A Tale of a Probe and a Parser

Rowan Hall Maudslay, Josef Valvoda, Tiago Pimentel, Adina Williams, Ryan Cotterell

University of Cambridge, Facebook AI Research, ETH Zürich
rh635@cam.ac.uk, jv406@cam.ac.uk, tp472@cam.ac.uk, adinawilliams@fb.com, ryan.cotterell@inf.ethz.ch

Abstract

Measuring what linguistic information is encoded in neural models of language has become popular in NLP. Researchers approach this enterprise by training “probes”—supervised models designed to extract linguistic structure from another model’s output. One such probe is the structural probe (Hewitt and Manning, 2019), designed to quantify the extent to which syntactic information is encoded in contextualised word representations. The structural probe has a novel design, unattested in the parsing literature, the precise benefit of which is not immediately obvious. To explore whether syntactic probes would do better to make use of existing techniques, we compare the structural probe to a more traditional parser with an identical lightweight parameterisation. The parser outperforms structural probe on UUAS in seven of nine analysed languages, often by a substantial amount (e.g. by 11.1 points in English). Under a second less common metric, however, there is the opposite trend—the structural probe outperforms the parser. This begs the question: which metric should we prefer?

1 Introduction

Recently, unsupervised sentence encoders such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) have become popular within NLP. These pre-trained models boast impressive performance when used in many language-related tasks, but this gain has come at the cost of interpretability. A natural question to ask, then, is whether these models encode the traditional linguistic structures one might expect, such as part-of-speech tags or syntactic trees. To this end, researchers have invested in the design of diagnostic tools commonly referred to as probes (Alain and Bengio, 2017; Conneau et al., 2018; Hupkes et al., 2018; Poliak et al., 2018; Marvin and Linzen, 2018; Niven and Kao, 2019). Probes are supervised models designed to extract a target linguistic structure from the output representation learned by another model.

Based on the authors’ reading of the probing literature, there is little consensus on where to draw the line between probes and models for performing a target task (e.g. a part-of-speech tagger versus a probe for identifying part-of-speech). The main distinction appears to be one of researcher intent: probes are, in essence, a visualisation method (Hupkes et al., 2018). Their goal is not to best the state-of-the-art, but rather to indicate whether certain information is readily available in a model—probes should not “dig” for information, they should just expose what is already present. Indeed, a sufficiently expressive probe with enough training data could learn any task (Hewitt and Liang, 2019), but this tells us nothing about a representation, so it is beside the point. For this reason, probes are made “simple” (Liu et al., 2019), which usually means they are minimally parameterised.

Syntactic probes, then, are designed to measure the extent to which a target model encodes syntax. A popular example is the structural probe (Hewitt and Manning, 2019), used to compare the syntax that is decodable from different contextualised word embeddings. Rather than adopting methodology from the parsing literature, this probe utilises a novel approach for syntax extraction. However, the precise motivation for this novel approach is not immediately clear, since it has nothing to do with model complexity, and appears orthogonal to the goal of a probe. Probes are designed to help researchers understand what information exists in a model, and unfamiliar ways of measuring this information may obscure whether we are actually gaining an insight about the representation we wish to measure, or the tool of measurement itself.

[^1]: An information-theoretic take on probe complexity is the subject of concurrent work; see Pimentel et al. (2020).
We observe that the syntactic distance between dis-
pleasure and everything is 2 (the red path).

Using the structural probe as a case study, we
explore whether there is merit in designing models
specifically for the purpose of probing—whether
we should distinguish between the fundamental
design of probes and models for performing an
equivalent task, as opposed to just comparing their
simplicity. We pit the structural probe against a
simple parser that has the exact same lightweight
parameterisation, but instead employs a standard
loss function for parsing. Experimenting on multi-
tilingual BERT (Devlin et al., 2019), we find that
in seven of nine typologically diverse languages
studied (Arabic, Basque, Czech, English, Finnish,
Japanese, Korean, Tamil, and Turkish), the parser
boosts UUAS dramatically (by, for example, 11.1
points in English).

In addition to using UUAS, Hewitt and Manning
(2019) also introduce a new metric—correlation
of pairwise distance predictions with the gold stan-
dard. We find that the structural probe outperforms
the more traditional parser substantially in terms
of this new metric, but it is unclear why this met-
ric matters more than UUAS. In our discussion,
we contend that, unless a convincing argument to
the contrary is provided, traditional metrics are
preferable. Justifying metric choice is of central
importance for probing, lest we muddy the waters
with a preponderance of ill-understood metrics.

2 Syntactic Probing Using Distance

Here we introduce syntactic distance, which we
will later train a probe to approximate.

Syntactic Distance The syntactic distance be-
tween two words in a sentence is, informally,
the number of steps between them in an undirected
parse tree. Let \( w = w_1 \cdots w_n \) be a sentence
of length \( n \). A parse tree \( t \) belonging to the sentence
\( w \) is an undirected spanning tree of \( n \) vertices, each
representing a word in the sentence \( w \). The syntac-
tic distance between two words \( w_i \) and \( w_j \), denoted
\( \Delta_t(w_i, w_j) \), is defined as the shortest path from
\( w_i \) to \( w_j \) in the tree \( t \) where each edge has weight
1. Note that \( \Delta_t(\cdot, \cdot) \) is a distance in the technical
sense of the word: it is non-negative, symmetric,
and satisfies the triangle inequality.

Tree Extraction Converting from syntactic dis-
tance to a syntactic tree representation (or vice
versa) is trivial and deterministic:

**Proposition 1.** There is a bijection between
syntactic distance and undirected spanning trees.

**Proof.** Suppose we have the syntactic distances
\( \Delta_t(w_i, w_j) \) for an unknown, undirected spanning
tree \( t \). We may uniquely recover that tree by con-
structing a graph with an edge between \( w_i \) and \( w_j \)
iff \( \Delta_t(w_i, w_j) = 1 \). (This analysis also holds if we
have access to only the ordering of the distances
between all \( \{w_i\}^d \) pairs of words, rather than the
perfect distance calculations—if that were the case,
the minimum spanning tree could be computed e.g.
with Prim’s.) On the other hand, if we have an
undirected spanning tree \( t \) and wish to recover the
syntactic distances, we only need to compute the
shortest path between each pair of words, with e.g.
Floyd–Warshall, to yield \( \Delta_t(\cdot, \cdot) \) uniquely.

3 Probe, Meet Parser

In this section, we introduce a popular syntactic
probe and a more traditional parser.

3.1 The Structural Probe

Hewitt and Manning (2019) introduce a novel
method for approximating the syntactic distance
\( \Delta_t(\cdot, \cdot) \) between any two words in a sentence. They
christen their method the structural probe, since it
is intended to uncover latent syntactic structure in
contextual embeddings.\(^2\) To do this, they define a
parameterised distance function whose parameters
are to be learned from data. For a word \( w_i \), let
\( h_i \in \mathbb{R}^d \) denote its contextual embedding, where
\( d \) is the dimensionality of the embeddings from
the model we wish to probe, such as BERT. Hew-
itt and Manning (2019) define the parameterised
distance function

\[
d_B(w_i, w_j) = \sqrt{(h_i - h_j)\top B\top B (h_i - h_j)}
\]

\(^2\)In actual fact, the structural probe consists of two probes,
one used to estimate the syntactic distance between words
(which recovers an undirected graph) and another to calculate
their depth in the tree (which is used to recover ordering). In
this work, we focus exclusively on the former.
where $B \in \mathbb{R}^{r \times d}$ is to be learned from data, and $r$ is a user-defined hyperparameter. The matrix $B^\top B$ is positive semi-definite and has rank at most $r$.\footnote{To see this, let $x \in \mathbb{R}^d$ be a vector. Then, we have that $x^\top B^\top Bx = (Bx)^\top (Bx) = ||Bx||^2 \geq 0$.}

The goal of the loss function, then, is to find $B$ such that the distance function $d_B(\cdot, \cdot)$ best approximates $\Delta(\cdot, \cdot)$. If we are to organise our training data into pairs, each consisting of a gold tree $t$ and the sentence it spans $w$, we can then define the local loss function as

$$\ell(B, \langle t, w \rangle) = \sum_{i=1}^{|w|} \sum_{j=i+1}^{|w|} \left| \Delta_t(w_i, w_j) - d_B(w_i, w_j) \right|$$

which is then averaged over the entire training set $D = \{ \langle t^{(k)}, w^{(k)} \rangle \}_{k=1}^N$ to create the following global loss function

$$\mathcal{L}(B) = \sum_{k=1}^N \frac{1}{|w^{(k)}|^2} \ell \left( B, \langle t^{(k)}, w^{(k)} \rangle \right)$$

Dividing the contribution of each local loss by the square of the length of its sentence (the $|w^{(k)}|^2$ factor in the denominator) ensures that each sentence makes an equal contribution to the overall loss, to avoid a bias towards the effect of longer sentences. This global loss can be minimised computationally using stochastic gradient descent.\footnote{He\textit{witt and Manning} found that replacing $d_B(\cdot, \cdot)$ in eq. (2) with $d_B(\cdot, \cdot)^2$ yielded better empirical results, so we do the same. For a discussion of this, refer to App. A.1 in Hewitt and Manning. Coenen et al. (2019) later offer a theoretical motivation, based on embedding trees in Euclidean space.}

### 3.2 A Structured Perceptron Parser

Given that probe simplicity seemingly refers to parameterisation rather than the design of loss function, we infer that swapping the loss function should not be understood as increasing model complexity. With that in mind, here we describe an alternative to the structural probe which learns parameters for the same function $d_B$—a structured perceptron dependency parser, originally introduced in McDonald et al. (2005).

This parser’s loss function works not by predicting every pairwise distance, but instead by predicting the tree based on the current estimation of the distances between each pair of words, then comparing the total weight of that tree to the total weight of the gold tree (based on the current distance predictions). The local perceptron loss is defined as

$$\ell(B, \langle t, w \rangle) = \sum_{(i,j) \in t} d_B(w_i, w_j) - \min_{t' \in T(w)} \sum_{(i',j') \in t'} d_B(w_{i'}, w_{j'})$$

When the predicted minimum spanning tree $t'$ perfectly matches the gold tree $t$, each edge will cancel and this loss will equal zero. Otherwise, it will be positive, since the sum of the predicted distances for the edges in the gold tree will necessarily exceed the sum in the minimum spanning tree. Local loss is then trivially summed into a global loss as

$$\mathcal{L}(B) = \sum_{k=1}^N \ell \left( B, \langle t^{(k)}, w^{(k)} \rangle \right)$$

This quantity can also be minimised with a (sub)gradient-based method.

Though both the structural probe and the structured perceptron parser may seem equivalent under Prop. 1, there is a subtle but important difference. To minimise the loss in eq. (2), the structural probe needs to encode (in $d_B$) the rank-ordering of the distances between each pair of words within a sentence. This is not necessarily the case for the structured perceptron. It could minimise the loss in eq. (4) by just encoding each pair of words as “near” or “far”—and Prim’s algorithm will do the rest.\footnote{One reviewer argued that, by injecting the tree constraint into the model in this manner, we lose the ability to answer the question of whether a probe discovered trees organically. While we believe this is valid, we do not see why the same criticism cannot be levelled against the structural probe—after all, it is trained on the same trees, just processed into pairwise distance matrices. The trees have been obfuscated, to be sure, but they remain in the data.}

### 4 Experimental Setup

#### 4.1 Processing Results

**Embeddings and Data** We experimented on the contextual embeddings in the final hidden layer of the pre-trained multilingual release of BERT (Devlin et al., 2019), and trained the models on the Universal Dependency (Nivre et al., 2016) treebanks (v2.4). This allowed our analysis to be multilingual. More specifically, we consider eight typologically diverse languages (Arabic, Basque, Czech, Finnish, Japanese, Korean, Tamil, and Turkish), plus English.
Decoding the Predicted Trees Having trained a model to find a $d_B(\cdot, \cdot)$ that approximates $\Delta_t(\cdot, \cdot)$, it is trivial to decode test sentences into trees (see Prop. 1). For an unseen sentence $w = w_1 \cdots w_n$, we compute the $n \times n$ pairwise distance matrix $D$:

$$D_{uv} = \begin{cases} 
  d_B(w_i, w_v) & \text{if } v > u \\
  0 & \text{otherwise}
\end{cases}$$  

We can then compute the predicted tree $t$ from $D$ using Prim’s algorithm, which returns the minimum spanning tree from the predicted distances.

4.2 Experiments

To compare the performance of the models, we used both metrics from Hewitt and Manning (2019), plus a new variant of the second.

UUAS The undirected unlabeled attachment score (UUAS) is a standard metric in the parsing literature, which reports the percentage of correctly identified edges in the predicted tree.

DSpr The second metric is the Spearman rank-order correlation between the predicted distances, which are output from $d_B$, and the gold-standard distances (computable from the gold tree using the Floyd–Warshall algorithm). Hewitt and Manning term this metric distance Spearman (DSpr). While UUAS measures whether the model captures edges in the tree, DSpr considers pairwise distances between all leaves in the tree—even those which are not connected in a single step.

DSpr$_{P+FW}$ As a final experiment, we run DSpr again, but first pass each pairwise distance matrix $D$ through Prim’s (to recover the predicted tree) then through Floyd–Warshall (to recover a new distance matrix, with distances calculated based on the predicted tree). This post-processing would convert a “near”—“far” matrix encoding to a precise rank-order one. This should positively affect the results, in particular for the parser, since that is trained to predict trees which result from the pairwise distance matrix, not the pairwise distance matrix itself.

5 Results

This section presents results for the structural probe and structured perceptron parser.

UUAS Results Figure 2a presents UUAS results for both models. The parser is the highest performing model on seven of the nine languages. In many of these the difference is substantial—in English, for instance, the parser outperforms the structural probe by $11.1$ UUAS points.

DSpr Results The DSpr results (Figure 2b) show the opposite trend: the structural probe outperforms the parser on all languages. The parser performs particularly poorly on Japanese and Arabic, which is surprising, given that these had the second and third largest sets of training data for BERT respectively (refer to Table 1 in the appendices). We speculate that this may be because in the treebanks used, Japanese and Arabic have a longer average sentence length than other languages.

DSpr$_{P+FW}$ Results Following the post-processing step, the difference in DSpr (shown in Figure 3) is far less stark than previously suggested—the mean difference between the two across all nine languages is just 0.0006 (in favour of the parser). Notice in particular the improvement

---

We used the UD treebanks rather than the Penn-Treebank (Marcus et al., 1993), and experimented on the final hidden layer of multilingual BERT using a subset of 12,000 sentences from the larger treebanks. This renders our numbers incomparable to those found in Hewitt and Manning (2019).
for both Arabic and Japanese—where previously (in the vanilla DSpr) the structured perceptron vastly underperformed, the post-processing step closes the gap almost entirely. Though Prop. 1 implies that we do not need to consider the full pairwise output of $d_B$ to account for global properties of tree, this is not totally borne out in our empirical findings, since we do not see the same trend in DSpr$_{P+FW}$ as we do in UUAS. If we recover the gold tree, we will have a perfect correlation with the true syntactic distance—but we do not always recover the gold tree (the UUAS is less than 100%), and therefore the errors the parser makes are pronounced.

6 Discussion: Probe v. Parser

Although we agree that probes should be somehow more constrained in their complexity than models designed to perform well on tasks, we see no reason why being a “probe” should necessitate fundamentally different design choices. It seems clear from our results that how you design a probe has a notable effect on the conclusions one might draw about a representation. Our parser was trained to recover trees (so it is more attuned to UUAS), whilst the structural probe was trained to recover pairwise distances (so it is more attuned to DSpr)—viewed this way, our results are not surprising in the least.

The fundamental question for probe designers, then, is which metric best captures a linguistic structure believed to be a property of a given representation—in this case, syntactic dependency. We suggest that probing research should focus more explicitly on this question—on the development and justification of probing metrics. Once a metric is established and well motivated, a lightweight probe can be developed to determine whether that structure is present in a model.

If proposing a new metric, however, the burden of proof lies with the researcher to articulate and demonstrate why it is worthwhile. Moreover, this process of exploring which details a new metric is sensitive to (and comparing with existing metrics) ought not be conflated with an analysis of a particular model (e.g. BERT)—it should be clear whether the results enable us to draw conclusions about a model, or about a means of analysing one.

For syntactic probing, there is certainly no a priori reason why one should prefer DSpr to UUAS. If anything, we tentatively recommend UUAS, pending further investigation. The DSpr$_{P+FW}$ results show no clear difference between the models, whereas UUAS exhibits a clear trend in favour of the parser, suggesting that it may be easier to recover pairwise distances from a good estimate of the tree than vice versa. UUAS also has the advantage that it is well described in the literature (and, in turn, well understood by the research community).

According to UUAS, existing methods were able to identify more syntax in BERT than the structural probe. In this context, though, we use these results not to give kudos to BERT, but to argue that the perceptron-based parser is a better tool for syntactic probing. Excluding differences in parameterisation, the line between what constitutes a probe or a model designed for a particular task is awfully thin, and when it comes to syntactic probing, a powerful probe seems to look a lot like a traditional parser.

7 Conclusion

We advocate for the position that, beyond some notion of model complexity, there should be no inherent difference between the design of a probe and a model designed for a corresponding task. We analysed the structural probe (Hewitt and Manning, 2019), and showed that a simple parser with an identical lightweight parameterisation was able to identify more syntax in BERT in seven of nine compared languages under UUAS. However, the structural probe outperformed the parser on a novel metric proposed in Hewitt and Manning (2019), bringing to attention a broader question: how should one choose metrics for probing? In our discussion, we argued that if one is to propose a new metric, they should clearly justify its usage.

Acknowledgements

We thank John Hewitt for engaging wholeheartedly with our work and sharing many helpful insights.
References


A Training Details

For all models (separately for each language), we considered three hyperparameters: $r$ (full rank when $r = 768$, since this is the dimensionality of the BERT embeddings), the learning rate, and the dropout (Srivastava et al., 2014). To optimise these, we performed a random search, selecting values as judged by loss on the development set. When training, we used a batch size of 64 sentences, and employed early stopping after five steps based on loss reduction. As the optimiser, we used Adam (Kingma and Ba, 2015).

For each language, we used the largest available Universal Dependency 2.4 treebank. One-word sentences and sentences of over 50 words were discarded, and the larger treebanks were pruned to 12,000 sentences (in an 8:1:1 data split).

We used the BERT implementation made by Wolf et al. (2019). Since BERT accepts WordPiece units (Wu et al., 2016) rather than words, where necessary we averaged the output to get word-level embeddings. This is clearly a naïve composition method; improving it would likely strengthen the results for both the probe and the parser.

B Multilingual BERT Details

The multilingual BERT has 12 layers, 768 hidden states, and a total of 110M parameters. It was trained on the complete Wikipedia dumps for the 104 languages with the largest Wikipedias. Table 1 reports the size of the Wikipedias for the languages considered in this paper. Further details of the training can be found on Google Research’s GitHub.

<table>
<thead>
<tr>
<th>Language</th>
<th>Articles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>1,016,152</td>
</tr>
<tr>
<td>Busque</td>
<td>342,426</td>
</tr>
<tr>
<td>Czech</td>
<td>439,467</td>
</tr>
<tr>
<td>English</td>
<td>5,986,229</td>
</tr>
<tr>
<td>Finnish</td>
<td>473,729</td>
</tr>
<tr>
<td>Japanese</td>
<td>1,178,594</td>
</tr>
<tr>
<td>Korean</td>
<td>476,068</td>
</tr>
<tr>
<td>Tamil</td>
<td>125,031</td>
</tr>
<tr>
<td>Turkish</td>
<td>336,380</td>
</tr>
</tbody>
</table>

Table 1: The number of articles in the Wikipedias of the languages considered.

---


https://github.com/google-research/bert/blob/master/multilingual.md