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 / Montreal Institute of Learning Algorithms

{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 almost entirely 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 on tasks specifically designed to be challenging for models that ignore word order. Our models 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 underscore the importance of curating challenging evaluation datasets that require deeper linguistic knowledge.

1 Introduction

Natural Language Processing 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., 2018), has proven wildly successful, but 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). 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). 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 kind of abstractions that we intuitively believe are important for representing natural language" rather than "simply modeling complex co-occurrence statistics" (ibid., p. 1).

In this work, we try to uncover how much of MLM's success comes from simple distributional information, as opposed to "the types of syntactic and semantic abstractions traditionally believed necessary for language processing" (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: while local distributional co-occurrences are not sensitive to word order changes, any sophisticated (English) NLP pipeline presumably depends on the important 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 MLM models (RoBERTa, Liu et al. 2019) on randomized data with permuted word order and examine their downstream performance. In our main experiments, we pre-train models on various permuted corpora that preserve sentence-level distributional information by randomly shuffling n-grams within the sentence (where $n \in \{1, 2, 3, 4\}$). We also experiment with training MLM transformers without positional embeddings, making them entirely order agnostic, and with training on a corpus sampled from the source corpus's uniform or unigram distribution, removing both distributional and word order information. We then evaluate these "permuted" models in a wide range of settings in comparison 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). This is partially explained by inductive biases in the model itself, as illustrated by the baselines performing reasonably well. That model is vastly outperformed, however, by permuted models with distributional information preserved at the sentence level, which perform remarkably close to regular MLM pre-training. It has recently been found that MLM models 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 (Gupta et al., 2021; Sinha et al., 2020). To the best of our knowledge, this work is the first to show that downstream performance for "unnatural language pre-training" is much closer to standard MLM than we might have expected.

In an effort to shed light on these findings, we 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 order is crosslinguistically (Greenberg 1963; Dryer 1992; Cinque 1999, i.a.).

Depending on one's position in NLP's great debates, the reader might draw different conclusions from these findings. One might argue that our downstream tasks are flawed evaluations, and that we need to examine models with examples that truly test strong generalization and compositionality. Alternatively, one could argue that human language understanding simply depends for a large part on the structure (i.e., order) of the *world*. Or, perhaps we are just not focusing enough on the tail of the distribution, where true syntactic capabilities would certainly be borne out. While these results may seem disappointing for NLP at large, this work also offers a message of hope and renewed purpose: can we design tasks that require more sophisticated reasoning, true compositionality, robustness against adversaries and strong human-like generalization? Can we design better experimental protocols for testing the phenomena we really care about? Can we be more careful with our baselines? This work is meant to deepen our understanding of MLM pre-training, and through this, orient future work towards promising new research directions.

2 Related Work

Sensitivity to word order in NLU. Sensitivity to word order for NLU tasks has recently been investigated concurrently by Sinha et al. (2020), Pham et al. (2020), and Gupta et al. (2021). 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). RoBERTa has been found to assign high confidence to targeted permutations, even with drastic changes in word order. Pham et al. (2020) show insensitivity on a larger set of tasks, including the entire GLUE benchmark, and inspect BERT, RoBERTa and ALBERT-based models. They find that certain tasks in GLUE, such as CoLA and RTE are more sensitive to permutations than others. Sinha et al. (2020) specifically investigate Natural Language Inference (NLI), and further add a distributional perspective on the rate of acceptance of permuted sentences by different Transformer and pre-Transformer era models. They also show similar effects in non-English datasets (e.g. the Chinese OCNLI corpus; Hu et al., 2020), and posit overly high model confidence to be one of the factors driving this issue. In this work, we go a step further and investigate the source of word order information in pre-training for a variety of downstream tasks and probing tasks.

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. It has also been shown 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 (Shen et al., 2020). 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., 2018), 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. (2021)

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 on non-linguistic data having 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 syntactic, semantic and world knowledge (Giulianelli et al., 2018; Rogers et al., 2020). Many of these works involve diagnostic classifiers or *parametric* probes, i.e. a function atop learned representations that is optimized to gauge aspects of linguistic information. How well the probe learns a given signal can be seen as a proxy for linguistic knowledge encoded in the representations (Hupkes et al., 2018). However, the community is divided on many aspects of probing (Belinkov, 2021) including how complex probes should be. On the one hand, proponents of *simple* linear probes argue that having explicit, focused classifiers should provide better signal as the function itself is too weak on its own (Alain and Bengio, 2016; Hewitt and Manning, 2019; Maudslay et al., 2020). On the other hand, proponents of *complex* probes claim that having strong representational capacity of the probe is actually helpful for extracting the most information from a representation (Pimentel et al., 2020b). There are also strong calls from the community to use information theoretic approaches instead of accuracy (Voita and Titov, 2020; Pimentel et al., 2020b), as well harder *tasks* for probing (Maudslay et al., 2020; Pimentel et al., 2020a). In this work, 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, stimuli-based probing (Linzen et al., 2016; Marvin and Linzen, 2018; Gulordava et al., 2018; Goldberg, 2019; Wolf, 2019; Warstadt et al., 2019a, 2020a,b; Ettinger, 2020) has been used to show that even without a learned probing function, BERT can predict syntactic properties with high confidence. Here, 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. To ensure fair comparability, we use the same RoBERTa (base) (Liu et al., 2019) architecture as the MLM model under investigation, primarily for its computational efficiency and good downstream task performance. We expect that other variations of MLM pre-training models 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) that was used for RoBERTa ablations (Liu et al., 2019). We denote the model trained on the original, un-modified BookWiki corpus as $\mathcal{M}_{\mathrm{N}}$. We use two methods for permuting word order: Sentence word order permutation (i.e., permuting word order at the sentence level) and Corpus word order permutation (i.e., permuting words at the whole corpus level).

Sentence word order permutation. To investigate to what extent the performance of MLM pre-training 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), where each sentence is stripped of its original word order. 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 account for n-grams, $\mathcal{F}_n$. 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 then train four RoBERTa models on the 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 data generation process, along with the pseudo code and sample quality, are provided in Appendix B.

Corpus word order bootstrap resample. The above formulation preserves higher order distributional information by grouping words in a sentence together. However, we need a baseline to understand how a model would perform devoid of this information. A randomly initialized RoBERTa is a good baseline for seeing how far we can get using only the model architecture as the inductive bias. We also construct a stronger baseline that captures word/subword information, without access to co-occurrence statistics. We denote this baseline as $\mathcal{M}_{\texttt{UG}}$, where we sample unigrams from BookWiki according to their frequencies, while also treating named entities as unigrams. Concretely, we first use Spacy's (Honnibal et al., 2020)[1] part-of-speech (POS) tagger and named entity recognition (NER) parser to extract all POS tag words and NER words (and their counts) from the source corpus (BookWiki), to form a word-to-frequency dictionary $\mathcal{W}$. For a phrase extracted by the NER parser, we remove the sub-tokens from the POS tag word set to maintain the same frequency of the tokens in BookWiki. Then, we replace the words in the original sentences $S_i \in$ BookWiki, with words drawn from a weighted distribution of words (by frequency) from $\mathcal{W}$. This allows us to construct $\mathcal{M}_{\texttt{UG}}$ such that it has exactly the same size as BookWiki but without 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 should provide a baseline of the limits of the inductive bias of the model alone in learning language.

Further ablations. We also train further model ablations with low distributional prior. Following the construction of corpus boostrap resample, we train a model where words are drawn uniformly from BookWiki corpus, without the frequency distribution ($\mathcal{M}_{\texttt{UF}}$). We also experiment with pre-training RoBERTa on the original corpus without positional embeddings to have a baseline where the model is stripped of its ability to learn word order. 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 $\mathcal{M}_{\texttt{NP}}$. Finally, we use a randomly initialized RoBERTa model as an ablation $\mathcal{M}_{\texttt{RI}}$ to observe the extent we can learn from each task with only the model's base inductive bias.

Pre-training details. Each model $\in (\mathcal{M}_{\texttt{N}}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_{\texttt{UG}}, \mathcal{M}_{\texttt{UF}}, \mathcal{M}_{\texttt{NP}})$ is a RoBERTa (base) model (12 layers, hidden size of 768, 12 attention heads, 125M parameters) which are trained for 100k updates using 8k batch-size, 20k warmup steps, 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 the FairSeq (Ott et al., 2019) toolkit for the pre-training and fine-tuning experiments.

3.2 Fine-tuning tasks

To evaluate the impact of word order during pre-training, we use the General Language Understanding and Evaluation (GLUE) benchmark, Paraphrase Adversaries from Word Scrambling (PAWS) dataset, as well as 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)[2], Multi-Genre NLI (MNLI) (Williams et al., 2017), 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

[1] `https://spacy.io/`

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

Model	QNLI	RTE	QQP	SST-2	MRPC	PAWS	MNLI-m/mm	CoLA
$\mathcal{M}_{\text{N}}$	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
$\mathcal{M}_1$	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
$\mathcal{M}_2$	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
$\mathcal{M}_3$	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
$\mathcal{M}_4$	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
$\mathcal{M}_{\texttt{RI}}$	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 +/- 1.60
$\mathcal{M}_{\text{NP}}$	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
$\mathcal{M}_{\texttt{UF}}$	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
$\mathcal{M}_{\texttt{UG}}$	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 +/- 2.10

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 ablations.

insensitivity of several GLUE tasks (QQP, SST-2), evaluated on publicly released pre-trained checkpoints. We additionally aim to determine whether those models learn about word order during pre-training or during fine-tuning.

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 inherent 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 $[1e^{-5}, 2e^{-5}, 3e^{-5}]$ and batch sizes $[16, 32]$. For pre-training, we train three different runs with three seeds. In our fine-tuning experiments, we select one of these at random, and for our probing experiments (in §5) we use all three. For each hyperparameter configuration, we train using 5 different seeds and report the mean and standard deviation on all tasks (Table 1).

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.

4.1 Word order-permuted pre-training

In our first set of experiments, we finetune the pre-trained models on the GLUE and PAWS tasks. Table 1 shows the results.[3] First, we observe that the model without access distributional or word order information, $\mathcal{M}_{\texttt{UG}}$ (corpus randomization) performs much worse than $\mathcal{M}_{\text{N}}$ overall: $\mathcal{M}_{\texttt{UG}}$ is 18 points worse than $\mathcal{M}_{\text{N}}$ on average across the accuracy-based tasks in Table 1, and has essentially no correlation with human judgments on CoLA. However, for some tasks, the inductive bias of $\mathcal{M}_{\texttt{UG}}$ alone is enough to reach surprisingly high scores, such as 85.6 on QQP, 83.2 on SST-2, and 71.9 on MNLI. Other baselines such as $\mathcal{M}_{\texttt{RI}}$, $\mathcal{M}_{\text{NP}}$ and $\mathcal{M}_{\texttt{UF}}$ (Table 1, bottom half) also perform significantly worse on GLUE and PAWS tasks, compared to $\mathcal{M}_{\text{N}}$. Since $\mathcal{M}_{\texttt{UG}}$ and $\mathcal{M}_{\texttt{UF}}$ have access to (bags of) words and word phrases (from NER), they perform significantly better on MNLI. However, access to these extra word phrases does not appear to be beneficial for other tasks. For the majority of tasks, the difference between $\mathcal{M}_{\text{NP}}$ and $\mathcal{M}_{\texttt{RI}}$ is small - a purely bag of words model performs comparably to a randomly intialized 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 $\mathcal{M}_{\text{N}}$ on the accuracy-based tasks, and within 0.3 points of $\mathcal{M}_{\text{N}}$ on QQP. Even on PAWS, which was designed to require knowledge of word order, $\mathcal{M}_1$ is within 5 points of $\mathcal{M}_{\text{N}}$. 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 $\mathcal{M}_{\text{N}}$

[3]The $\mathcal{M}_{\text{N}}$ 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. We restrict our experiments to 16GB BookWiki corpus and 100K updates for computational reasons, mirroring the RoBERTa ablations.

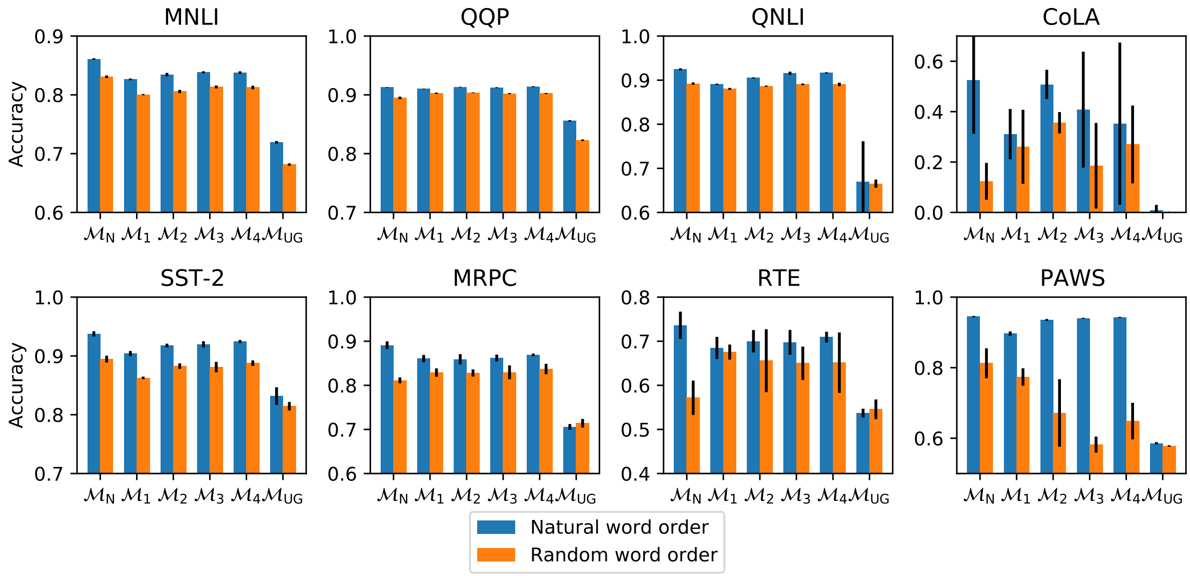

Figure 1: GLUE & PAWS task dev performance when finetuned on naturally and randomly ordered text, respectively, using pre-trained RoBERTa (base) models trained on different versions of BookWiki corpus.

by only 1.3 points on average, and even comes within 0.2 points of $\mathcal{M}_{\text{N}}$ on PAWS. We observe a somewhat different pattern on CoLA, where $\mathcal{M}_2$ does almost as well as $\mathcal{M}_{\text{N}}$ and outperforms $\mathcal{M}_3$ and $\mathcal{M}_4$, though we also observe very high variance across random seeds for CoLA. Crucially, we observe $\mathcal{M}_1$ to outperform $\mathcal{M}_{\text{NP}}$ by a large margin. This shows that positional embeddings are critical for MLM to learn, even when the word orders themselves are not natural. Overall, these results confirm our hypothesis that RoBERTa's good performance on downstream tasks can be largely explained 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 solve them; or they do need some word order information, but that information can be acquired during fine-tuning. To distinguish between these two hypotheses, 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 when models are fine-tuned on shuffled data, suggesting that word order is not very important for these tasks. Models can also achieve above 80% accuracy on MNLI, SST-2, and MRPC when fine-tuned on shuffled data, suggesting that purely lexical information is quite useful on its own for these tasks.[4]

On the other hand, for the other six datasets, 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. 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.

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. In contrast, since $\mathcal{M}_1$

[4]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.

never learns to be sensitive to word order during pre-training or fine-tuning, it does not suffer from that issue.

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 examine the language model's predictions via similarity.

5.1 Parametric Probing

To probe the array of pre-trained models for syntactic, semantic and other linguistic properties, we investigate dependency parsing using Pareto probing (Pimentel et al., 2020a) as well as the probing tasks from Conneau et al. (2018) available in the SentEval toolkit (Conneau and Kiela, 2018).

5.1.1 Pareto Probing

Pimentel et al. (2020a) proposed probing for given linguistic information in contextual representations using Pareto optimality. 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), as well as the "easy" probes: dependency arc labeling (DAL) and POS tag prediction (POS). For all tasks, we probe with Linear and MLP probes, and inspect the task accuracy in terms of Unlabeled Attachment Score (UAS; for DEP) and accuracy (for POS and DAL). The dependency parsing probe used in Pimentel et al. (2020a) builds on the Biaffine Dependency Parser (Dozat and Manning, 2016), but with the complexity of the probe reduced by removing the LSTM layer in order to restrict access to context. This probe consists of two identical MLPs - one to process the heads of the dependencies, and another to process the tails. The final biaffine transformation is simply a learned mapping function among output of the MLPs which is normalized to obtain the probabilities.[5]

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 running on three seeds of the pre-trained models. Since Pimentel et al. (2020a) explored multi-lingual corpora, they only experimented with a single English dataset, derived from Universal Dependencies EWT (UD EWT) (Bies et al., 2012; Silveira et al., 2014) containing 12,543 training sentences. Additionally, we experiment on the Penn Treebank dataset (PTB), which contains 39,832 training sentences.[6] We report the mean test accuracy over three seeds for the best dev set accuracy for each task.

Results. For DEP, we observe that the UAS scores also follow a linear trend as the fine-tuning results in that $\mathcal{M}_{\text{UG}} \approx \mathcal{M}_1 < \mathcal{M}_2 < \mathcal{M}_3 < \mathcal{M}_4 < \mathcal{M}_{\text{N}}$ (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. Interestingly, the performance gap heavily depends on the task and the probe chosen. In the case of UD EWT, the gap of UAS between $\mathcal{M}_1$ and $\mathcal{M}_{\text{N}}$ is higher for the MLP probe than that of the counterpart in PTB

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
$\mathcal{M}_1$	69.26 +/- 6.00	56.24 +/- 5.05	79.43 +/- 0.96	57.20 +/- 2.76
$\mathcal{M}_2$	78.22 +/- 0.88	64.96 +/- 2.32	84.72 +/- 0.55	64.69 +/- 2.50
$\mathcal{M}_3$	77.80 +/- 3.09	64.89 +/- 2.63	85.89 +/- 1.01	66.11 +/- 1.68
$\mathcal{M}_4$	78.04 +/- 2.06	65.61 +/- 1.99	85.62 +/- 1.09	66.49 +/- 2.02
$\mathcal{M}_{\text{UG}}$	74.15 +/- 0.93	65.69 +/- 7.35	80.07 +/- 0.79	57.28 +/- 1.42

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

.

Model	UD EWT		PTB	
	MLP	Linear	MLP	Linear
$\mathcal{M}_{\text{N}}$	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
$\mathcal{M}_{\text{UG}}$	89.10 +/- 0.21	79.75 +/- 0.5	94.12 +/- 0.01	84.15 +/- 0.51

Table 3: Accuracy on the part-of-speech labelling task (POS) on two datasets, UD EWT and PTB, using the Pareto Probing framework (Pimentel et al., 2020a).

[5] We experimented with a much stronger, state-of-the-art Second order Tree CRF Neural Dependency Parser (Zhang et al., 2020), although we were unable to observe any difference in UAS with different pre-trained models (detailed discussion in Appendix F)

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

Model	UD EWT		PTB	
	MLP	Linear	MLP	Linear
$\mathcal{M}_{\mathrm{N}}$	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
$\mathcal{M}_{\mathrm{UG}}$	84.93 +/- 0.34	76.30 +/- 0.52	89.98 +/- 0.43	78.59 +/- 0.68

Table 4: Accuracy on the dependency arc labelling task (DAL) on two datasets, UD EWT and PTB, using the Pareto Probing framework (Pimentel et al., 2020a).

(11.15 vs. 7.56). This trend is also mildly reflected in the case of the Linear probe (10.02 vs. 9.27 UD EWT). However, we observe a low performance gap in several scenarios, the lowest being between $\mathcal{M}_{\mathrm{N}}$ vs. $\mathcal{M}_3$, for PTB dataset using MLP probe (86.99 vs. 85.89 UAS).

For POS (Table 3) and DAL (Table 4), since these tasks are simpler than DEP, the gap between $\mathcal{M}_{\mathrm{N}}$ and unnaturally pre-trained models reduces even more drastically. The gap between $\mathcal{M}_{\mathrm{N}}$ and $\mathcal{M}_1$ reduces to just 3.5 points on average for PTB in both POS and DAL.

5.1.2 SentEval Probes

We also utilize 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-investigate 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 linear and two-layer MLP 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. For all hyperparameters, we select the best performance based on the dev set, and report the test set accuracy.

Results. We provide the results in Table 5. The $\mathcal{M}_{\mathrm{N}}$ pre-trained model scored better than the unnatural word order models for only 1 out of 5 semantic tasks and in none of the lexical tasks. However, $\mathcal{M}_{\mathrm{N}}$ does score higher for 2 out of 3 syntactic tasks. Even for these two syntactic tasks, the gap among $\mathcal{M}_{\mathrm{UG}}$ and $\mathcal{M}_{\mathrm{N}}$ is much higher than $\mathcal{M}_1$ and $\mathcal{M}_{\mathrm{N}}$. These results show that while natural word order is useful for at least some probing tasks, the distributional prior of sentence word order randomized models alone is enough to achieve a reasonably high accuracy on syntax sensitive probing.

5.2 Non-Parametric Probing

Figure 2: The difference in word probabilities for stimuli in Marvin and Linzen (2018). 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).

The literature on parametric probing is currently struggling with many questions about how to probe effectively (Maudslay et al., 2020; Belinkov, 2021). From our results so far, it is unclear whether the parametric probing tasks are able to meaningfully separate models trained with corrupted sentence order from those trained with normal orders. Thus, we also investigate an array of non-parametric

Model	Length (Surface)	WordContent (Surface)	TreeDepth (Syntactic)	TopConstituents (Syntactic)	BigramShift (Syntactic)	Tense (Semantic)	SubjNumber (Semantic)	ObjNumber (Semantic)	OddManOut (Semantic)	CoordInversion (Semantic)
$\mathcal{M}_{\mathrm{N}}$	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
$\mathcal{M}_1$	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
$\mathcal{M}_2$	**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
$\mathcal{M}_3$	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
$\mathcal{M}_4$	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
$\mathcal{M}_{\mathrm{UG}}$	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 5: SentEval Probing (Conneau et al., 2018; Conneau and Kiela, 2018) results on different model variants.

probes (Linzen et al., 2016; Marvin and Linzen, 2018; Gulordava et al., 2018) as initially curated and analyzed by Goldberg (2019), and later re-purposed for GPT2 (Radford et al., 2019) by Wolf (2019). Since these probes do not contain any learnable parameters, they are called "non-parametric".

We consider a set of non-parametric probes that contains 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 correct word than to an incorrect one. Since both the correct and incorrect option occupy the same sentential position, they are deemed "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 then perform evaluation on sentences containing at least one noun between the verb and its subject, which are known as "agreement attractors". Gulordava et al. (2018) investigate semantic selectional features by replacing words with random substitutes from the same part-of-speech and inflection. The model then provides the probability of the correct inflection of the original verb. Finally, Marvin and Linzen (2018) construct minimal pairs of grammatical and ungrammatical sentences, and compute the probability assigned by the model to the words that differ (known as the "focus verb").

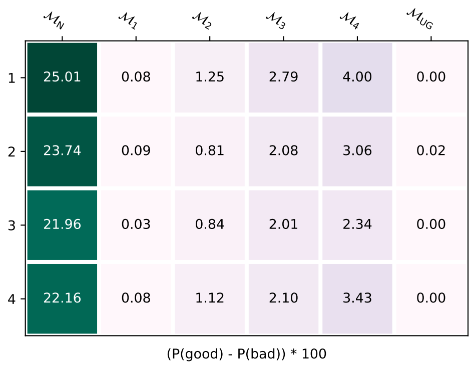

Figure 3: Linzen et al. (2016)

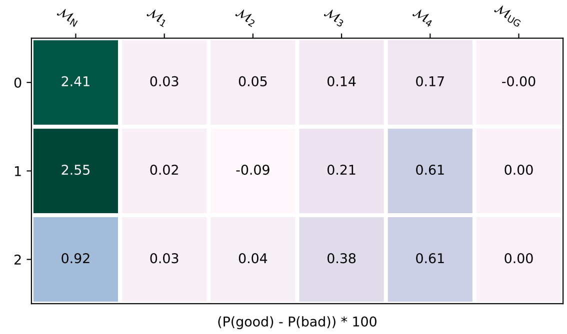

Figure 4: Gulordava et al. (2018)

Setup. In our experiments, we mask the focus words in the stimuli and compute the probability of the good and bad token respectively. In order to work around the Byte-Pair Encoding (BPE) token split, we use the WordPiece (Wu et al., 2016) tokens prepended with a space, as used in RoBERTa to compute the probability of a correct focus word and an incorrect focus word.

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 though focus word probability decreases drastically from $\mathcal{M}_{\mathrm{N}}$to $\mathcal{M}_{\mathrm{UG}}$(we also report the original accuracy metric in Appendix H). Thus, we report instead 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.

Results. For all three non-parametric probing datasets (Figure 2, Figure 3, and Figure 4) we observe the highest difference between probabilities of the correct and incorrect focus words for the model pretrained on the natural word order ($\mathcal{M}_{\mathrm{N}}$). Moreover, with each step from $\mathcal{M}_1$ to $\mathcal{M}_4$, the difference between probablities of correct and in-

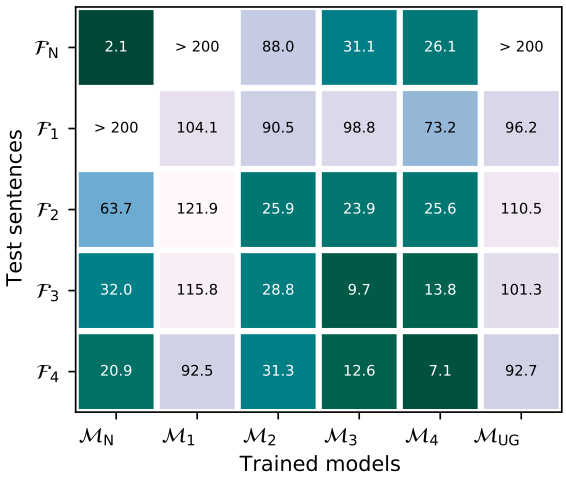

Figure 5: BPPL scores per model per test scenario.

correct focus words increases, albeit marginally, showing that pre-trained models with fewer n-gram words perturbed capture more syntax. $\mathcal{M}_{\text{UG}}$, with the distributional prior ablated, performs the worst, as expected.

Randomization models were found to be quite similar to $\mathcal{M}_{\text{N}}$ according to the parametric probes, but according to the non-parametric ones, they all are markedly worse than $\mathcal{M}_{\text{N}}$. This suggests that non-parametric probes are better at capturing syntactic distinctions that parametric probes may not pick up on.

6 Analysis

In this section, we perform further analysis to determine how much word order information is present in the RoBERTa representation after pre-training. Specifically, we first use perplexity as a proxy for measuring whether certain word order information is contained in the representation after pre-training. Secondly, we investigate the effect of the learned word order during the early stages of fine-tuning.

6.1 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:

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

We bootstrap the $\texttt{PLL}$ 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. (2020):

$$\texttt{BPLL}_T = \exp(-\frac{1}{N} \sum_{S \in W} \texttt{PLL}(S)), \quad (2)$$

where W is the combined bootstrap sample containing N sentences drawn with replacement from T. We compute this score on 5 pre-trained models ($\mathcal{M}_{\text{N}}$, $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, $\mathcal{M}_4$), 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 $\texttt{BPLL}$ scores, which we plot in Figure 5.

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). Interestingly, 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.

6.2 The usefulness of word order

The results in §4.1 suggest that with proper fine-tuning, an unnaturally trained model can reach comparable performance with a naturally pre-trained model. However, we want to understand whether natural word order pre-training offers some 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

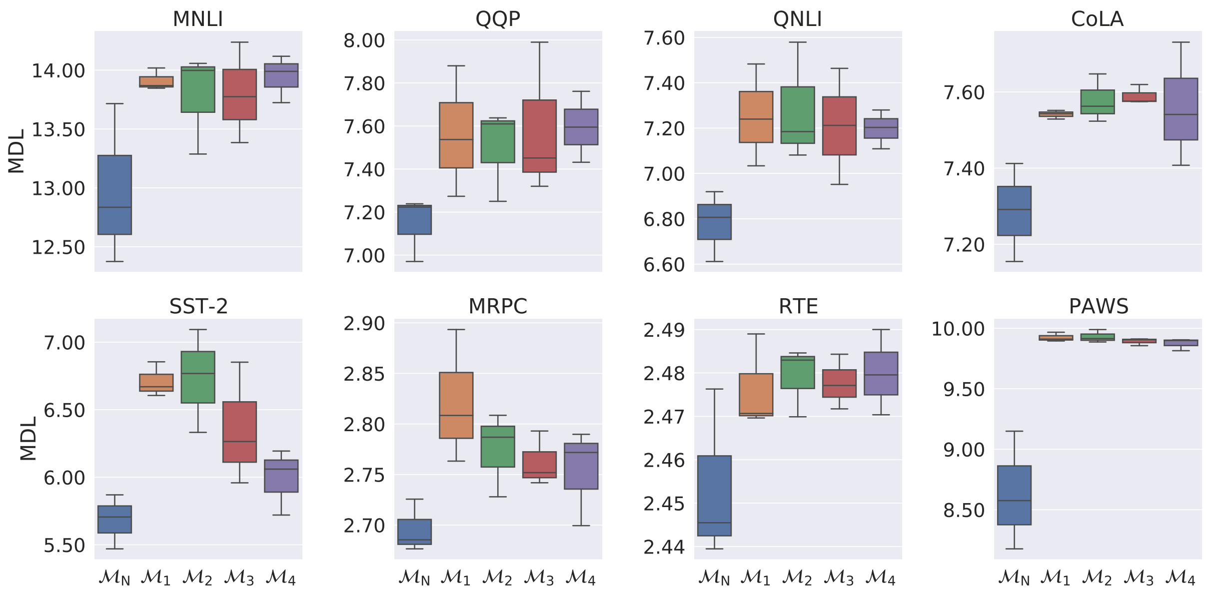

Figure 6: Rissanen Data Analysis (Perez et al., 2021) on the GLUE benchmark and PAWS datasets. The lower minimum description length (MDL, measured in K-bits), the better the learning ability of the model.

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 (coined as *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 6). 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., 2019) and few sample (Zhang et al., 2021) learning and evaluation could potentially require more word order signal, and thereby encouraging the model to leverage its own learned syntax better.

7 Discussion

The fine-tuning results in §4.1 and the parametric probing results in §5.1 suggest that MLM does not need to rely on the "classical NLP pipeline" for achieving high accuracy, assuming that such a pipeline would rely upon word order. 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, most linguistic constructs are undefined (e.g., dependency or constituency parses would no longer be syntactic trees, what would sentences be but mere lists of words).

One might ask, though, whether such 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? Even if the model were able to somehow learn to reorder or "unshuffle" the words under our unnatural pretraining set up, it could only do so based on distributional information. Models would then abductively learn only the most likely word order—as evidenced by the effectiveness of the distributional prior alone in achieving good enough fine-tuning and probing performance, high perplexity for randomized models on the original data, and our non-parametric probing results. Even if models infer a distribution over possible orders and use that information to structure their representations, 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 either we 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 alternatively, we need harder and more comprehensive evaluations (cf. Bowman and Dahl 2021), if we genuinely want to measure linguistic abilities, however those are defined, in our models.

Another 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 and/or more morphologically-rich languages would suffer from the same problem. One could imagine that the methods discussed in this work constitute a possible way for determining the degree of order-dependence for particular languages in general.

8 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, using targeted pre-training on sentences with various degrees of randomization in their word order. We 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. Instead, our experiments suggest that MLM's success can be mostly 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 learned representations.

Acknowledgements

We would like to thank Tiago Pimentel, Shruti Bhosale, Naman Goyal and Shagun Sodhani for their help in resolving technical doubts during experimentation and feedback on the manuscript. We thank Kyunghyun Cho, Ethan Perez, Myle Ott, Mona Diab, and Mike Lewis for their feedback on an earlier draft.

References

Guillaume Alain and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.

Trapit Bansal, Rishikesh Jha, and Andrew McCallum. 2019. Learning to few-shot learn across diverse natural language classification tasks. *arXiv preprint arXiv:1911.03863*.

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*.

Samuel R Bowman and George E Dahl. 2021. What will it take to fix benchmarking in natural language understanding? *arXiv preprint arXiv:2104.02145*.

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. *arXiv preprint arXiv:1803.05449*.

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. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

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*.

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.

Yoav Goldberg. 2019. Assessing BERT's Syntactic Abilities. 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. 2018. 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. *arXiv preprint arXiv:2101.03453*.

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*.

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.

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. *arXiv preprint arXiv:1902.10186*.

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.

Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Eisuke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh. 2021. Pre-training without Natural Images. *arXiv:2101.08515 [cs]*.

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.

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. *arXiv:1611.01368 [cs]*.

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. *arXiv:1808.09031 [cs]*.

Rowan Hall Maudslay, Josef Valvoda, Tiago Pimentel, Adina Williams, and Ryan Cotterell. 2020. A Tale of a Probe and a Parser. *arXiv:2005.01641 [cs]*.

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.

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 and Dan Jurafsky. 2020. Learning Music Helps You Read: Using Transfer to Study Linguistic Structure in Language Models. *arXiv:2004.14601 [cs]*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. Rissanen Data Analysis: Examining Dataset Characteristics via Description Length. *arXiv:2103.03872 [cs, stat]*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

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. *arXiv:2010.02180 [cs]*.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020b. Information-Theoretic Probing for Linguistic Structure. page 14.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

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. *arXiv:2002.12327 [cs]*.

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. *arXiv preprint arXiv:1301.2811*.

Sheng Shen, Alexei Baevski, Ari S. Morcos, Kurt Keutzer, Michael Auli, and Douwe Kiela. 2020. Reservoir Transformer. *arXiv:2012.15045 [cs]*.

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. *arXiv preprint arXiv:1905.05950*.

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 R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

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: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

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. *arXiv preprint arXiv:1901.10444*.

Adina Williams, Andrew Drozdov*, and Samuel R Bowman. 2018. 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 R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

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.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021. Revisiting Few-sample BERT Fine-tuning. *arXiv:2006.05987 [cs]*.

Yu Zhang, Zhenghua Li, and Zhang Min. 2020. Efficient second-order TreeCRF for neural dependency parsing. In *Proceedings of ACL*, pages 3295–3305.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. Paws: Paraphrase adversaries from word scrambling. *arXiv preprint arXiv:1904.01130*.

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):

$$p(t \mid w; \theta) = \frac{e^{f(t,w)}}{\sum_{t' \in V} e^{f(t',w)}} \quad (3)$$

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.

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 in which w occurs.

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	0.04 +/- 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 6: BLEU-2,3,4 scores on a sample of 1M sentences drawn from the corpus used to train $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$and $\mathcal{M}_4$ when compared to $\mathcal{M}_{\text{N}}$.

B Data generation

We provide a pseudo-code of $\mathcal{F}_i$ in Algorithm 1. Following Sinha et al. (2020), we do not explicitly control if the permuted words occur in the same neighborhood as their original neighbors. Thus, a certain amount of extra n-grams is expected to co-occur, which is purely a product of random shuffling. We verify the amount of such shuffling on a sample of 1 million sentences drawn from the BookWiki random corpus, and provide the BLEU-2, BLEU-3 and BLEU-4 scores in Table 6. We provide a sample snapshot of the generated data in Table 13.

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 FairSeq (Ott et al., 2019) toolkit to pre-train RoBERTa (base) models on the different variants of BookWiki corpus. We follow the default parameters as reported in Liu et al. (2019), with the

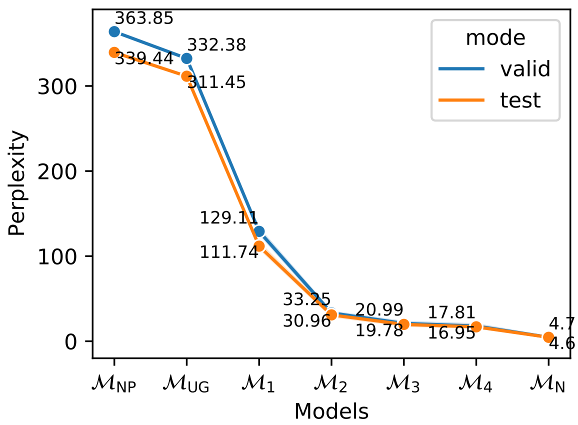

Figure 7: Perplexity of various models on Wiki 103 valid and test sets.

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
$\mathcal{M}_{\text{UG}}$	27.74	27.25	6.26	11.35	20.91	98.24	38.20	16.56
$\mathcal{M}_{\text{NP}}$	16.16	25.77	3.83	11.30	18.42	95.48	39.66	26.10

Table 7: $\Delta_{\{D_i\}}(\mathcal{T})$, scaled by a factor of 100 for GLUE and PAWS tasks.

Model	MRPC	RTE	PAWS	SST-2	CoLA	QNLI
$\mathcal{M}_{\text{N}}$	76.11 +/- 0.98	54.93 +/- 2.91	62.31 +/- 0.08	80.83 +/- 0.36	5.05 +/- 2.22	79.04 +/- 0.68
$\mathcal{M}_1$	81.24 +/- 1.28	65.53 +/- 3.57	62.63 +/- 0.1	83.84 +/- 0.35	23.42 +/- 6.49	87.29 +/- 0.17
$\mathcal{M}_2$	80.48 +/- 0.7	66.55 +/- 2.01	61.24 +/- 3.06	82.99 +/- 0.28	10.36 +/- 1.9	86.13 +/- 0.51
$\mathcal{M}_3$	79.51 +/- 0.92	64.95 +/- 2.99	62.27 +/- 0.09	83.0 +/- 0.2	9.0 +/- 5.15	84.68 +/- 1.23
$\mathcal{M}_4$	79.06 +/- 0.97	65.96 +/- 1.39	62.18 +/- 0.19	83.09 +/- 0.42	6.62 +/- 6.09	84.76 +/- 0.77

Table 8: Evaluation of the fine-tuned models on unigram randomized version of dev set for a subset of tasks in GLUE and PAWS.

following changes: max steps 100k, warmup 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 7). We observe that perplexity monotonically increases from $\mathcal{M}_{\text{N}}$, $\mathcal{M}_4$ to $\mathcal{M}_1$, $\mathcal{M}_{\text{UG}}$, $\mathcal{M}_{\text{NP}}$.

D Measuring Relative difference

In this section we further measure the difference of downstream tasks reported in §4.1 using a metric of *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 performance gap, i.e. how much the performance differs among the natural and unnatural models. Thus we define *Relative Difference* ($\Delta_{\{D\}}(\mathcal{T})$):

$$\Delta_{\{D\}}(\mathcal{T}) = \frac{\mathcal{A}(\mathcal{T}|OR) - \mathcal{A}(\mathcal{T}|D))}{\mathcal{A}(\mathcal{T}|OR) - \mathcal{A}(\mathcal{T}|\emptyset)}, \quad (4)$$

where $\mathcal{A}(\mathcal{T}|\emptyset)$ is the random performance on the task $\mathcal{T}$ (which is 0.33 for MNLI, 0 for CoLA, and 0.5 for rest) $\Delta_{\{D\}}(\mathcal{T}) \to 0$ 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 7. CoLA has the largest $\Delta_{\{D\}}(\mathcal{T})$ among all tasks, suggesting the high word order reliance of this task. $\Delta_{\{D\}}(\mathcal{T})$ is lowest for QQP, which falls on the opposite end of the spectrum.

E Fine-tuning evaluation on randomized data

We perform additional experiments using the fine-tuned models from §4.1. Specifically, we construct unigram randomized test sets of a subset of tasks to evaluate whether models fine-tuned on natural task data (having natural or unnatural pre-training prior) is able to understand unnatural data during testing. Sinha et al. (2020) show for MNLI there exists at least one permutation for many examples which can be predicted correctly by the model. However, they also show that every sentence can have many permutations which cannot be predicted correctly as well. Thus, in this evaluation, we construct 100 permutations for each example in the dev set for each task, to capture the overall accuracy. We observe that all models perform poorly on the unigram randomized test set, compared to results in Table 1. 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 62.31) 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.

F 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 code-

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 9: 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)

base. [7] We provide the results on UD EWT and PTB corpus in Table 9. Strangely enough, we find the gap to be even less in using 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 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 minimal gap among BERT-full and BERT-less models.

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

Results from §4.1 begs us to ask the question, when does pre-training model, if at all, learns the natural word order? We aim to answer that question by comparing downstream task performance of RoBERTa base on intermediate checkpoints with that of random word order pretrained models. The idea is to find *when* during pre-training on natural corpus the model exceeds task performance from the random pre-training models.

We observe that all tasks show rapid increase in performance during the first 20-25 epochs of pre-training. For some tasks, the word order information in the pre-training helps only after 30-50 epochs.

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

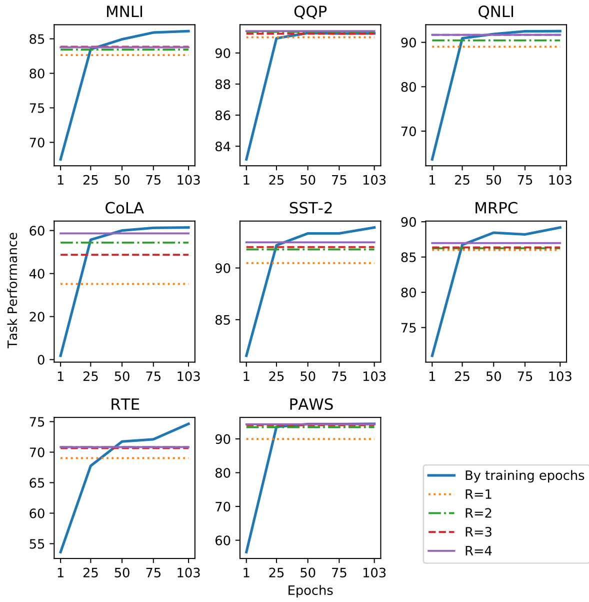

Figure 8: Comparison among GLUE task performance from different steps in pre-training of RoBERTa on BookWiki Corpus.

H Non parametric probes

We provide the accuracy as measured by Goldberg (2019); Wolf (2019) on the probing stimuli in Table 10, Table 12 and Table 11. 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 $\{OR, R_1, R_2, R_3, R_4, R_C\}$. However, the low token probabilities in $\mathcal{M}_{\text{UG}}$tend to present unreliable scores. For example, in the case of Gulordava et al. (2018) stimuli, unnatural models provide better scores compared to the natural model. We also observe for Linzen et al. (2016) stimuli, results on model condition 4 (number of attractors) is surprisingly high for $\mathcal{M}_{\text{UG}}$ wheras the individual token probabilities are lowest. We believe these inconsistencies stem from extremely low token probabilities themselves, and thus report on absolute probability difference in §5.2.

Model Condition	$\mathcal{M}_{\text{N}}$	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$	$\mathcal{M}_{\text{UG}}$
1	93.92 (2.5e+01)	62.1 (7.8e-02)	64.83 (1.3e+00)	65.27 (2.8e+00)	70.79 (4.0e+00)	63.14 (1.6e-03)
2	93.02 (2.4e+01)	62.86 (8.5e-02)	62.75 (8.1e-01)	65.08 (2.1e+00)	65.44 (3.1e+00)	71.98 (1.7e-02)
3	88.82 (2.2e+01)	62.74 (3.4e-02)	58.99 (8.4e-01)	63.34 (2.0e+00)	62.85 (2.3e+00)	75.71 (3.2e-03)
4	90.53 (2.2e+01)	63.16 (8.5e-02)	63.94 (1.1e+00)	66.41 (2.1e+00)	66.28 (3.4e+00)	80.54 (4.0e-03)

Table 10: Linzen et al. (2016) stimuli results in raw accuracy. Values in parenthesis reflect the $P(\text{good}) - P(\text{bad})$ metric.

Model Condition	$\mathcal{M}_{\text{N}}$	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$	$\mathcal{M}_{\text{UG}}$
0	72.22 (2.4e+00)	42.84 (3.2e-02)	52.46 (4.8e-02)	55.56 (1.4e-01)	56.94 (1.7e-01)	35.26 (-1.4e-04)
1	77.08 (2.5e+00)	76.56 (1.9e-02)	72.4 (-8.9e-02)	71.88 (2.1e-01)	71.35 (6.1e-01)	59.38 (9.5e-04)
2	55.56 (9.2e-01)	81.48 (3.1e-02)	51.85 (4.0e-02)	62.96 (3.8e-01)	74.07 (6.1e-01)	77.78 (2.7e-04)

Table 11: Gulordava et al. (2018) stimuli results in raw accuracy. Values in parenthesis reflect the $P(\text{good})-P(\text{bad})$ metric.

Model Condition	$\mathcal{M}_{\text{N}}$	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$	$\mathcal{M}_{\text{UG}}$
AOR	90.02 (2.9e+01)	60.36 (1.7e+00)	66.45 (3.9e+00)	63.43 (2.5e+00)	61.25 (1.2e+00)	50.0 (4.2e-05)
AOR-T	77.34 (1.4e+01)	78.94 (1.5e-01)	52.24 (4.3e-01)	48.87 (2.5e-01)	56.62 (5.5e-01)	50.0 (-1.9e-04)
APP	89.86 (2.7e+01)	70.22 (6.8e-01)	53.33 (2.0e-01)	53.43 (7.9e-01)	60.74 (9.1e-01)	50.0 (-3.2e-03)
ARC	84.90 (1.2e+00)	62.45 (1.1e-01)	74.59 (7.0e-02)	67.51 (6.7e-02)	62.74 (3.4e-02)	50.0 (-1.1e-05)
ASR	87.08 (2.6e+01)	79.08 (1.9e+00)	81.73 (2.9e+00)	63.03 (1.4e+00)	67.34 (1.5e+00)	50.0 (-2.9e-03)
IOR	89.96 (4.0e-01)	56.32 (6.2e-03)	59.32 (3.8e-02)	70.42 (4.1e-02)	62.74 (8.4e-02)	50.0 (-1.5e-03)
IOR-T	74.21 (2.0e-01)	52.57 (6.2e-03)	57.30 (1.3e-02)	62.22 (6.2e-03)	55.33 (2.7e-02)	50.0 (-9.9e-04)
ISC	86.45 (5.3e+00)	67.88 (6.7e-02)	82.79 (4.8e-03)	77.78 (3.1e-01)	68.4 (5.9e-02)	50.0 (3.1e-04)
LVC	93.58 (2.7e+01)	70.42 (9.7e-02)	87.75 (1.4e+00)	85.25 (6.3e-01)	81.42 (6.6e-01)	50.0 (-3.3e-05)
SCM	88.66 (5.7e+00)	63.31 (1.2e-01)	82.27 (1.7e-01)	86.67 (2.6e+00)	80.07 (2.6e-01)	50.0 (1.7e-02)
SNP	95.39 (2.3e-01)	18.43 (-3.1e-03)	51.76 (3.0e-03)	77.51 (1.7e+00)	53.93 (8.2e-02)	50.0 (-3.1e-04)
SRX	89.88 (2.7e+00)	87.38 (1.0e-01)	91.07 (3.2e-01)	94.29 (2.4e-02)	91.43 (3.1e+00)	50.0 (2.0e-04)
SVA	95.28 (3.1e+01)	87.22 (5.7e-01)	88.61 (3.2e+00)	94.17 (6.1e+00)	89.72 (3.1e+01)	50.0 (2.2e-02)
SVC	97.45 (1.9e+01)	82.82 (8.4e-01)	83.94 (1.2e+00)	92.64 (6.4e+00)	80.74 (1.1e+01)	50.0 (2.8e-04)

Table 12: Marvin and Linzen (2018) stimuli results in raw accuracy. Values in parenthesis reflect the $P(\text{good}) - P(\text{bad})$ metric. 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).

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