91.97 +/- 0.5	86.22 +/- 0.8	94.03 +/- 0.1	83.83 +/- 0.2 / 83.71 +/- 0.1	40.78 +/- 23.0
$\mathcal{M}_2$	90.51 +/- 0.1	70.00 +/- 2.5	91.33 +/- 0.0	91.78 +/- 0.3	85.90 +/- 1.2	93.53 +/- 0.3	83.45 +/- 0.3 / 83.54 +/- 0.3	50.83 +/- 5.80
$\mathcal{M}_1$	89.05 +/- 0.2	68.48 +/- 2.5	91.01 +/- 0.0	90.41 +/- 0.4	86.06 +/- 0.8	89.69 +/- 0.6	82.64 +/- 0.1 / 82.67 +/- 0.2	31.08 +/- 10.0
$\mathcal{M}_{\mathtt{512}}$	84.97 +/- 0.3	56.09 +/- 0.6	90.15 +/- 0.1	86.11 +/- 0.7	79.41 +/- 0.6	77.3 +/- 12.63	77.58 +/- 0.3 / 77.89 +/- 0.4	12.54 +/- 5.57
$\mathcal{M}_{\mathtt{NP}}$	77.59 +/- 0.3	54.78 +/- 2.2	87.78 +/- 0.4	83.21 +/- 0.6	72.78 +/- 1.6	57.22 +/- 1.2	63.35 +/- 0.4 / 63.63 +/- 0.2	2.37 +/- 3.20
$\mathcal{M}_{\mathtt{UF}}$	77.69 +/- 0.4	53.84 +/- 0.6	85.92 +/- 0.1	84.00 +/- 0.6	71.35 +/- 0.8	58.43 +/- 0.3	72.10 +/- 0.4 / 72.58 +/- 0.4	8.89 +/- 1.40
$\mathcal{M}_{\mathtt{UG}}$	66.94 +/- 9.2	53.70 +/- 1.0	85.57 +/- 0.1	83.17 +/- 1.5	70.57 +/- 0.7	58.59 +/- 0.3	71.93 +/- 0.2 / 71.33 +/- 0.5	0.92 +/- 2.10
$\mathcal{M}_{\mathtt{RI}}$	62.17 +/- 0.4	52.97 +/- 0.2	81.53 +/- 0.2	82.0 +/- 0.7	70.32 +/- 1.5	56.62 +/- 0.0	65.70 +/- 0.2 / 65.75 +/- 0.3	8.06 +/- 1.60

Table 6: GLUE and PAWS-Wiki dev set results on different ablations of the RoBERTa (base) models, trained on variants of the BookWiki corpus (with mean and std dev). The top row is the original model, the middle half contains the sentence randomization models, and the bottom half contains the ablations.

Model	RTE	MRPC	SST-2	CoLA	QQP	QNLI	MNLI	PAWS
$\mathcal{M}_1$	68.48	85.97	90.41	31.07	91.01	89.05	82.64	89.69
$\mathcal{M}_1{}^*$	68.41	85.75	90.17	50.14	91.02	89.50	82.92	91.99

Table 7: Reconstruction experiments on shuffled word order sentences by fixing the same seed for every sentence ($\mathcal{M}_1$) and having different seed for different shards of the corpus ($\mathcal{M}_1{}^*$). We observe minimal difference in the downstream GLUE and PAWS scores.

Model	QNLI	RTE	QQP	SST-2	MRPC	CoLA	PAWS	MNLI
$\mathcal{M}_1$	3.70	7.04	0.26	3.58	3.42	40.74	5.12	3.62
$\mathcal{M}_2$	2.11	4.95	-0.09	2.12	3.61	3.09	9.06	2.63
$\mathcal{M}_3$	0.97	5.30	0.03	1.91	3.24	22.25	0.49	2.31
$\mathcal{M}_4$	0.87	3.67	-0.15	1.39	2.47	32.79	0.25	2.19
$\mathcal{M}_{\mathtt{UG}}$	27.74	27.25	6.26	11.35	20.91	98.24	38.20	16.56
$\mathcal{M}_{\mathtt{NP}}$	16.16	25.77	3.83	11.30	18.42	95.48	39.66	26.10

Table 8: $\Delta_{\{D_i\}}(\mathcal{T})$, scaled by a factor of 100 for GLUE and PAWS tasks.

mance gap, i.e. how much the performance differs between our natural and unnatural models. Thus, we define the *Relative Difference* ($\Delta_{\{D\}}(\mathcal{T})$):

$$\Delta_{\{D\}}(\mathcal{T}) = \frac{\mathcal{A}(\mathcal{T}|OR) - \mathcal{A}(\mathcal{T}|D))}{\mathcal{A}(\mathcal{T}|OR) - \mathcal{A}(\mathcal{T}|\emptyset)}, \quad (2)$$

where $\mathcal{A}(\mathcal{T}|\emptyset)$ is the random performance on the task $\mathcal{T}$ (0.33 for MNLI, 0 for CoLA, and 0.5 for rest) $\Delta_{\{D\}}(\mathcal{T}) \rightarrow 0$ when the performance of a pre-trained model reaches that of the pre-trained model trained with natural word order.

We observe the relative difference on the tasks in Table 8. CoLA has the largest $\Delta_{\{D\}}(\mathcal{T})$ among all tasks, suggesting the expected high word order reliance. $\Delta_{\{D\}}(\mathcal{T})$ is lowest for QQP.

F Fine-tuning with randomized data

We perform additional experiments using the fine-tuned models from §4.1. Specifically, we construct unigram randomized train and test sets (denoted as *shuffled*) of a subset of tasks to evaluate whether models fine-tuned on natural or unnatural task data (having natural or unnatural pre-training prior) are able to understand unnatural data during testing. Sinha et al. showed for MNLI there exists at least one permutation for many examples which can be predicted correctly by the model. However, they also showed that every sentence can have many permutations which cannot be predicted correctly as well. We follow them in this evaluation, and construct 100 permutations for each example in the dev set for each task to capture the overall accuracy.

Concretely, we use $\mathcal{M}_{\mathtt{N}}$, $\mathcal{M}_1$ and $\mathcal{M}_{\mathtt{UG}}$ as our pre-trained representations (trained with natural, unigram sentence shuffle and corpus shuffle data respectively) and evaluate the effect of training and evaluation on natural and unnatural data in Table 9. We observe that all models perform poorly on the *shuffled* test set, compared to natural evaluation. However, interestingly, models have a slight advantage with a unigram randomized prior ($\mathcal{M}_1$), with CoLA having the biggest performance gain. PAWS task suffers the biggest drop in performance (from 94.49 to 62.22) but the lowest gain in $\mathcal{M}_1$, confirming our conclusion from §4.1 that most of the word order information necessary for PAWS is learned from the task itself.

Furthermore, training on shuffled data surprisingly leads to high performance on natural data for $\mathcal{M}_{\mathtt{N}}$ in case of several tasks, the effect being weakest in case of CoLA and PAWS. This suggests that for tasks other than CoLA and PAWS, spurious correlations are leveraged by the models during fine-tuning (cf. Gururangan et al. 2018; Poliak et al. 2018; Tsuchiya 2018). We also observe evidence of *domain matching*, where models improve their performance on evaluation on shuffled data when

name	fine-tune-train	fine-tune-eval	MNLI	QNLI	RTE	CoLA	MRPC	SST-2	PAWS
$\mathcal{M}_{\text{N}}$	natural	natural	86.08 +/- 0.15	92.45 +/- 0.24	73.62 +/- 3.09	52.44 +/- 21.22	89.09 +/- 0.88	93.75 +/- 0.44	94.49 +/- 0.18
	natural	shuffled	68.11 +/- 0.52	81.08 +/- 0.38	56.72 +/- 3.29	4.77 +/- 1.82	75.94 +/- 1.01	80.78 +/- 0.37	62.22 +/- 0.09
	shuffled	natural	82.99 +/- 0.16	89.32 +/- 0.23	57.9 +/- 4.71	0.0 +/- 0.0	79.71 +/- 2.57	89.12 +/- 0.5	72.03 +/- 13.79
	shuffled	shuffled	79.96 +/- 0.1	87.51 +/- 0.09	59.07 +/- 3.2	1.4 +/- 2.17	79.17 +/- 0.35	86.11 +/- 0.5	65.15 +/- 0.48
$\mathcal{M}_{1}$	natural	natural	82.64 +/- 0.15	89.05 +/- 0.15	68.48 +/- 2.51	31.07 +/- 9.97	85.97 +/- 0.89	90.41 +/- 0.43	89.69 +/- 0.59
	natural	shuffled	76.67 +/- 0.34	87.21 +/- 0.17	65.8 +/- 6.11	23.06 +/- 5.3	81.84 +/- 0.43	83.94 +/- 0.33	62.86 +/- 0.19
	shuffled	natural	79.87 +/- 0.1	87.81 +/- 0.36	65.65 +/- 2.33	24.53 +/- 13.63	82.51 +/- 0.82	86.45 +/- 0.41	73.34 +/- 6.88
	shuffled	shuffled	79.75 +/- 0.0	88.21 +/- 0.24	64.88 +/- 6.32	22.43 +/- 10.79	82.65 +/- 0.42	86.25 +/- 0.4	63.15 +/- 2.2
$\mathcal{M}_{\text{UG}}$	natural	natural	71.93 +/- 0.21	66.94 +/- 9.21	53.7 +/- 1.01	0.92 +/- 2.06	70.57 +/- 0.66	83.17 +/- 1.5	58.59 +/- 0.33
	natural	shuffled	62.27 +/- 0.57	63.13 +/- 7.13	52.42 +/- 2.77	0.09 +/- 0.21	70.56 +/- 0.33	79.41 +/- 0.63	56.91 +/- 0.16
	shuffled	natural	67.62 +/- 0.3	66.49 +/- 0.49	52.17 +/- 1.26	0.0 +/- 0.0	70.37 +/- 0.93	79.93 +/- 1.01	57.59 +/- 0.29
	shuffled	shuffled	67.02 +/- 0.33	66.24 +/- 0.33	53.44 +/- 0.53	0.08 +/- 0.18	70.28 +/- 0.62	80.05 +/- 0.4	57.38 +/- 0.16

Table 9: Fine-tuning evaluation by varying different sources of word order (with mean and std dev). We vary the word order contained in the pre-trained model ($\mathcal{M}_{\text{N}}$,$\mathcal{M}_{1}$,$\mathcal{M}_{\text{UG}}$); in fine-tuning training set (natural and shuffled); and in fine-tuning evaluation (natural and shuffled). Here, *shuffled* corresponds to unigram shuffling of words in the input. In case of fine-tune evaluation containing shuffled input, we evaluate on a sample of 100 unigram permutations for each data point in the dev set of the corresponding task.

Model	UD EWT		PTB	
	UAS	LAS	UAS	LAS
$\mathcal{M}_{\text{N}}$	90.92%	87.87%	95.42%	93.75%
$\mathcal{M}_{1}$	91.18%	88.19%	95.90%	94.35%
$\mathcal{M}_{2}$	91.11%	88.12%	95.74%	94.16%
$\mathcal{M}_{3}$	91.05%	87.94%	95.73%	94.14%
$\mathcal{M}_{4}$	90.88%	87.78%	95.77%	94.16%
$\mathcal{M}_{\text{UG}}$	90.47%	87.42%	95.81%	94.28%

Table 10: Unlabeled Attachment Score (UAS) on Dependency parsing task on two datasets, UD EWT and PTB, using the Second order Tree CRF Neural Dependency Parser (Zhang et al., 2020)

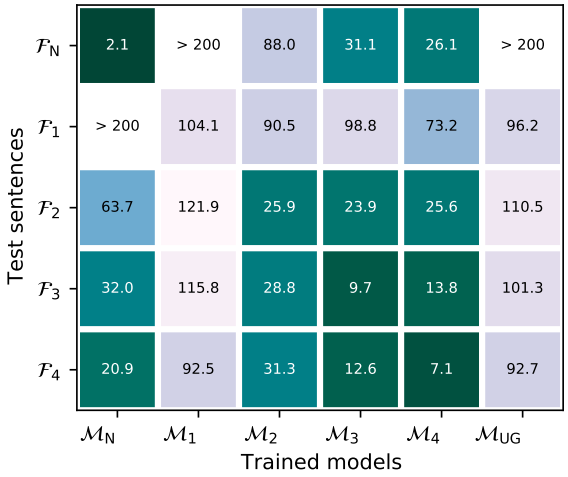

Figure 4: BPPL scores per model per test scenario.

the training data source is changed from natural to shuffled (for $\mathcal{M}_{\text{N}}$, MNLI shuffled evaluation improves from 68.11 to 79.96 just by changing the training corpus from natural to shuffled). We observe this behavior consistently for all tasks with all pre-trained representations.

G Dependency parsing using Second order Tree CRF Neural Dependency Parser

We also conduct extensive experiments with Second Order Tree CRF Neural Dependency parser from Zhang et al. (2020), using their provided codebase.[13] We report the results on UD EWT and PTB corpus in Table 10. Strangely enough, we find the gap to be even smaller across the different randomization models, even for some cases the performance on R_1 improves over OR. We suspect this result is due to two reasons: **(a)** Due to the presence of the complex Biaffine Dependency parser consisting of multiple LSTMs and individual MLP heads for each dependency arc (left and right), the majority of learning of the task is done by the parser itself; **(b)** Zhang et al. (2020) downsample the BERT representation to 100 dimensions which is then combined with the learned LSTM representations, thereby minimizing the impact of the pre-trained representations. Our hypothesis is confirmed by the published results of Zhang et al. (2020) on the Github repository, which shows a minimal gap between models with or without BERT.

[13] https://github.com/yzhangcs/parser

H Perplexity analysis

We measure perplexity of various pre-trained randomization models on text that is randomized using the same function $\mathcal{F}$. Conventional language models compute the perplexity

of a sentence S by using past tokens ($S_{<t} = (w_1, w_2, \ldots, w_{t-1})$) and the application of chain rule ($\sum_{t=1}^{|S|} \log P_{LM}(w_t|S_{t-1})$). However, this formulation is not defined for MLM, as a word is predicted using the entire sentence as a context. Following Salazar et al. (2020), we measure *Pseudo-Perplexity*, i.e., given a sentence S, we compute the log-probability of the missing word in S by iteratively masking out the specific word, and computing the average log-probability per word in S:

$$\texttt{PLL}(S) = \frac{1}{|S|} \sum_{w \in S} \log P_{\text{MLM}}(w|S_{\setminus w}; \theta) \quad (3)$$

We bootstrap the PLL score of a test corpus T by drawing 100 samples five times with replacement. We also similarly compute the bootstrap perplexity following Salazar et al.:

$$\texttt{BPLL}_T = \exp(-\frac{1}{N} \sum_{S \in W} \texttt{PLL}(S)), \quad (4)$$

where W is the combined bootstrap sample containing N sentences drawn with replacement from T. We compute this score on 6 pre-trained models, over four randomization schemes on the bootstrapped sample W (i.e., we use the same n-gram randomization function $\mathcal{F}_i$). Thus, we obtain a 5x6 matrix of BPLL scores, which we plot in Figure 4.

We observe that the pre-trained model $\mathcal{M}_{\text{N}}$ has the lowest perplexity on the sentences with natural word order. Pre-trained models with random word order exhibit significantly higher perplexity than the normal word order sentences (top row). With the exception of $\mathcal{M}_1$, the models pre-trained on randomized data ($\mathcal{M}_2$, $\mathcal{M}_3$ and $\mathcal{M}_4$) all display the lowest perplexity for their respective $n = 2, 3, 4$ randomizations. These results indicate that the models retain and detect the specific word order for which they were trained.

I The usefulness of word order

The results in §4.1 suggest that, with proper fine-tuning, an unnaturally trained model can reach a level of performance comparable to that of a naturally pre-trained model. However, we want to understand whether natural word order pre-training offers any advantage during the early stages of fine-tuning. Towards that end, we turn to compute the Minimum Description Length (MDL; Rissanen, 1984). MDL is designed to characterize the complexity of data as the length of the shortest program required to generate it. Thus, the length of the minimum description (in bits) should provide a fair estimate of how much word order is useful for fine-tuning in a few-shot setting. Specifically, we leverage the Rissanen Data Analysis (RDA) framework from Perez et al. (2021) to evaluate the MDL of pre-trained models on our set of downstream tasks. Under mild assumptions, if a pre-trained model θ_1 is useful for solving a particular task T over θ_2, then the MDL in bits obtained by using θ_1 should be shorter than θ_2. We follow the experimental setup of Perez et al. to compute the MDL on several tasks using $\theta = \{\mathcal{M}_{\text{N}}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4\}$, over three seeds and on three epochs of training. Concretely, RDA involves sampling 9 blocks of data from the dataset at random, where the size of each block is increased monotonically, training on 8 blocks while evaluating the model's loss (or *codelength*) on the ninth. The minimum number of data samples in the smallest block is set at 64, while the largest number of data samples used in the last block is 10,000.

We observe that the value of MDL is consistently lowest for naturally pre-trained data (Figure 5). For purportedly word order reliant datasets such as RTE, CoLA and PAWS, the gap between the MDL scores among the natural and unnatural models is high. PAWS, specifically, has the largest advantage in the beginning of optimization, however with more fine-tuning, the model re-learns correct word order (§4.1). The present analyses, when taken in conjunction with our main results in §4.1, suggest that fine-tuning on large training datasets with complex classifiers in the pursuit of state-of-the-art results has mostly nullified the impact of word order in the pre-trained representations. Few shot (Bansal et al., 2020) and few sample (Zhang et al., 2021) learning and evaluation could potentially require more word order signal, thereby encouraging the model to leverage its own learned syntax better.

J At what point do models learn word order during pre-training?

Results from §4.1 beg the question: when, if at all, during pre-training does a model learn the natural word order? We aim to answer that question by comparing downstream task performance of RoBERTa base on intermediate checkpoints with

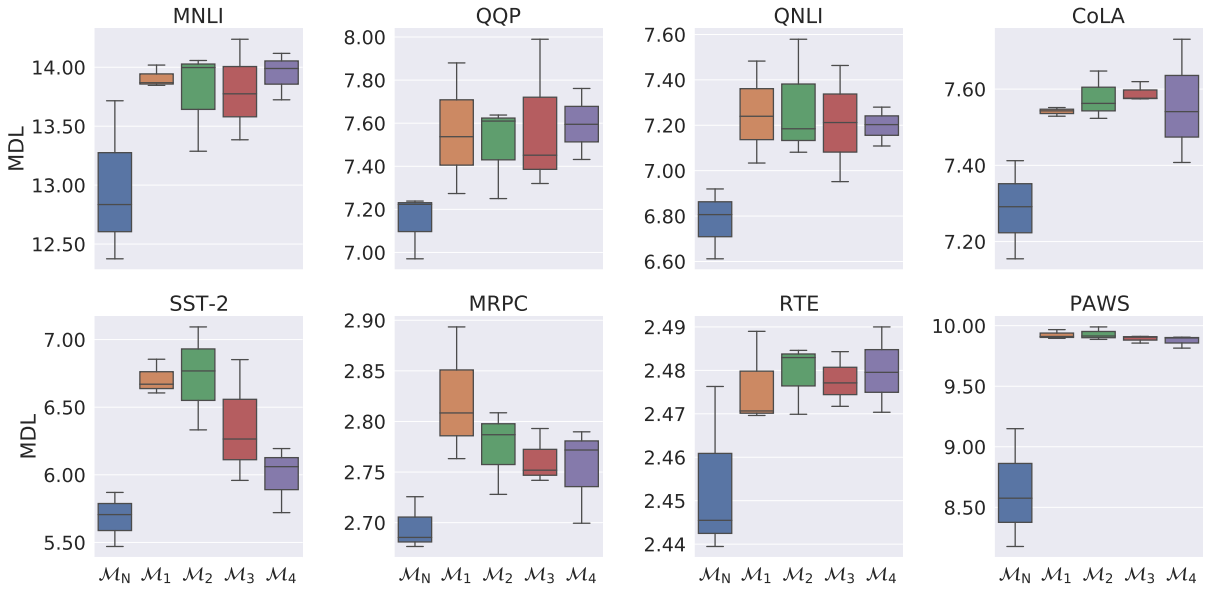

Figure 5: Rissanen Data Analysis (Perez et al., 2021) on the GLUE benchmark and PAWS datasets. The lower minimum description length (MDL, measured in kilobits), the better the learning ability of the model.

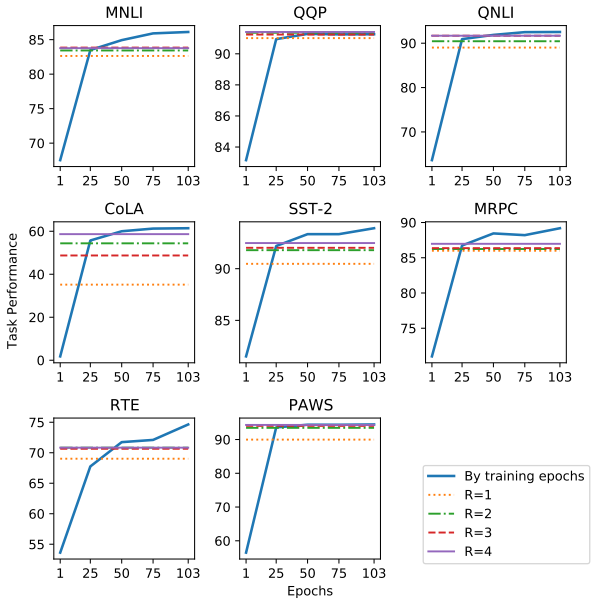

Figure 6: Comparison among GLUE task performance from different steps in pre-training of RoBERTa on BookWiki Corpus.

Model	UD EWT		PTB	
	MLP	Linear	MLP	Linear
$\mathcal{M}_N$	93.74 +/- 0.15	88.82 +/- 0.42	97.07 +/- 0.38	93.1 +/- 0.65
$\mathcal{M}_1$	88.60 +/- 3.43	80.76 +/- 3.38	95.33 +/- 0.37	87.83 +/- 1.86
$\mathcal{M}_2$	93.39 +/- 0.45	87.58 +/- 1.06	96.96 +/- 0.15	91.80 +/- 0.50
$\mathcal{M}_3$	92.89 +/- 0.65	86.78 +/- 1.32	97.03 +/- 0.13	91.70 +/- 0.70
$\mathcal{M}_4$	92.83 +/- 0.61	87.23 +/- 0.77	96.96 +/- 0.12	92.08 +/- 0.39
$\mathcal{M}_{UG}$	89.10 +/- 0.21	79.75 +/- 0.5	94.12 +/- 0.01	84.15 +/- 0.51

Table 11: Accuracy on the part-of-speech labelling task (POS) on two datasets, UD EWT and PTB, using the Pareto Probing framework (Pimentel et al., 2020b).

Model	UD EWT		PTB	
	MLP	Linear	MLP	Linear
$\mathcal{M}_N$	89.63 +/- 0.60	84.35 +/- 0.78	93.96 +/- 0.63	88.35 +/- 1.00
$\mathcal{M}_1$	83.55 +/- 3.31	75.26 +/- 3.08	91.10 +/- 0.38	82.34 +/- 1.37
$\mathcal{M}_2$	88.57 +/- 0.68	82.05 +/- 1.10	93.27 +/- 0.26	86.88 +/- 0.87
$\mathcal{M}_3$	88.69 +/- 1.09	82.37 +/- 1.26	93.46 +/- 0.29	87.12 +/- 0.72
$\mathcal{M}_4$	88.66 +/- 0.76	82.58 +/- 1.04	93.49 +/- 0.33	87.30 +/- 0.79
$\mathcal{M}_{UG}$	84.93 +/- 0.34	76.30 +/- 0.52	89.98 +/- 0.43	78.59 +/- 0.68

Table 12: Accuracy on the dependency arc labelling task (DAL) on two datasets (with mean and std dev), UD EWT and PTB, using the Pareto Probing framework (Pimentel et al., 2020a).

that of the random word order pretrained models. The idea is to find the point during pre-training on natural corpus at which the model exceeds the task performance of the random pre-training model.

Performance on all tasks (Figure 6) increases rapidly during the first 20-25 epochs of pre-training. For some tasks, the word order information only helps after 30-50 pre-training epochs.

K More results from Syntactic Probes

We computed the Pareto Hypervolume on the dependency parsing task (Pimentel et al., 2020a). The Pareto Hypervolume is computed as the Area Under Curve (AUC) score over all hyperparameter runs, where the models are arranged based on their complexity. We observe minimal differences in the Pareto Hypervolumes (Table 13) among $\mathcal{M}_N$ and

Model	UD EWT	PTB
$\mathcal{M}_{\text{N}}$	0.528 +/- 0.01	0.682 +/- 0.01
$\mathcal{M}_1$	0.489 +/- 0.03	0.648 +/- 0.01
$\mathcal{M}_2$	0.529 +/- 0.00	0.681 +/- 0.01
$\mathcal{M}_3$	0.528 +/- 0.02	0.689 +/- 0.01
$\mathcal{M}_4$	0.525 +/- 0.00	0.683 +/- 0.01
$\mathcal{M}_{\text{UG}}$	0.510 +/- 0.01	0.640 +/- 0.05

Table 13: Pareto Hypervolume of dependency parsing task (DEP) on two datasets (with mean and std dev), UD EWT and PTB, using the Pareto Probing framework (Pimentel et al., 2020b).

the randomization models for both datasets.

We also investigated two "easy" tasks, Part-of-Speech tagging (POS) and Dependency Arc Labeling (DAL) from the Pareto Probing framework. For POS (Table 11) and DAL (Table 12), since these tasks are simpler than DEP, the gap between $\mathcal{M}_{\text{N}}$ and unnaturally pre-trained models reduces even more drastically. The gap between $\mathcal{M}_{\text{N}}$ and $\mathcal{M}_1$ reduces to just 3.5 points on average for PTB in both POS and DAL.

L Non parametric probes

Probability difference. In the original formulation (Goldberg, 2019; Wolf, 2019), the effectiveness of each stimulus is determined by the accuracy metric, computed as the number of times the probability of the correct focus word is greater than that of the incorrect word ($P(\text{good}) > P(\text{bad})$). We observed that this metric might not be reliable per se, since the probabilities may themselves be extremely low for all tokens, even when focus word probability decreases drastically from $\mathcal{M}_{\text{N}}$ to $\mathcal{M}_{\text{UG}}$. Thus, we also report the mean difference of probabilities, $(\frac{1}{N}\sum_i^N P(\text{good}_i) - P(\text{bad}_i))$, scaled up by a factor of 100 for ease of observation, in Figure 9, Figure 8 and Figure 7. We observe the highest difference between probabilities of the correct and incorrect focus words for the model pretrained on the natural word order ($\mathcal{M}_{\text{N}}$). Moreover, with each step from $\mathcal{M}_1$ to $\mathcal{M}_4$, the difference between probablities of correct and incorrect focus words increases, albeit marginally, showing that pre-trained models with fewer n-gram words perturbed capture more word order information. $\mathcal{M}_{\text{UG}}$, the model with the distributional prior ablated, performs the worst, as expected.

Accuracy comparison. We provide the accuracy as measured by Goldberg (2019); Wolf (2019) on the probing stimuli in Table 14, Table 15 and Table 16. We also highlight the difference in probability ($P(\text{good}) - P(\text{bad})$) in the table to provide a more accurate picture. All experiments were conducted on three pre-trained seeds for each model in our set of models. However, the low token probabilities in $\mathcal{M}_{\text{UG}}$ tend to present unreliable scores. For example, in the case of Gulordava et al. (2018b) stimuli, unnatural models provide better scores compared to the natural model. We also observe for the Linzen et al. (2016) stimuli that the results on model condition 4 (number of attractors) are surprisingly high for $\mathcal{M}_{\text{UG}}$ whereas the individual token probabilities are lowest. We believe these inconsistencies stem from extremely low token probabilities themselves.

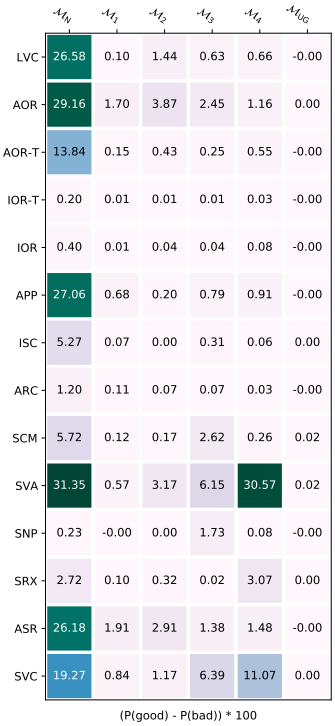

Figure 7: The difference in word probabilities for stimuli in Marvin and Linzen (2018): Simple Verb Agreement (SVA), In a sentential complement (SCM), Short VP Coordination (SVC), Long VP Coordination (LVC), Across a prepositional phrase (APP), Across a subject relative clause (ASR), Across an object relative clause (AOR), Across an object relative (no *that*) (AOR-T), In an object relative clause (IOR), In an object relative clause (no *that*) (IOR-T), Simple Reflexive (SRX), In a sentential complement (ISC), Across a relative clause (ARC), Simple NPI (SNP).

Balancing datasets on inflection by upsampling. The stimuli datasets of Linzen et al. (2016) and Gulordava et al. (2018b) turned out to be heavily skewed towards words where singular was the cor-

model condition	$\mathcal{M}_{\text{N}}$	$\mathcal{M}_{\text{UG}}$	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$
1	93.45 (0.89) [25.04]	58.87 (0.41) [0.0]	59.96 (1.58) [0.08]	63.63 (0.6) [1.25]	64.7 (1.44) [2.79]	70.47 (1.9) [4.01]
2	92.8 (1.22) [23.8]	63.03 (1.35) [0.01]	58.22 (1.5) [0.09]	61.15 (2.07) [0.82]	63.84 (2.41) [2.09]	64.7 (1.92) [3.07]
3	87.71 (1.34) [22.03]	64.06 (3.52) [0.0]	56.69 (2.98) [0.03]	56.83 (3.63) [0.85]	61.1 (0.32) [2.02]	63.0 (3.36) [2.35]
4	92.67 (0.52) [22.16]	76.33 (1.38) [0.0]	62.33 (7.61) [0.08]	63.17 (9.09) [1.12]	69.42 (1.77) [2.1]	67.67 (7.02) [3.43]

Table 14: Linzen et al. (2016) stimuli results in raw accuracy. Values in parenthesis reflect the standard deviation over different seeds of pre-training. Values in square brackets indicate the mean probability difference among correct and incorrect words.

model condition	$\mathcal{M}_{\text{N}}$	$\mathcal{M}_{\text{UG}}$	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$
0	79.42 (5.5) [2.43]	47.83 (3.76) [-0.0]	53.67 (1.38) [0.03]	58.75 (6.38) [0.05]	63.58 (4.11) [0.14]	63.75 (3.28) [0.17]
1	72.83 (4.07) [2.55]	44.5 (0.5) [0.0]	70.83 (5.8) [0.02]	64.83 (1.76) [-0.09]	71.67 (6.71) [0.21]	71.5 (2.65) [0.61]
2	55.56 (0.0) [0.92]	88.89 (11.11) [0.0]	81.48 (12.83) [0.03]	51.85 (6.42) [0.04]	62.96 (6.42) [0.38]	74.07 (16.97) [0.61]

Table 15: Gulordava et al. (2018b) stimuli results in raw accuracy.Values in parenthesis reflect the standard deviation over different seeds of pre-training. Values in square brackets indicate the mean probability difference among correct and incorrect words.

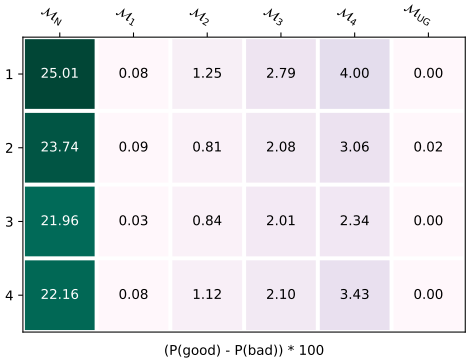

Figure 8: Linzen et al. (2016)

rect inflection (as opposed to plural). This dataset imbalance caused the weak models (such as $\mathcal{M}_{\text{UG}}$) to have surprisingly high scores - the weak models were consistently providing higher probability for the singular inflection (Table 17). We upsample for both datasets, balancing the frequency of correct singular and plural inflections. We compute the upsampling number to the next multiple of 100 of the count of original singular inflections. For example, in condition 4 of Linzen et al. (2016) dataset, we upsample both S and P to 300 rows each. This type of balancing via upsampling largely alleviated the inconsistencies we observed, and might prove to be useful when evaluating other models on these datasets in future.

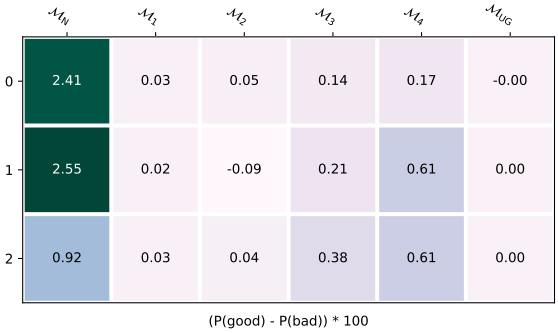

Figure 9: Gulordava et al. (2018b)

Model condition	$\mathcal{M}_{\text{N}}$	$\mathcal{M}_{\text{UG}}$	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$
AOR	89.98 (1.96) [29.16]	50.0 (0.01) [0.0]	60.17 (1.61) [1.7]	66.61 (7.1) [3.87]	63.57 (2.39) [2.45]	61.26 (4.91) [1.16]
AOR-T	77.4 (7.74) [13.84]	50.0 (0.0) [0.0]	78.88 (0.64) [0.15]	52.17 (2.14) [0.43]	48.85 (3.8) [0.25]	57.06 (3.49) [0.55]
APP	89.94 (4.16) [27.06]	50.01 (0.02) [-0.0]	70.34 (1.9) [0.68]	53.61 (3.3) [0.2]	53.03 (1.75) [0.79]	60.6 (4.41) [0.91]
ARC	85.06 (5.92) [1.2]	50.05 (0.08) [-0.0]	62.39 (1.91) [0.11]	74.57 (5.99) [0.07]	67.55 (3.84) [0.07]	62.88 (3.45) [0.03]
ASR	87.19 (3.58) [26.18]	50.0 (0.0) [-0.0]	78.55 (10.01) [1.91]	81.73 (5.1) [2.91]	62.8 (0.35) [1.38]	67.23 (6.82) [1.48]
IOR	89.83 (3.33) [0.4]	50.55 (0.95) [-0.0]	56.28 (2.66) [0.01]	58.96 (4.28) [0.04]	70.49 (2.2) [0.04]	62.82 (8.51) [0.08]
IOR-T	74.05 (8.26) [0.2]	50.61 (1.05) [-0.0]	52.63 (2.07) [0.01]	57.35 (4.88) [0.01]	61.85 (4.75) [0.01]	55.16 (6.59) [0.03]
ISC	85.87 (9.6) [5.27]	50.0 (0.0) [0.0]	67.85 (2.62) [0.07]	82.66 (9.43) [0.0]	77.69 (4.51) [0.31]	68.65 (5.71) [0.06]
LVC	93.0 (0.75) [26.58]	49.92 (0.14) [-0.0]	70.42 (6.79) [0.1]	87.5 (7.26) [1.44]	85.42 (3.84) [0.63]	81.08 (5.13) [0.66]
SCM	88.6 (3.49) [5.72]	50.0 (0.0) [0.02]	63.73 (7.94) [0.12]	82.12 (0.92) [0.17]	86.44 (3.67) [2.62]	80.27 (2.46) [0.26]
SRX	91.0 (6.07) [2.72]	50.0 (0.0) [0.0]	88.0 (10.11) [0.1]	92.25 (10.27) [0.32]	94.25 (5.02) [0.02]	91.0 (6.5) [3.07]
SVA	95.33 (7.23) [31.35]	50.0 (0.0) [0.02]	86.0 (5.29) [0.57]	85.17 (12.87) [3.17]	94.67 (5.25) [6.15]	88.83 (9.57) [30.57]
SVC	97.54 (1.58) [19.27]	50.0 (0.0) [-0.0]	83.58 (4.58) [0.84]	83.71 (8.78) [1.17]	93.29 (7.4) [6.39]	81.04 (3.66) [11.07]

Table 16: Marvin and Linzen (2018) stimuli results in raw accuracy. Values in parenthesis reflect the standard deviation over different seeds of pre-training. Values in square brackets indicate the mean probability difference among correct and incorrect words. Abbreviations: Simple Verb Agreement (SVA), In a sentential complement (SCM), Short VP Coordination (SVC), Long VP Coordination (LVC), Across a prepositional phrase (APP), Across a subject relative clause (ASR), Across an object relative clause (AOR), Across an object relative (no *that*) (AOR-T), In an object relative clause (IOR), In an object relative clause (no *that*) (IOR-T), Simple Reflexive (SRX), In a sentential complement (ISC), Across a relative clause (ARC), Simple NPI (SNP).

Model condition	$\mathcal{M}_{\text{N}}$	$\mathcal{M}_{\text{UG}}$	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$	S/P
1	94.04 (0.8)	62.64 (0.5)	62.18 (1.33)	64.91 (0.14)	65.35 (1.78)	70.88 (1.88)	14011 / 10112
2	93.28 (0.94)	71.24 (0.85)	63.03 (1.69)	62.92 (2.57)	65.25 (3.13)	65.61 (2.35)	3120 / 1312
3	89.1 (0.58)	74.05 (1.85)	62.94 (3.13)	59.18 (3.32)	63.54 (1.72)	63.05 (2.0)	733 / 215
4	90.53 (0.9)	80.03 (0.59)	63.16 (4.83)	63.94 (6.92)	66.41 (3.17)	66.28 (4.64)	206 / 51

Table 17: Linzen et al. (2016) stimuli results in raw accuracy on original, unbalanced data. Values in parenthesis reflect the standard deviation. S/P reflects the count of correct singular and plural focus words.

	OR	R1	R2	R3	R4
1	They are commonly known as daturas, but also known as devil's trumpets, not to be confused with angel's trumpets, its closely related genus "Brugmansia".	be They angel's also but trumpets, genus related devil's as commonly closely known its daturas, trumpets, as "Brugmansia". confused with known are to not	as devil's They genus not to trumpets, closely related "Brugmansia". are commonly trumpets, its also known known as be confused daturas, but with angel's	"Brugmansia". related They are commonly trumpets, its closely as daturas, but known genus also known as trumpets, confused with angel's devil's not to be	its closely related genus They are commonly known trumpets, as trumpets, daturas, but also known as "Brugmansia". not to be confused with angel's devil's
2	They are also sometimes called moonflowers, jimsonweed, devil's weed, hell's bells, thorn-apple, and many more.	are devil's bells, called weed, hell's thorn-apple, and many They also more. moonflowers, jimsonweed, sometimes	more. They are hell's bells, also sometimes and many called moonflowers, jimsonweed, devil's weed, thorn-apple,	jimsonweed, devil's weed, They are also thorn-apple, and many bells, more. hell's sometimes called moonflowers,	moonflowers, They are also sometimes bells, thorn-apple, and many more. called jimsonweed, devil's weed, hell's
3	Its precise and natural distribution is uncertain, owing to its extensive cultivation and naturalization throughout the temperate and tropical regions of the globe.	throughout owing precise extensive temperate and naturalization and tropical of to natural is its Its distribution cultivation the globe. uncertain, regions the and	and natural distribution is tropical to its and naturalization throughout the the temperate and globe. Its precise uncertain, owing extensive cultivation regions of	uncertain, owing to Its precise and its extensive cultivation of globe. natural distribution is the the and tropical regions and naturalization throughout temperate	globe. Its precise and natural cultivation distribution the is uncertain, owing to its extensive and naturalization throughout the temperate and tropical regions of
4	Its distribution within the Americas and North Africa, however, is most likely restricted to the United States, Mexico and Southern Canada in North America, and Tunisia in Africa where the highest species diversity occurs.	distribution Mexico occurs. likely diversity North however, species most the Tunisia where in and and North Canada Southern America, highest Africa United the and in Americas Its within States, is to the restricted Africa,	and Tunisia the Americas distribution within Mexico and is most United States, Africa, however, Africa where in North Its and North in Southern Canada America, the to the likely restricted occurs. highest species diversity	likely Its highest species diversity United States, Mexico restricted to the Africa where the occurs. distribution within the and Tunisia in however, is most Americas and Southern Canada and North Africa, in North America,	Tunisia occurs. Its distribution within the Africa where the highest in restricted to the United Canada in North America, most North Africa, however, is and Americas likely diversity States, Mexico and Southern species and
5	All species of "Datura" are poisonous, especially their seeds and flowers.	seeds and species of poisonous, "Datura" their are All flowers. especially	"Datura" are especially their flowers. seeds and of All species poisonous,	especially their seeds flowers. "Datura" are poisonous, All species of and	flowers. poisonous, species of "Datura" are All especially their seeds and
6	Some South American plants formerly thought of as "Datura" are now treated as belonging to the distinct genus "Brugmansia" ("Brugmansia" differs from "Datura" in that it is woody, making shrubs or small trees, and it has pendulous flowers, rather than erect ones).	and "Datura" treated from than flowers, it small belonging woody, thought as ones). South differs Some "Brugmansia" American as are in the rather pendulous distinct making now erect "Datura" to ("Brugmansia" of formerly trees, or is it that plants genus has shrubs	"Brugmansia" ("Brugmansia" than erect pendulous genus and ones). is woody, small trees, of as the distinct flowers, rather Some South differs from American plants treated as formerly thought belonging to "Datura" in making that it "Datura" are it has now shrubs or	woody, small trees, and has pendulous flowers, as belonging to Some making shrubs or as rather than erect "Datura" are now "Brugmansia" ("Brugmansia" differs the distinct genus from "Datura" in formerly thought of it treated that it is ones). South American plants	belonging to the distinct has making Some ("Brugmansia" differs from "Datura" in are now treated as genus pendulous shrubs flowers, rather than erect or ones). "Brugmansia" that it is woody, South American plants formerly thought of as "Datura" small trees, and it
7	Other related taxa include	taxa Other include related	include Other related taxa	include Other related taxa	Other related taxa include
8	"Hyoscyamus niger", "Atropa belladonna", "Mandragora officinarum", Physalis, and many more.	and many niger", officinarum", belladonna", "Mandragora "Atropa "Hyoscyamus more. Physalis,	belladonna", "Mandragora "Hyoscyamus niger", many Physalis, and more. officinarum", "Atropa	more. Physalis, and many belladonna", "Mandragora officinarum", "Hyoscyamus niger", "Atropa	niger", more. belladonna", "Mandragora officinarum", Physalis, "Atropa many and "Hyoscyamus
9	The name "Datura" is taken from Sanskrit ' 'thorn-apple', ultimately from Sanskrit ' 'white thorn-apple' (referring to "Datura metel" of Asia).	of Asia). taken from name The "Datura" ' is to 'thorn-apple', Sanskrit ' Sanskrit metel" 'white (referring from "Datura thorn-apple' ultimately	"Datura" is taken from to ' 'thorn-apple', Sanskrit ' 'white of thorn-apple' (referring Asia). The name Sanskrit ultimately from "Datura metel"	Sanskrit ' The name "Datura" 'thorn-apple', ultimately from metel" Asia). is taken from of 'white (referring to "Datura Sanskrit ' thorn-apple'	Asia). The name "Datura" is from taken of from Sanskrit ' 'thorn-apple', ultimately Sanskrit ' 'white thorn-apple' (referring to "Datura metel"
10	In the Ayurvedic text Sushruta different species of Datura are also referred to as ' and '.	the of also Sushruta Datura are referred to as In Ayurvedic and different species ' text '.	species of referred to are also Datura Sushruta different and as ' Ayurvedic text In the '.	as ' and In the Ayurvedic also referred to species of Datura are text Sushruta different '.	different In the Ayurvedic text also referred to as and Sushruta ' species of Datura are '.

Table 18: First 10 lines from the BookWiki corpus, and their respective n-gram permutations.

Model	RTE	MRPC	SST-2	CoLA	QQP	QNLI	MNLI	PAWS
$\mathcal{M}_{\texttt{N}}$	2e-05	2e-05	1e-05	2e-05	1e-05	1e-05	1e-05	2e-05
$\mathcal{M}_{\texttt{1}}$	2e-05	1e-05	1e-05	1e-05	3e-05	1e-05	2e-05	2e-05
$\mathcal{M}_{\texttt{2}}$	2e-05	2e-05	1e-05	1e-05	2e-05	1e-05	1e-05	3e-05
$\mathcal{M}_{\texttt{3}}$	3e-05	1e-05	2e-05	2e-05	3e-05	1e-05	1e-05	2e-05
$\mathcal{M}_{\texttt{4}}$	3e-05	1e-05	2e-05	2e-05	2e-05	1e-05	1e-05	2e-05
$\mathcal{M}_{\texttt{512}}$	1e-05	3e-05	2e-05	2e-05	3e-05	2e-05	3e-05	2e-05
$\mathcal{M}_{\texttt{UG}}$	2e-05	1e-05	3e-05	1e-05	3e-05	3e-05	3e-05	2e-05
$\mathcal{M}_{\texttt{UF}}$	2e-05	1e-05	3e-05	2e-05	3e-05	3e-05	3e-05	1e-05
$\mathcal{M}_{\texttt{RI}}$	1e-05	1e-05	3e-05	1e-05	1e-05	1e-05	2e-05	1e-05
$\mathcal{M}_{\texttt{NP}}$	1e-05	3e-05	2e-05	1e-05	1e-05	1e-05	1e-05	1e-05

Table 19: Fine-tuning hyperparam Learning rate of each model for each task in GLUE and PAWS

Model	RTE	MRPC	SST-2	CoLA	QQP	QNLI	MNLI	PAWS
$\mathcal{M}_{\texttt{N}}$	16	16	32	16	16	32	32	16
$\mathcal{M}_{\texttt{1}}$	32	32	16	32	32	16	32	16
$\mathcal{M}_{\texttt{2}}$	32	16	32	16	32	32	16	32
$\mathcal{M}_{\texttt{3}}$	32	32	16	32	32	16	32	32
$\mathcal{M}_{\texttt{4}}$	32	16	32	16	32	32	32	32
$\mathcal{M}_{\texttt{512}}$	32	16	16	32	32	16	16	16
$\mathcal{M}_{\texttt{UG}}$	16	16	16	16	32	16	16	32
$\mathcal{M}_{\texttt{UF}}$	16	32	16	16	32	16	16	16
$\mathcal{M}_{\texttt{RI}}$	16	16	32	16	16	16	32	16
$\mathcal{M}_{\texttt{NP}}$	16	32	16	16	32	16	16	16

Table 20: Finetuning hyperparam batch size of each model for each task in GLUE and PAWS