Using Roark-Hollingshead Distance to Probe BERT's Syntactic Competence

Jingcheng Niu[RH] **Wenjie Lu**[R] **Eric Corlett**[R] **Gerald Penn**[RH]
University of Toronto[R] Vector Institute[H]
{niu,luwenjie,ecorlett,gpenn}@cs.toronto.edu

Abstract

Probing BERT's general ability to reason about syntax is no simple endeavour, primarily because of the uncertainty surrounding how large language models represent syntactic structure. Many prior accounts of BERT's agility as a syntactic tool (Clark et al., 2013; Lau et al., 2014; Marvin and Linzen, 2018; Chowdhury and Zamparelli, 2018; Warstadt et al., 2019, 2020; Hu et al., 2020) have therefore confined themselves to studying very specific linguistic phenomena, and there has still been no definitive answer as to whether BERT "knows" syntax.

The advent of *perturbed masking* (Wu et al., 2020) would then seem to be significant, because this is a parameter-free probing method that directly samples syntactic trees from BERT's embeddings. These sampled trees outperform a right-branching baseline, thus providing preliminary evidence that BERT's syntactic competence bests a simple baseline. This baseline is underwhelming, however, and our reappraisal below suggests that this result, too, is inconclusive.

We propose *RH Probe*, an encoder-decoder probing architecture that operates on two probing tasks. We find strong empirical evidence confirming the existence of important syntactic information in BERT, but this information alone appears not to be enough to reproduce syntax in its entirety. Our probe makes crucial use of a conjecture made by Roark and Hollingshead (2008) that a particular lexical annotation that we shall call RH distance is a sufficient encoding of unlabelled binary syntactic trees, and we prove this conjecture.

1 Introduction

BERT (Devlin et al., 2019) is a structurally rich system both because of its neural architecture and because of the knowledge it captures through pre-training. Many have studied the similarities between these architectures and linguistically motivated structures or processing pipelines through probing (Conneau et al., 2018; Jawahar et al., 2019; Tenney et al., 2019a; Zhu et al., 2022; Niu et al., 2022) to argue for an enigmatic "BERT revolution," into which large parts of previous research in computational linguistics are now subsumed.

In this paper, we present *RH Probe*[1], a novel encoder-decoder-based probing architecture that utilises Roark-Hollingshead (2008) syntactic distance (RH distance) to examine the overall capacity of BERT's comprehension of syntax.

We introduce this probing architecture because there still has been no definitive answer as to whether the knowledge that BERT acquires during its pre-training process can in fact serve as a replacement for a phrase-structure-based notion of syntax. Limited by their probing methodology, prior performance-based probing attempts only investigated individual, fine-grained linguistic phenomena, until Wu et al. (2020).

Wu et al. (2020) proposed *perturbed masking*, a parameter-free probing method that directly looks for traces of well-established linguistic structures such as constituency trees latent in BERT representations. The baseline used in Wu et al.'s (2020), a right-branching tree, is overly simplistic. Upon conducting a thorough reappraisal of their results, we found that their parameter-free constituency-tree parser only marginally outperforms a naïve, right-branching baseline, and their *impact matrix* which formed the basis of their analysis only has weak to no correlation with constituency tree information.

We believe Wu et al. (2020) has overly fixated on parameter-free probing. They want to address the criticism that, because traditional supervised probes introduce supervised data, we cannot directly attribute evidence of "knowledge" to the pre-trained base language model itself — perhaps

[1]The implementation and results of RH Probe are available online: https://github.com/frankniujc/rh_probe.

Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, pages 325 - 334

everything is learned *post hoc* by the classifier through the probe dataset (Hewitt and Liang, 2019). But this risk can be mitigated. In particular, the ablation study is still a valid experimental design for probing (Zhu et al., 2022).

In our view, the only satisfactory evidence of "knowledge" that there has ever been in artificial intelligence has been the ability to use the model in question to perform inference. The present case should be no different. What we need then is parsing, or some other derivative task, which is easy to perform given a sufficient grasp of syntactic structure and much more difficult to perform without it, but in an experimental setting in which several sources of information are provided as input. BERT's output is one such potential source of information. Ablation studies then take the form of removing one or more of those sources during training. We also need alternative encodings of that knowledge for the purposes of comparison.

Roark-Hollingshead distance (2008) annotates word tokens in sequence but has been conjectured to encode unlabelled syntactic constituency trees. It is used by PRPN (Shen et al., 2018) as an important component of its internal model of tree structure that the network develops in the course of learning to parse unsupervised. RH distances are an important bridge between token sequences (language model) and arboreal structure (traditional syntax).

Therefore, we propose RH Probe, an encoder-decoder-based probing architecture. With RH Probe, we conduct two experiments: (1) an *ablation probe* that observes whether the removal of a feature source during training (language model output, RH distance, or part-of-speech (POS) information for reference) can decrease the probe's performance; and (2) an *"attack" probe* that observes whether randomization of certain features during testing can cause the performance to drop. The results of these experiments suggest that word embeddings contain important syntactic information, but that this information alone is not enough to reproduce traditional syntactic representations such as phrase structure in their entirety. In our experiments, we have also not been able to find any correlation between the quality of the language model and the amount of syntactic information.

After surveying previous probing methods, including probability probes, performance-based probes, and Wu et al.'s (2020) parameter-free probe, we introduce our RH Probe architecture and probing task design, including the definition of RH distance. We prove Roark and Hollingshead's (2008) conjecture that RH distance encodes unlabelled binary syntactic trees, and then present two experimental trials that use RH Probe to study the syntactic information carried by various sources. Our analysis of these results broadly confirms the existence of important syntactic information within BERT, but militates against the conclusion that this information alone can reproduce syntax in its entirety.

2 Probing for Syntax in BERT

2.1 Probability Probing

Since language models estimate the probability distribution over sequences of tokens, it is natural to compare the probability of a syntactically well-formed sentence with a syntactically ill-formed sentence; and reason about a language model's knowledge of syntax based on how often it can correctly assign a higher probability to the well-formed sentence. This practice of "probing" can trace its roots to Pereira (2000) on pre-neural language models. The same method is used in various work (Clark et al., 2013; Lau et al., 2014; Marvin and Linzen, 2018; Chowdhury and Zamparelli, 2018; Warstadt et al., 2019, 2020; Hu et al., 2020) to study neural language models from Mikolov et al. (2013) to contemporary large-scale models on different syntactic features, ranging from long-distance dependencies to anaphora.

Although simple and intuitive, this method of probing suffers two primary difficulties. First, probability is not a particularly good reflection of syntactic well-formedness. Other features of the input sequence, such as sequence length, token frequency, semantics, social bias, and even punctuation, can affect a language model's score. Second, this method of probing does not reflect the modern usage of language models after BERT introduced the pretrain/finetune paradigm. Although the probability distribution is still useful for text generation, language models are often used to encode a sequence of tokens into a vector space. But analysing probability alone does not provide insight into whether linguistic structures are latent in the language model's representations.

2.2 Performance-based Probing

To better reflect this new pretrain/finetune paradigm, *performance-based* probes (Adi et al.,

2017; Conneau and Kiela, 2018; Hupkes and Zuidema, 2018; Jawahar et al., 2019; Hewitt and Liang, 2019; Tenney et al., 2019a,b; Pimentel et al., 2020; Zhu et al., 2022) were introduced. These probes introduce the linguistic feature of interest as an auxiliary task and train a supervised classifier (often referred to as a *probe classifier* or *diagnostic classifier*) that takes BERT's embeddings as input. If this bolt-on classifier acquires good performance, people have concluded that the language model contains the relevant linguistic knowledge.

Overall, these probes use small classifiers with simple architectures to avoid introducing extra variables into the probing process — there is no good in explaining a black box with another black box. As a consequence, the auxiliary linguistic task design is also restricted to be simplistic and fine-grained (see Niu et al. (2022) for a more detailed discussion).

But as Hewitt and Liang (2019) presciently ask: "When a probe achieves high accuracy on a linguistic task using a representation, can we conclude that the representation encodes linguistic structure, or has the probe just learned the task?"

2.3 Perturbed Masking

This question is particularly salient for Wu et al. (2020), who resort to parameter-free probing based on the pairwise impact between tokens in a sentence. This impact is computed as the distance (either Euclidean or an information-theoretic difference in distributions) between the BERT representations of the sentence with the first token masked (with `[MASK]`) and the sentence with both tokens masked. Wu et al. (2020) theorized that tokens in the same constituent have higher impact scores among each other than with tokens outside the constituent. Using this pairwise impact score, they devised a matrix-based top-down parsing algorithm (MART) to induce constituency tree structures from BERT. Given an input sentence and its token impact information, the algorithm recursively chooses a splitting position that separates the sentence into two parts with the highest average impact between intraconstituent tokens, until a binary tree emerges. By evaluating those binary trees as constituency trees, they observed a better performance than simple right-branching and left-branching baselines.

Reappraising Wu et al. (2020) The performance increases are unfortunately slight. As shown in Table 1, MART only outperforms the right-branching baseline by a 1.3/2.3% F1 increase, whereas using RH distances in place of BERT vectors leads to considerably better F1 scores. Furthermore, a comparison of MART-generated trees to right-branching trees (with right-branching trees set as the reference) shows that MART-generated trees more closely resemble right-branching trees than constituency trees (Table 2).

To further investigate Wu et al.'s (2020) hypothesized connection between pairwise token impact and constituency, we can use it as a proxy for RH distance (as we will show in further detail in Section 3). When a token has a high RH distance from its predecessor, it means the two tokens' common ancestor is higher up in the syntactic tree, and therefore they are not both in a same, lower constituent.

	MART	RB Tree	LB Tree	RH	Random
WSJ10	58.0	56.7	19.6	67.04	51.6
WSJ23	42.1	39.8	9.0	50.08	29.69

Table 1: Wu et al.'s (2020) MART F1 performance compared to two naïve baselines: right-branching (RB) trees and left-branching (LB) trees, and the substitution of RH distances (RH) or random vectors (Random) for BERT vectors within MART. MART barely outperforms the right-branching baseline with a 1.3/2.3% F1 increase on the two evaluation datasets.

	MART vs. Const. Tree	MART vs. RB Tree
WSJ10	58.0	78.6
WSJ23	42.1	56.1

Table 2: Parsing F1 comparison. The "Const. Tree" column shows the parsing F1 of MART evaluated with the original PTB annotations as gold standard. The "RB Tree" column shows the same evaluation, but with the right-branching trees as a baseline. MART generates trees more closely resembling right-branching trees than constituency trees.

Test Split	Direction	mean r	median r	macro r
WSJ10	t_{i-1}, t_i	0.3	0.365	0.159
	t_i, t_{i-1}	0.153	0.223	0.261
	sum	0.258	0.323	0.25
WSJ23	t_{i-1}, t_i	0.246	0.255	0.195
	t_i, t_{i-1}	0.195	0.218	0.213
	sum	0.259	0.273	0.242

Table 3: Correlation between pairwise token impact and constituent level (RH distance). Following Wu et al. (2020), we calculated the result on the WSJ10 and WSJ23 splits. The mean correlation (r) and median correlation between impact score and RH distance are reported. We can see weak to no correlation for both test splits.

So there should be a high anti-correlation between impact score and RH distance.

We report the Pearson correlation of these in Table 3. Wu et al.'s (2020) token impact matrix is asymmetrical. The impact of token t_i on t_{i-1} is different from the impact of t_{i-1} on t_i. Therefore we showed correlation results in three different ways: the impact of a token's predecessor on itself (t_{i-1}, t_i), the impact of a token on its predecessor (t_i, t_{i-1}), and the sum of the impacts in both directions (sum). We also used two methods of calculating Pearson's correlation. First, we compute the correlation between the impact scores and RH distances of every sentence and report the mean and median. Second, we compute the "macro" correlation between the impact score and RH distance of every pair of adjacent tokens. We can see there are weak to no positive correlations on both the WSJ10 and WSJ23 splits in either direction. We did not observe the expected, high negative correlation.

2.4 Discussion: Parameter-free Probing and Ablation Study

Parameter-free probing is not the only solution to Hewitt and Liang's (2019) criticism of performance-based probing. Ablation is widely recognised as a means of mitigating this issue. When we subtract a source of information or signal from the input to the probe classifier, decreased performance of the probe can be interpreted as good evidence that those removed features or signals contained task-relevant information to the neural network. While there is still an ongoing debate on how well the magnitude of this difference correlates to the relevance of the information (Hewitt and Liang, 2019; Pimentel et al., 2020; Zhu and Rudzicz, 2020), that does not change the validity of using performance decreases as evidence for the mere presence of knowledge.

Although parameter-free probing does not use supervised data, the clustering algorithm itself encodes rigid prior information about trees and what they should look like structurally (the Random scores of Table 1 can be construed as indications of syntactic information latent in the algorithm itself). Right-branching trees satisfy these constraints, and MART-generated trees are similar to right-branching trees. We propose to reinstate a supervised finetuning regimen, in which BERT's presence in the input can be compared to other lexicalized encodings of syntactic structure. Sequences of RH distances are one such encoding. POS tag sequences can arguably serve as an imperfect, indirect encoding of the same.

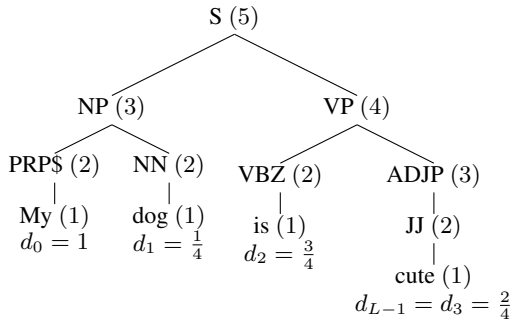

Figure 1: An RH distance calculation example. The height of nodes (h) are annotated in brackets.

3 RH Probe

3.1 Background: RH Distance

Before introducing our probing architecture, we need to introduce RH distance. Given a sentence $[t_0, t_1, \ldots, t_{L-1}]$, RH distance d_i is a measure of the syntactic distance between a token t_i and its predecessor t_{i-1}. It is calculated based on the height of the lowest common ancestor[2] of t_i and t_{i-1}. The precise calculations of both height and RH distances used in PRPN are somewhat parochial, but we adhere to them in equation 1 and figure 1 (r is the root of the tree) because this application of RH distance is perhaps the one best known to this audience.

$$\begin{aligned} d_i &= \frac{h(t_{i-1}, t_i) - 2}{h(r) - 1} \\ h(t_{-1}, t_0) &= h(t_{L-1}, t_L) = h(r) + 1 \\ h(u, v) &= h(u \cup v), \text{everywhere else} \end{aligned} \tag{1}$$

With RH distance information $[d_0, d_1, \ldots, d_L]$, it is sufficient to reconstruct the structure of an entire binary constituency tree. This was conjectured by Roark and Hollingshead (2008), but remained unproven until now. We will prove it in Section 3.5. Sequences of RH distance can be regarded as the encoding of an entire unlabelled constituency tree, and we can use this linear encoding of arboreal information to form the basis of our RH Probe architecture.

[2]The lowest common ancestor of two tree nodes u and v is denoted as $u \cup v$.

3.2 Probe Architecture

RH distance is a lexicalized measure, a sequence of which encodes the structure of certain constituency trees. We can use that information to create a series of probes into BERT's "understanding" of syntax. RH Probe has two modes of operation: an *ablative probe* observes the classifier's performance decrease when an input source is removed from training; and an *"attack" probe* observes its performance decrease after input features are obscured through randomization during testing (after a noise-free training).

3.3 Ablative Probe

The successful generation of a constituency tree is simple given the necessary structural information (RH distance), POS tags and internal node labels. Our ablation probe aims to determine whether language models provide crucial syntactic information in the presence of either RH distance or POS tags, neither of which directly encodes the labels of internal nodes.

To avoid providing structural hints to the probe in the decoder itself, we implemented an encoder-decoder probing architecture as shown in Figure 2. The output of the probe is a flattened version of the PTB parse trees. The terminal tokens are dropped to simplify the output space. The input is a concatenation of all the chosen input feature sources (word embedding, RH distance, POS tag embedding), as shown in Figure 3. For the choice of word embeddings, we use two different sizes of BERT, and also word2vec (Mikolov et al., 2013) as well as no embedding. word2vec vectors (300 dimensions) are significantly smaller than bert-base-cased vectors (784 dimensions), and so the 256-dimensional BERTtiny (Bhargava et al., 2021) was included for reference. When POS tags are provided, they are vectorised by the same embedding as the decoder, as POS tags are a subset of constituent tags.

Evaluation Metrics We want to evaluate whether the probe classifier can correctly predict the structure of the tree, as well as the pre-terminal and phrasal labels. Two of these three aspects correspond to possible input sources. Nevertheless, it is impossible to evaluate the quality of the labels completely independently of the structure. Therefore, we instead measure quality in two ways:

- **Tree Integrity** the average of a binary variable that is 1 iff the decoder's output can be successfully evaluated as a tree. This includes bracket balancing and constituent label location, but not the labels themselves.
- **Unlabelled Exact Match Accuracy** the average of a binary variable that is 1 iff the produced sequence is exactly correct, ignoring labels.

The overall quality of the trees is measured through:

- **Levenshtein (1965) distance** (the average of) the minimum number of insertions, deletions, or replacements of output tokens (including left or right brackets and constituent labels) to edit the decoder output to the the gold standard, normalised by the gold standard's sequence length.
- **Labelled Tree Exact Match Accuracy** the average of a binary variable that is 1 iff the produced sequence is exactly correct, including labels.

3.4 Attack Probe

The ablation task is not a probe into the disjointness of knowledge between two sources of information nor a probe into the relative weights of each information source. Therefore, we also introduce an *Attack Probe*. Figure 4 depicts this process. With a fully trained probe classifier, we evaluate the classifier's performance on perturbed input. By replacing one of the information sources (RH distance, word embeddings, or POS embeddings) with random noise, we can observe by how much the classifier's performance drops. A bigger performance drop can be interpreted as higher feature importance. The evaluation method is exactly the same as in the ablative task.

3.5 Proof of RH Conjecture

It can be proved that RH sequences uniquely encode unlabelled binary trees with no unary branches above the pre-terminal connections that they implicitly assume to be present at every word token. Roark and Hollingshead (2008) were the first to speculate about the expressive capacity of these sequences to encode syntactic trees. The definition of RH sequences itself provides a unique, constructive recipe for translating any such tree into an RH sequence. As for the reverse direction, the proof proceeds by induction on $h(r)$, the height of the root node. The base case is a single word, with a single pre-terminal, which is also the root, at height 2. This is encoded by a single RH distance of 1, which is the only length-1 RH sequence that can exist, and its tree is the only tree that can exist

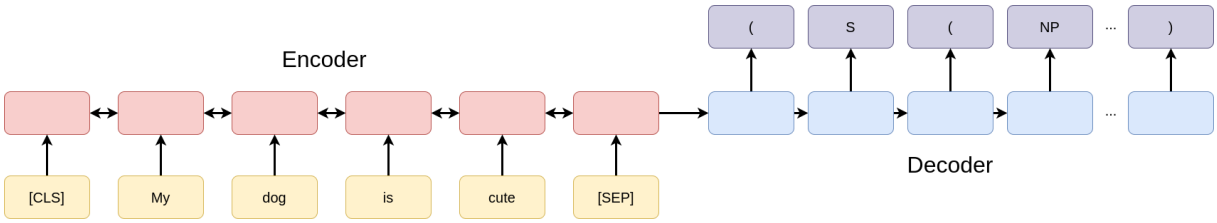

Figure 2: RH Probe Encoder-Decoder Architecture. The input features are fed into a bidirectional encoder, and the encoded representation of the sentence is decoded by a unidirectional decoder. The decoder outputs the flattened tree with all of the pre-terminal POS tags but no terminal tokens. The constituency tree of the example sentence *my dog is cute*, displayed in figure 1, will be flattened into `(S(NP(PRP$)(NN))(VP(VBZ)(ADJP(JJ))))`.

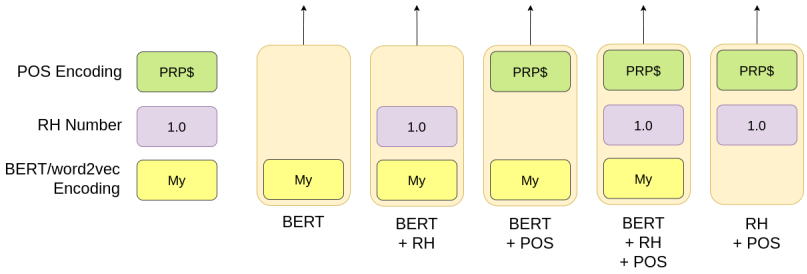

Figure 3: Ablation probe input selection. The input features are concatenated into one vector.

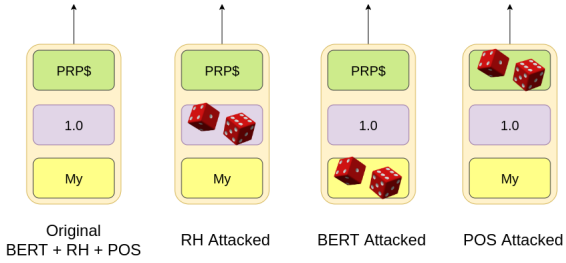

Figure 4: Attack Probe. Attack on different input means replace that input with random noise. Performance drop after this attack is then observed.

with a yield of 1, given the constraints on trees that we have assumed.

Assuming that all constrained trees of height $h(r) = n$ are uniquely encoded by RH sequences, we seek to show that any tree of height $h(r) = n+1$ is also uniquely encoded by reconstructing it from its RH sequence. Given the smallest number in a well-formed RH sequence, it is possible to infer the height of its corresponding tree. This number will be $\frac{1}{h(r)-1}$, since the witness to the height of the root $h(r)$ must contain a node of height 3, which, being of height greater than 2, must be the join of some adjacent pair of word tokens in our constrained trees. We posit joins of adjacent pairs in the tree that we are reconstructing wherever we see the number $\frac{1}{h(r)-1}$ on the second (right-hand) number of a pair. Note that no two adjacent word tokens can both have RH numbers of $\frac{1}{h(r)-1}$ in a well-formed RH sequence for our trees.

Let us now contemplate what the RH sequence of the remainder of the tree would look like. This remainder is itself a tree that satisfies our constraints, if we were to remove these adjacent, joined words, replacing them with a single, made-up token, and their join with a new, made-up POS tag, which would have height 2 rather than 3. Although it is not possible in general to predict what the new RH sequence would be if we were to remove one such pair from a tree, it is actually possible to predict what the new RH sequence would be if we removed all such pairs at once from the tree. This would remove all nodes of height 3. Every longest path to every node of height higher than 3 passes through exactly one height-3 node, which means that the root and the remaining joins in the new tree all decrement by exactly 1. So, without knowing the structure of the remainder of the tree yet, we can recalibrate the RH sequence to it: because we could infer $h(r)$ in the old tree, we can recover the unnormalized join heights by multiplying by $h(r)-1$, then decrement the unnormalized heights, and then renormalize using $h(r)-2$ rather than $h(r)-1$. This recalibrated sequence now corresponds to a tree of height one less, and so by the inductive hypothesis, its tree can be recovered.

Model	Tree Integrity	Unlabelled EM	Levenshtein	Labelled EM
BERT	0.124	0.0244	0.352	0.012
BERT + RH	0.177	0.0824	0.284	0.0269
BERT + POS	0.16	0.043	0.283	0.0344
BERT + RH + POS	0.181	0.055	0.268	0.0472
BERTtiny	0.11	0.0356	0.341	0.0161
BERTtiny + RH	0.168	0.065	0.301	0.0157
BERTtiny + POS	0.161	0.0464	0.293	0.0368
BERTtiny + RH + POS	0.168	0.0555	0.274	0.0435
word2vec	0.119	0.048	0.333	0.0178
word2vec + RH	0.179	0.0679	0.302	0.0236
word2vec + POS	0.18	0.0588	0.277	0.048
word2vec + RH + POS	0.209	0.0795	0.258	0.0642
RH	0.147	0.0774	0.322	0.0132
POS	0.162	0.0629	0.295	0.0509
RH + POS	0.165	0.0803	0.27	0.053

Table 4: Ablation probe result. For Levenshtein distance, the lower, the better. For the others, higher is better.

4 Experimental Setup

Constituency trees were obtained from the Penn Treebank (PTB) (Marcus et al., 1994). The corpus was processed and split into training/validation/test sets using Kim et al.'s (2019) scripts, available online[3]. Trees were binarized, as usual, but unary branches except for the pre-terminal connections were removed, in accordance with the limitations on the expressive capacity of sequences of RH distances.

A hyperparameter search was conducted using Optuna (Akiba et al., 2019) for 150 trials with `bert-base-cased` embeddings and no RH distances nor POS tags were provided as hints. The search result concluded with a 50 PTB-tag embedding size, a 0.0005 Adam (Kingma and Ba, 2015) learning rate, and a 5-layer GRU encoder-decoder with a 350 hidden-unit size. Finally, a beam search algorithm was implemented in the decoder with a width of 5. This set of hyperparameters was used for every probe conducted here. This hyperparameter selection process can be generalised to other large-scale pretrained language models.

5 Experimental Results

5.1 Ablation Probe Results

Table 4 shows our ablation probe results. Each probe was trained for 200 epochs and for each evaluation metric, we reported the test set performance on the epoch that has the best validation set performance. We note the following key findings:

Language models provide useful information for parsing. Performance is lower when no word embedding is provided (RH, POS, RH+POS). The removal of word embeddings derived from any type of language model decreases the probe's performance. This shows that language models can learn through pre-training information useful and recognizable to downstream syntactic tasks.

POS and RH distance provide performance increases across the board. With the labelled tree exact match ratio of BERTtiny *vs.* BERTtiny + RH being the only exception (0.04% performance increase), all settings when POS and RH distance information are removed result in a performance drop. Furthermore, we can observe a lower structural performance when RH distance is removed, and a lower overall performance when POS tags are removed. This is expected as RH distance only contains structural information. While language models contain useful syntactic information, the information they contain is still notably disjoint from syntax, as the removal of RH sequences and/or POS tags in the presence of BERT is also deleterious to performance.

Better language model $\neq$ more syntactic knowledge. Finally, the quality of the language model is not directly correlated to the probe's performance. If the central presupposition of performance-based probing is in fact true, that the magnitude of performance increases with the quantity of knowledge, then a better language model does not mean better understanding of syntax. BERTtiny and word2vec outperformed BERT(-base-cased) in several respects, and the hyperparameters were searched with the BERT model. While it may be controversial to compare the quality of BERT and word2vec, as it is not uncommon

[3] https://github.com/harvardnlp/compound-pcfg

Model	Attack	Tree Integrity		Unlabelled Acc		Levenshtein		Tree EM Acc	
		score	Δ	score	Δ	score	Δ	score	Δ
BERT + RH + POS	original	0.181	-	0.0675	-	0.263	-	0.0571	-
	attack RH	0.148	-0.0328	0.0261	-0.0414	0.334	0.0712	0.0219	-0.0352
	attack BERT	0.0803	-0.101	0.00455	-0.0629	0.467	0.205	0.0029	-0.0542
	attack POS	0.0629	-0.118	0.000828	-0.0666	0.516	0.253	0.0	-0.0571
BERTtiny + RH + POS	original	0.168	-	0.0642	-	0.273	-	0.0517	-
	attack RH	0.154	-0.0136	0.0435	-0.0207	0.319	0.0461	0.0373	-0.0145
	attack BERT	0.101	-0.0674	0.00745	-0.0567	0.407	0.134	0.00497	-0.0468
	attack POS	0.055	-0.113	0.000828	-0.0633	0.633	0.36	0.0	-0.0517
word2vec + RH + POS	original	0.209	-	0.0795	-	0.261	-	0.06	-
	attack RH	0.118	-0.0915	0.0207	-0.0588	0.369	0.108	0.0199	-0.0401
	attack w2v	0.115	-0.0935	0.00455	-0.0749	0.423	0.162	0.00207	-0.0579
	attack POS	0.0741	-0.135	0.000828	-0.0786	0.502	0.241	0.0	-0.06

Table 5: Attack probe results. The probe's test set performance after the attack is displayed in the score columns, and the magnitude of the performance drop is displayed in the Δ columns. All attacks are conducted on the epoch with the highest validation set tree integrity.

Metric	BERT	BERTtiny	word2vec
Tree Integrity	79.61%	54.17%	4.76%
Unlabelled EM	85.25%	84.47%	81.25%
Levenshtein	73.38%	28.05%	40.16%
Labelled EM	86.79%	86.67%	89.58%

Table 6: Relative "importance" of word embedding input, calculated as $\frac{S_{\text{embed}} - S_{\text{rh}}}{S_{\text{pos}} - S_{\text{rh}}}$, where S_{embed}, S_{pos}, S_{rh} represent the underlying measure after attacks on the respective information source.

to have simpler language modelling architectures outperforming more powerful ones (Edwards et al., 2020), BERT and BERTtiny share the same architecture.

5.2 Attack Probe Results

Table 5 presents our attack probe results. We trained each model for 200 epochs, found the epoch that gives the best tree-integrity performance on the validation set and then performed attack probing on that epoch. Some of the results are paradoxical. As with Section 5.1, language models apparently do not embody a complete account of syntactic information, but an attack on BERT brings a noticeably more adverse impact on performance than an attack, for example, on RH distance. This may be connected to the greater dispersion of BERT inputs over a larger dimensionality of input.

In other words, although RH + POS and BERT + RH + POS exhibit very close performance (Table 4) and RH is important for parsing, the model still weighs BERT more than RH distance. We can see in the second row of Table 5 that, over all measures, attacks on BERT always bring a more significant performance drop. This indicates that BERT's higher dimensionality makes information easier to extract better, increasing the model's generalizability to different downstream tasks. This advantage outweighs RH distance's provable adequacy as a representation of unlabelled binary syntactic trees.

We also note that the performance drop of attacking the word embeddings always stays between the performance drop of attacking the RH distance and that of attacking POS embeddings. However, this relative performance drop differs from language model to language model. Therefore, we calculate the relative importance of word embedding information as $\frac{S_{\text{embed}} - S_{\text{rh}}}{S_{\text{pos}} - S_{\text{rh}}}$, where S_{embed}, S_{pos}, and S_{rh} represent the measure S after attacks on the respective information source. As shown in Table 6, except for labelled EM, where every model gives similar scores that are very close to 0 (so the percentage is almost the same for every model), BERT has higher relative importance than the other two simpler language models. This indicates that although better language models do not necessarily contain more syntactic information, better language models will make information easier to pick out or extract for downstream classifiers, and therefore give better performance.

6 Conclusion

Does BERT know syntax? Our RH Probe results demonstrate that BERT contains important syntactic information, although this information alone cannot reproduce syntax in its entirety. Our empirical evidence is not subject to the criticism that the observed syntactic knowledge is not obtained by the language model through pretraining, but rather emerges from the probe classifier itself. The evidence is drawn from two carefully designed tasks, in which the substantial performance changes can be observed.

Nevertheless, we strongly agree with the insight (Wu et al., 2020) that the internal structure of BERT, although it may not embody a complete syntactic characterization of its input, can provide useful information for downstream applications. Recent probing attempts (Hewitt and Liang, 2019; Jawahar et al., 2019; Tenney et al., 2019a) have proved that the utility of knowledge acquired through pretraining is not limited to lexical semantics — word embedding also encodes syntax-adjacent information. Our probing results reinforce this argument through a more general examination of the relation between BERT and syntax.

References

Y. Adi, E. Kermany, Y. Belinkov, O. Lavi, and Y. Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *Proceedings of ICLR*. OpenReview.net.

T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of ACM KDD*, pages 2623–2631.

P. Bhargava, A. Drozd, and A. Rogers. 2021. Generalization in NLI: Ways (Not) To Go Beyond Simple Heuristics. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 125–135. ACL.

S.A. Chowdhury and R. Zamparelli. 2018. RNN Simulations of Grammaticality Judgments on Long-distance Dependencies. In *Proceedings of COLING*, pages 133–144.

A. Clark, G. Giorgolo, and S. Lappin. 2013. Statistical Representation of Grammaticality Judgements: The Limits of N-Gram Models. In *Proceedings of the Fourth Annual Workshop on Cognitive Modeling and Computational Linguistics (CMCL)*, pages 28–36. ACL.

A. Conneau and D. Kiela. 2018. SentEval: An Evaluation Toolkit for Universal Sentence Representations. In *Proceedings of LREC*, pages 1699–1704.

A. Conneau, G. Kruszewski, G. Lample, L. Barrault, and M. Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of ACL*, pages 2126–2136.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL*, pages 4171–4186.

A. Edwards, J. Camacho-Collados, H. De Ribaupierre, and A. Preece. 2020. Go Simple and Pre-Train on Domain-Specific Corpora: On the Role of Training Data for Text Classification. In *Proceedings of COLING*, pages 5522–5529.

J. Hewitt and P. Liang. 2019. Designing and Interpreting Probes with Control Tasks. In *Proceedings of EMNLP*, pages 2733–2743.

J. Hu, S. Chen, and R. Levy. 2020. A Closer Look at the Performance of Neural Language Models on Reflexive Anaphor Licensing. *Proceedings of SCiL*, 3(1):382–392.

D. Hupkes and W. Zuidema. 2018. Visualisation and 'Diagnostic Classifiers' Reveal how Recurrent and Recursive Neural Networks Process Hierarchical Structure (Extended Abstract). In *Proceedings of IJCAI*, pages 5617–5621.

G. Jawahar, B. Sagot, and D. Seddah. 2019. What Does BERT Learn about the Structure of Language? In *Proceedings of ACL*, pages 3651–3657.

Y. Kim, C. Dyer, and A. Rush. 2019. Compound Probabilistic Context-Free Grammars for Grammar Induction. In *Proceedings of ACL*, pages 2369–2385.

D.P. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

J.H. Lau, A. Clark, and S. Lappin. 2014. Measuring Gradience in Speakers' Grammaticality Judgements. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 36:821–826.

V.I. Levenshtein. 1965. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics. Doklady*, 10:707–710.

M. Marcus, G. Kim, M.A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. 1994. The Penn Treebank: Annotating Predicate Argument Structure. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 8-11, 1994*, pages 114–119.

R. Marvin and T. Linzen. 2018. Targeted Syntactic Evaluation of Language Models. In *Proceedings of EMNLP*, pages 1192–1202.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of ICLR*.

J. Niu, W. Lu, and G. Penn. 2022. Does BERT rediscover a classical NLP pipeline? In *Proceedings of COLING*, pages 3143–3153.

F. Pereira. 2000. Formal grammar and information theory: Together again? *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 358(1769):1239–1253.

T. Pimentel, J. Valvoda, R. Hall Maudslay, R. Zmigrod, A. Williams, and R. Cotterell. 2020. Information-Theoretic Probing for Linguistic Structure. In *Proceedings of ACL*, pages 4609–4622.

B. Roark and K. Hollingshead. 2008. Classifying Chart Cells for Quadratic Complexity Context-Free Inference. In *Proceedings of COLING*, pages 745–752.

Y. Shen, Z. Lin, C. Huang, and A. Courville. 2018. Neural Language Modeling by Jointly Learning Syntax and Lexicon. In *Proceedings of ICLR*.

I. Tenney, D. Das, and E. Pavlick. 2019a. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of ACL*, pages 4593–4601.

I. Tenney, P. Xia, B. Chen, A. Wang, A. Poliak, R.T. McCoy, N. Kim, B. Van Durme, S.R. Bowman, D. Das, and E. Pavlick. 2019b. What do you learn from context? Probing for sentence structure in contextualized word representations. In *Proceedings of ICLR*.

A. Warstadt, A. Parrish, H. Liu, A. Mohananey, W. Peng, S.-F. Wang, and S.R. Bowman. 2020. BLiMP: The Benchmark of Linguistic Minimal Pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

A. Warstadt, A. Singh, and S.R. Bowman. 2019. Neural Network Acceptability Judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Z. Wu, Y. Chen, B. Kao, and Q. Liu. 2020. Perturbed Masking: Parameter-free Probing for Analyzing and Interpreting BERT. In *Proceedings of ACL*, pages 4166–4176.

Z. Zhu and F. Rudzicz. 2020. An information theoretic view on selecting linguistic probes. In *Proceedings of EMNLP*, pages 9251–9262.

Z. Zhu, S. Shahtalebi, and F. Rudzicz. 2022. Predicting fine-tuning performance with probing. In *ACL BigScience Workshop*.