A Structural Probe for Finding Syntax in Word Representations

John Hewitt
Stanford University
johnhew@stanford.edu

Christopher D. Manning
Stanford University
manning@stanford.edu

Abstract

Recent work has improved our ability to detect linguistic knowledge in word representations. However, current methods for detecting syntactic knowledge do not test whether syntax trees are represented in their entirety. In this work, we propose a *structural probe*, which evaluates whether syntax trees are embedded in a linear transformation of a neural network's word representation space. The probe identifies a linear transformation under which squared L2 distance encodes the distance between words in the parse tree, and one in which squared L2 norm encodes depth in the parse tree. Using our probe, we show that such transformations exist for both ELMo and BERT but not in baselines, providing evidence that entire syntax trees are embedded implicitly in deep models' vector geometry.

1 Introduction

As pretrained deep models that build contextualized representations of language continue to provide gains on NLP benchmarks, understanding what they learn is increasingly important. To this end, probing methods are designed to evaluate the extent to which representations of language encode particular knowledge of interest, like part-of-speech (Belinkov et al., 2017), morphology (Peters et al., 2018a), or sentence length (Adi et al., 2017). Such methods work by specifying a *probe* (Conneau et al., 2018; Hupkes et al., 2018), a supervised model for finding information in a representation.

什么是探针

Of particular interest, both for linguistics and for building better models, is whether deep models' representations encode syntax (Linzen, 2018). Despite recent work (Kuncoro et al., 2018; Peters et al., 2018b; Tenney et al., 2019), open questions remain as to whether deep contextual models encode entire parse trees in their word representations.

In this work, we propose a *structural probe*, a simple model which tests whether syntax trees are consistently embedded in a linear transformation of a neural network's word representation space. Tree structure is embedded if the transformed space has the property that squared L2 distance between two words' vectors corresponds to the number of edges between the words in the parse tree. To reconstruct edge directions, we hypothesize a linear transformation under which the squared L2 norm corresponds to the depth of the word in the parse tree. Our probe uses supervision to find the transformations under which these properties are best approximated for each model. If such transformations exist, they define inner products on the original space under which squared distances and norms encode syntax trees – even though the models being probed were never given trees as input or supervised to reconstruct them. This is a structural property of the word representation space, akin to vector offsets encoding word analogies (Mikolov et al., 2013). Using our probe, we conduct a targeted case study, showing that ELMo (Peters et al., 2018a) and BERT (Devlin et al., 2019) representations embed parse trees with high consistency in contrast to baselines, and in a low-rank space.[1]

句法结构的具体含义

In summary, we contribute a simple structural probe for finding syntax in word representations (§2), and experiments providing insights into and examples of how a low-rank transformation recovers parse trees from ELMo and BERT representations (§3,4). Finally, we discuss our probe and limitations in the context of recent work (§5).

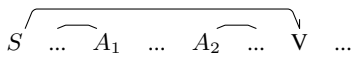
2 Methods

Our goal is to design a simple method for testing whether a neural network embeds each sentence's

[1]We release our code at https://github.com/john-hewitt/structural-probes.

Proceedings of NAACL-HLT 2019, pages 4129–4138
Minneapolis, Minnesota, June 2 - June 7, 2019.

dependency parse tree in its contextual word representations – a structural hypothesis. Under a reasonable definition, to embed a graph is to learn a vector representation of each node such that geometry in the vector space—distances and norms—approximates geometry in the graph (Hamilton et al., 2017). Intuitively, why do parse tree distances and depths matter to syntax? The distance metric—the path length between each pair of words—recovers the tree T simply by identifying that nodes u, v with distance $d_T(u, v) = 1$ are neighbors. The node with greater norm—depth in the tree—is the child. Beyond this identity, the distance metric explains hierarchical behavior. For example, the ability to perform the classic hierarchy test of subject-verb number agreeement (Linzen et al., 2016) in the presence of "attractors" can be explained as the verb (V) being closer in the tree to its subject (S) than to any of the attactor nouns:



Intuitively, if a neural network embeds parse trees, it likely will not use its entire representation space to do so, since it needs to encode many kinds of information. Our probe learns a linear transformation of a word representation space such that the transformed space embeds parse trees across all sentences. This can be interpreted as finding the part of the representation space that is used to encode syntax; equivalently, it is finding the distance on the original space that best fits the tree metrics.

2.1 The structural probe

In this section we provide a description of our proposed structural probe, first discussing the distance formulation. Let $\mathcal{M}$ be a model that takes in a sequence of n words $w^{\ell}_{1:n}$ and produces a sequence of vector representations $\mathbf{h}^{\ell}_{1:n}$, where ℓ identifies the sentence. Starting with the dot product, recall that we can define a family of inner products, $\mathbf{h}^T A \mathbf{h}$, parameterized by any positive semi-definite, symmetric matrix $A \in \mathbb{S}^{m \times m}_{+}$. Equivalently, we can view this as specifying a linear transformation $B \in \mathbb{R}^{k \times m}$, such that $A = B^T B$. The inner product is then $(B\mathbf{h})^T(B\mathbf{h})$, the norm of $\mathbf{h}$ once transformed by B. Every inner product corresponds to a distance metric. Thus, our family of squared distances is defined as:

$$d_B(\mathbf{h}^{\ell}_i, \mathbf{h}^{\ell}_j)^2 = \left(B(\mathbf{h}^{\ell}_i - \mathbf{h}^{\ell}_j)\right)^T \left(B(\mathbf{h}^{\ell}_i - \mathbf{h}^{\ell}_j)\right) \quad (1)$$

where i, j index the word in the sentence.[2] The parameters of our probe are exactly the matrix B, which we train to recreate the tree distance between all pairs of words (w^{ℓ}_i, w^{ℓ}_j) in all sentences T^{ℓ} in the training set of a parsed corpus. Specifically, we approximate through gradient descent:

$$\min_B \sum_{\ell} \frac{1}{|s^{\ell}|^2} \sum_{i,j} \left| d_{T^{\ell}}(w^{\ell}_i, w^{\ell}_j) - d_B(\mathbf{h}^{\ell}_i, \mathbf{h}^{\ell}_j)^2 \right|$$

where $|s^{\ell}|$ is the length of the sentence; we normalize by the square since each sentence has $|s^{\ell}|^2$ word pairs.

2.2 Properties of the structural probe

Because our structural probe defines a valid distance metric, we get a few nice properties for free. The simplest is that distances are guaranteed non-negative and symmetric, which fits our probing task. Perhaps most importantly, the probe tests the concrete claim that there exists an inner product on the representation space whose squared distance—a global property of the space—encodes syntax tree distance. This means that the model not only encodes which word is governed by which other word, but each word's proximity to every other word in the syntax tree.[3] This is a claim about the structure of the representation space, akin to the claim that analogies are encoded as vector-offsets in uncontextualized word embeddings (Mikolov et al., 2013). One benefit of this is the ability to query the nature of this structure: for example, the dimensionality of the transformed space (§ 4.1).

2.3 Tree depth structural probes

The second tree property we consider is the *parse depth* $\|w_i\|$ of a word w_i, defined as the number of edges in the parse tree between w_i and the root of the tree. This property is naturally represented as a norm – it imposes a total order on the words in the sentence. We wish to probe to see if there exists a squared norm on the word representation

[2] As noted in Eqn 1, in practice, we find that approximating the parse tree distance and norms with the *squared* vector distances and norms consistently performs better. Because a distance metric and its square encode exactly the same parse trees, we use the squared distance throughout this paper. Also strictly, since A is not positive definite, the inner product is indefinite, and the distance a pseudometric. Further discussion can be found in our appendix.

[3] Probing for distance instead of headedness also helps avoid somewhat arbitrary decisions regarding PP headedness, the DP hypothesis, and auxiliaries, letting the representation "disagree" on these while still encoding roughly the same global structure. See Section 5 for more discussion.

Method	Distance UUAS	Distance DSpr.	Depth Root%	Depth NSpr.
LINEAR	48.9	0.58	2.9	0.27
ELMO0	26.8	0.44	54.3	0.56
DECAY0	51.7	0.61	54.3	0.56
PROJ0	59.8	0.73	64.4	0.75
ELMO1	77.0	0.83	86.5	0.87
BERTBASE7	79.8	0.85	88.0	0.87
BERTLARGE15	**82.5**	0.86	89.4	0.88
BERTLARGE16	81.7	**0.87**	**90.1**	**0.89**

Table 1: Results of structural probes on the PTB WSJ test set; baselines in the top half, models hypothesized to encode syntax in the bottom half. For the distance probes, we show the Undirected Unlabeled Attachment Score (UUAS) as well as the average Spearman correlation of true to predicted distances, DSpr. For the norm probes, we show the root prediction accuracy and the average Spearman correlation of true to predicted norms, NSpr.

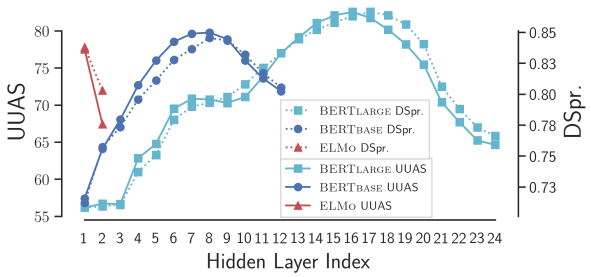


Figure 1: Parse distance UUAS and distance Spearman correlation across the BERT and ELMo model layers.

space that encodes this tree norm. We replace the vector distance function $d_B(\mathbf{h}_i, \mathbf{h}_j)$ with the squared vector norm $\|\mathbf{h}_i\|_B^2$, replacing Equation 1 with $\|\mathbf{h}_i\|_A = (B\mathbf{h}_i)^T(B\mathbf{h}_i)$ and training B to recreate $\|w_i\|$. Like the distance probe, this norm formulation makes a concrete claim about the structure of the vector space.

不算距离，直接平方

3 Experiments

Using our probe, we evaluate whether representations from ELMo and BERT, two popular English models pre-trained on language modeling-like objectives, embed parse trees according to our structural hypothesis. Unless otherwise specified, we permit the linear transformation B to be potentially full-rank (i.e., B is square.) Later, we explore what rank of transformation is actually necessary for encoding syntax (§ 4.1).

Representation models We use the 5.5B-word pre-trained ELMo weights for all ELMo representations, and both BERT-base (cased) and BERT-large (cased). The representations we evaluate are denoted ELMOK, BERTBASEK, BERTLARGEK, where K indexes the hidden layer of the corresponding model. All ELMo and BERT-large layers are dimensionality 1024; BERT-base layers are dimensionality 768.

Data We probe models for their ability to capture the Stanford Dependencies formalism (de Marneffe et al., 2006), claiming that capturing most aspects of the formalism implies an understanding of English syntactic structure. To this end, we obtain fixed word representations for sentences of the parsing train/dev/test splits of the Penn Treebank (Marcus et al., 1993), with no pre-processing.[4]

Baselines Our baselines should encode features useful for training a parser, but not be capable of parsing themselves, to provide points of comparison against ELMo and BERT. They are as follows:

LINEAR : The tree resulting from the assumption that English parse trees form a left-to-right chain. A model that encodes the positions of words should be able to meet this baseline.

ELMO0 : Strong character-level word embeddings with no contextual information. As these representations lack even position information, we should be completely unable to find syntax trees embedded.

DECAY0 : Assigns each word a weighted average of all ELMO0 embeddings in the sentence. The weight assigned to each word decays exponentially as $\frac{1}{2^d}$, where d is the linear distance between the words.

PROJ0 : Contextualizes the ELMO0 embeddings with a randomly initialized BiLSTM layer of dimensionality identical to ELMo (1024), a surprisingly strong baseline for contextualization (Conneau et al., 2018).

3.1 Tree distance evaluation metrics

We evaluate models on how well the predicted distances between all pairs of words reconstruct gold parse trees and correlate with the parse trees' distance metrics. To evaluate tree reconstruction, we take each test sentence's predicted parse tree distances and compute the minimum spanning tree. We evaluate the predicted tree on undirected

[4]Since BERT constructs subword representations, we align subword vectors with gold Penn Treebank tokens, and assign each token the average of its subword representation. This thus represents a lower-bound on BERT's performance.

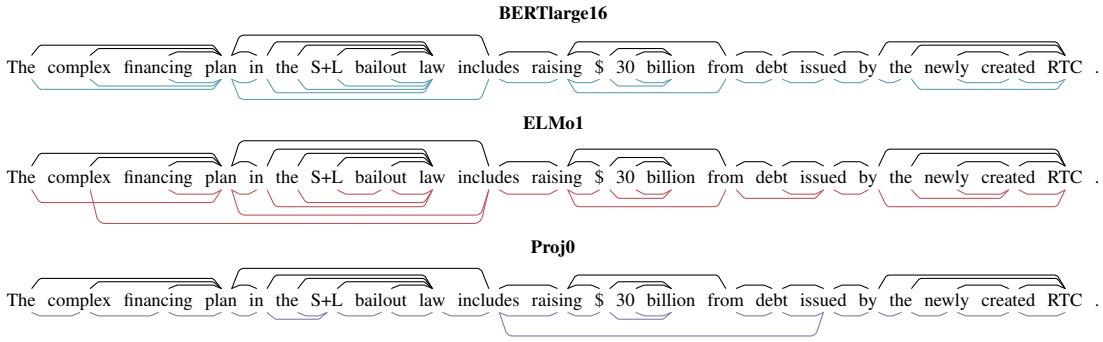


Figure 2: Minimum spanning trees resultant from predicted squared distances on BERTLARGE16 and ELMO1 compared to the best baseline, PROJ0. Black edges are the gold parse, above each sentence; blue are BERTLARGE16, red are ELMO1, and purple are PROJ0.

attachment score (UUAS)—the percent of undirected edges placed correctly—against the gold tree. For distance correlation, we compute the Spearman correlation between true and predicted distances for each word in each sentence. We average these correlations between all sentences of a fixed length, and report the macro average across sentence lengths 5–50 as the "distance Spearman (DSpr.)" metric.[5]

3.2 Tree depth evaluation metrics

We evaluate models on their ability to recreate the order of words specified by their depth in the parse tree. We report the Spearman correlation betwen the true depth ordering and the predicted ordering, averaging first between sentences of the same length, and then across sentence lengths 5–50, as the "norm Spearman (NSpr.)". We also evaluate models' ability to identify the root of the sentence as the least deep, as the "root%".[6]

4 Results

We report the results of parse distance probes and parse depth probes in Table 1. We first confirm that our probe can't simply "learn to parse" on top of any informative representation, unlike parser-based probes (Peters et al., 2018b). In particular, ELMO0 and DECAY0 fail to substantially outperform a right-branching-tree oracle that encodes the linear sequence of words. PROJ0, which has all of the representational capacity of ELMO1 but none of the training, performs the best among the baselines. Upon inspection, we found that our probe on PROJ0 improves over the linear hypothesis with

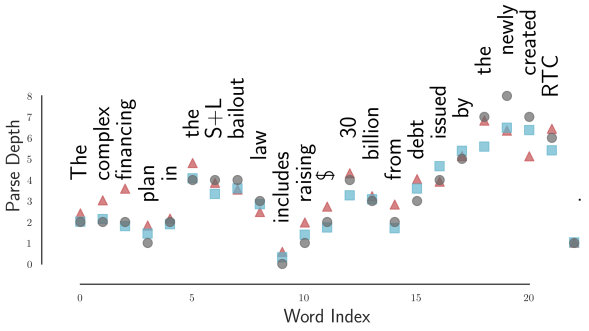


Figure 3: Parse tree depth according to the gold tree (black, circle) and the norm probes (squared) on ELMO1 (red, triangle) and BERTLARGE16 (blue, square).

mostly simple deviations from linearity, as visualized in Figure 2.

We find surprisingly robust syntax embedded in each of ELMo and BERT according to our probes. Figure 2 shows the surprising extent to which a minimum spanning tree on predicted distances recovers the dependency parse structure in both ELMo and BERT. As we note however, the distance metric itself is a global notion; all pairs of words are trained to know their distance – not just which word is their head; Figure 4 demonstrates the rich structure of the true parse distance metric recovered by the predicted distances. Figure 3 demonstrates the surprising extent to which the depth in the tree is encoded by vector norm after the probe transformation. Between models, we find consistently that BERTLARGE performs better than BERTBASE, which performs better than ELMO.[7] We also find, as in Peters et al. (2018b), a clear difference in syntactic information between layers; Figure 1 reports the performance

[5]The 5–50 range is chosen to avoid simple short sentences as well as sentences so long as to be rare in the test data.

[6]In UUAS and "root%" evaluations, we ignore all punctuation tokens, as is standard.

[7]It is worthwhile to note that our hypotheses were developed while analyzing LSTM models like ELMo, and applied without modification on the self-attention based BERT models.

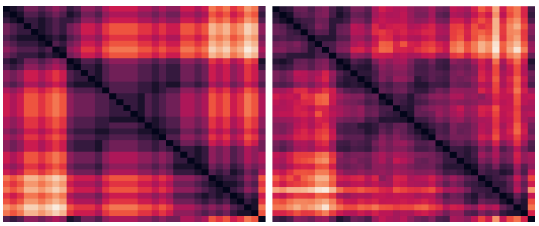


Figure 4: (left) Matrix representing gold tree distances between all pairs of words in a sentence, whose linear order runs top-to-bottom and left-to-right. Darker colors indicate close words, lighter indicate far. (right) The same distances as embedded by BERTLARGE16 (squared). More detailed graphs available in the Appendix.

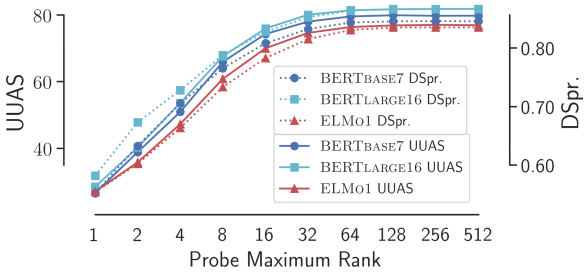


Figure 5: Parse distance tree reconstruction accuracy when the linear transformation is constrained to varying maximum dimensionality.

of probes trained on each layer of each system.

4.1 Analysis of linear transformation rank

With the result that there exists syntax-encoding vector structure in both ELMo and BERT, it is natural to ask how compactly syntactic information is encoded in the vector space. We find that in both models, the effective rank of linear transformation required is surprisingly low. We train structural probes of varying k, that is, specifying a matrix $B \in \mathbb{R}^{k \times m}$ such that the transformed vector $B\mathbf{h}$ is in $\mathbb{R}^k$. As shown in Figure 5, increasing k beyond 64 or 128 leads to no further gains in parsing accuracy. Intuitively, larger k means a more expressive probing model, and a larger fraction of the representational capacity of the model being devoted to syntax. We also note with curiosity that the three models we consider all seem to require transformations of approximately the same rank; we leave exploration of this to exciting future work.

几个表示的最佳秩一致

5 Discussion & Conclusion

Recent work has analyzed model behavior to determine if a model understands hierarchy and other linguistic phenomena (Linzen, 2018; Gulordava et al., 2018; Kuncoro et al., 2018; Linzen and Leonard, 2018; van Schijndel and Linzen, 2018; Tang et al., 2018; Futrell et al., 2018). Our work extends the literature on linguistic probes, found at least in (Peters et al., 2018b; Belinkov et al., 2017; Blevins et al., 2018; Hupkes et al., 2018). Conneau et al. (2018) present a task similar to our parse depth prediction, where a sentence representation vector is asked to classify the maximum parse depth ever achieved in the sentence. Tenney et al. (2019) evaluates a complementary task to ours, training probes to learn the *labels* on structures when the gold structures themselves are given. Peters et al. (2018b) evaluates the extent to which constituency trees can be extracted from hidden states, but uses a probe of considerable complexity, making less concrete hypotheses about how the information is encoded.

Probing tasks and limitations Our reviewers rightfully noted that one might just probe for headedness, as in a bilinear graph-based dependency parser. More broadly, a deep neural network probe of some kind is almost certain to achieve higher parsing accuracies than our method. Our task and probe construction are designed not to test for some notion of syntactic knowledge broadly construed, but instead for an extremely strict notion where all pairs of words know their syntactic distance, and this information is a global structural property of the vector space. However, this study is limited to testing that hypothesis, and we foresee future probing tasks which make other tradeoffs between probe complexity, probe task, and hypotheses tested.

In summary, through our structural probes we demonstrate that the structure of syntax trees emerges through properly defined distances and norms on two deep models' word representation spaces. Beyond this actionable insight, we suggest our probe may be useful for testing the existence of different types of graph structures on any neural representation of language, an exciting avenue for future work.

6 Acknowledgements

We would like to acknowledge Urvashi Khandelwal and Tatsunori B. Hashimoto for formative advice in early stages, Abigail See, Kevin Clark, Siva Reddy, Drew A. Hudson, and Roma Patel for helpful comments on drafts, and Percy Liang, for guidance on rank experiments. We would also like to thank the reviewers, whose helpful comments led to increased clarity and extra experiments. This research was supported by a gift from Tencent.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *International Conference on Learning Representations*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872. Association for Computational Linguistics.

Terra Blevins, Omer Levy, and Luke Zettlemoyer. 2018. Deep RNNs encode soft hierarchical syntax. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 14–19. Association for Computational Linguistics.

Alexis Conneau, Germán 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. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics.

Richard Futrell, Ethan Wilcox, Takashi Morita, and Roger Levy. 2018. RNNs as psycholinguistic subjects: Syntactic state and grammatical dependency. *arXiv preprint arXiv:1809.01329*.

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

William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*.

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.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1426–1436.

Tal Linzen. 2018. What can linguistics and deep learning contribute to each other? *arXiv preprint arXiv:1809.04179*.

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

Tal Linzen and Brian Leonard. 2018. Distinct patterns of syntactic agreement errors in recurrent networks and humans. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society*, pages 692–697. Cognitive Science Society, Austin, TX.

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

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. Dynet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*.

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

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018b. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509.

Marten van Schijndel and Tal Linzen. 2018. Modeling garden path effects without explicit hierarchical syntax. In Tim Rogers, Marina Rau, Jerry Zhu, and Chuck Kalish, editors, *Proceedings of the 40th Annual Conference of the Cognitive Science Society*, pages 2600–2605. Cognitive Science Society, Austin, TX.

Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. 2018. Why self-attention? A targeted evaluation of neural machine translation architectures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4263–4272. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

A Appendix: Implementation Details

A.1 Squared L2 distance vs. L2 distance

In Section 2.2, we note that while our distance probe specifies a distance metric, we recreate it with a squared vector distance; likewise, while our norm probe specifies a norm, we recreate it with a squared vector norm. We found this to be important for recreating the exact parse tree distances and norms. This does mean that in order to recreate the exact scalar values of the parse tree structures, we need to use the squared vector quantities. This may be problematic, since for example squared distance doesn't obey the triangle inequality, whereas a valid distance metric does.

However, we note that in terms of the graph structures encoded, distance and squared distance are identical. After training with the squared vector distance, we can square-root the predicted quantities to achieve a distance metric. The relative ordering between all pairs of words will be unchanged; the same tree is encoded either way, and none of our quantitative metrics will change; however, the exact scalar distances will differ from the true tree distances.

This raises a question for future work as to why squared distance works better than distance, and beyond that, what function of the L2 distance (or perhaps, what L^p distance) would best encode tree distances. It is possibly related to the gradients of the loss with respect to the function of the distance, as well as how amenable the function is to matching the exact scalar values of the tree distances.

A.2 Probe training details

All probes are trained to minimize L1 loss of the predicted squared distance or squared norm w.r.t. the true distance or norm. Optimization is performed using the Adam optimizer (Kingma and Ba, 2014) initialized at learning rate 0.001, with $\beta_1 = .9, \beta_2 = .999, \epsilon = 10^{-8}$. Probes are trained to convergence, up to 40 epochs, with a batch size of 20. For depth probes, loss is summed over all predictions in a sentence, normalized by the length of the sentence, and then summed over all sentences in a batch before a gradient step is taken. For distance probes, normalization is performed by the square of the length of the sentence. At each epoch, dev loss is computed; if the dev loss does not achieve a new minimum, the optimizer is reset (no momentum terms are kept) with an initial learning rate multiplied by 0.1. All models were implemented in both DyNet (Neubig et al., 2017), and in PyTorch (Paszke et al., 2017).

B Appendix: Extra examples

In this section we provide additional examples of model behavior, including baseline model behavior, across parse distance prediction and parse depth prediction. In Figure 6 and Figure 7, we present a single sentence with dependency trees as extracted from many of our models and baselines. In Figure 8, we present tree depth predictions on a complex sentence from ELMO1, BERTLARGE16, and our baseline PROJ0. Finally, in Figure 9, we present gold parse distances and predicted squared parse distances between all pairs of words in large, high-resolution format.

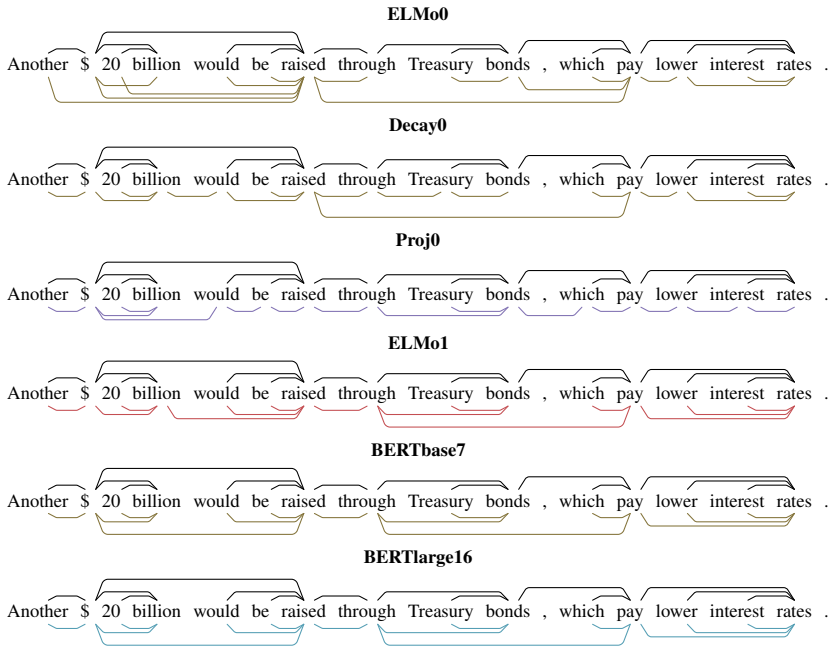


Figure 6: A relatively simple sentence, and the minimum spanning trees extracted by various models.

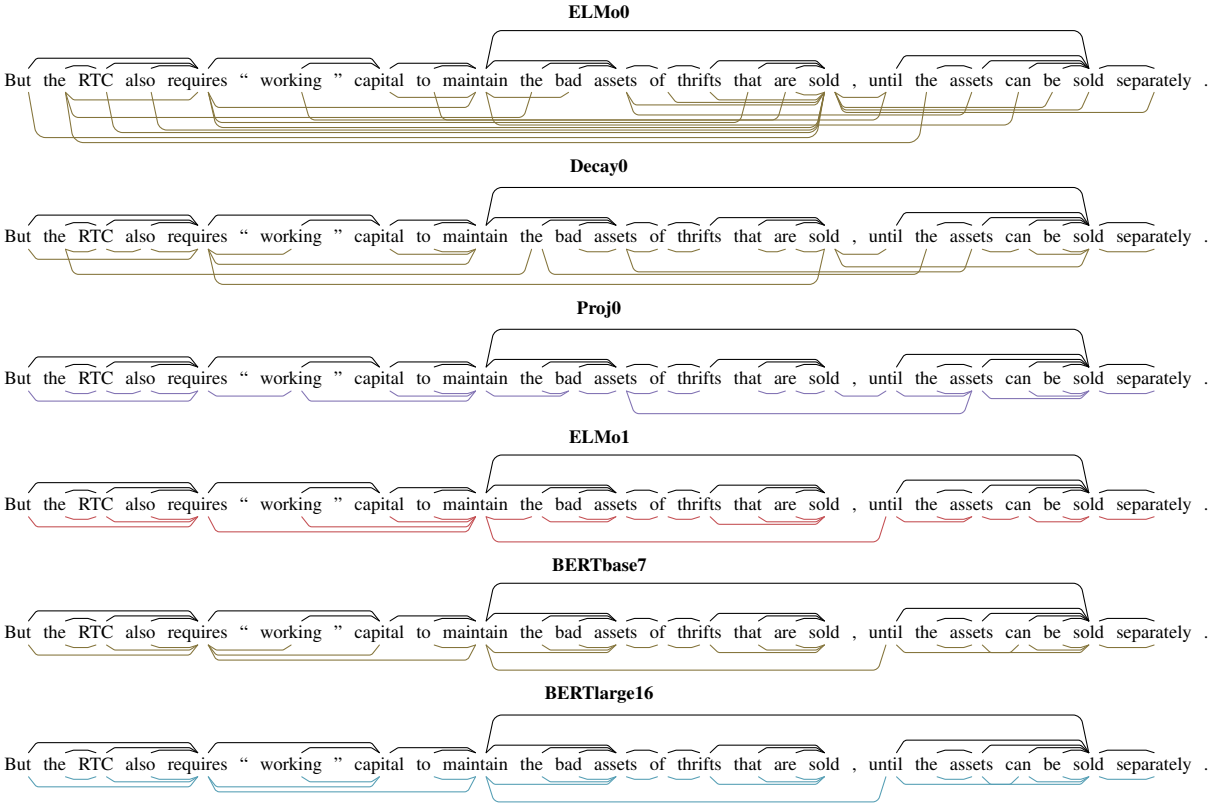


Figure 7: A complex sentence, and the minimum spanning trees extracted by various models.

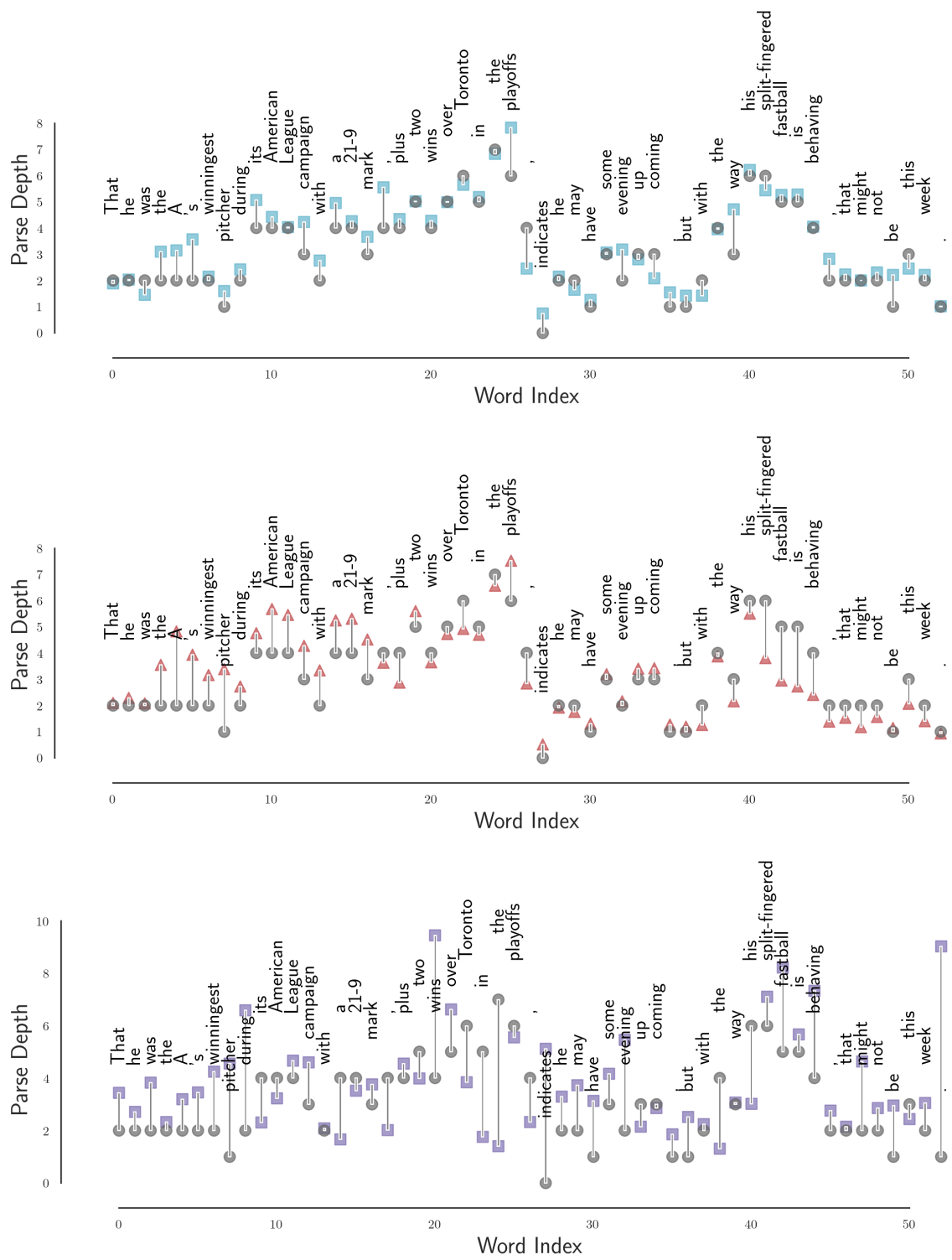


Figure 8: A long sentence with gold dependency parse depths (grey) and dependency parse depths (squared) as extracted by BERTLARGE16 (blue, top), ELMO1 (red, middle), and the baseline PROJ0 (purple, bottom). Note the non-standard subject, "that he was the A's winningest pitcher".

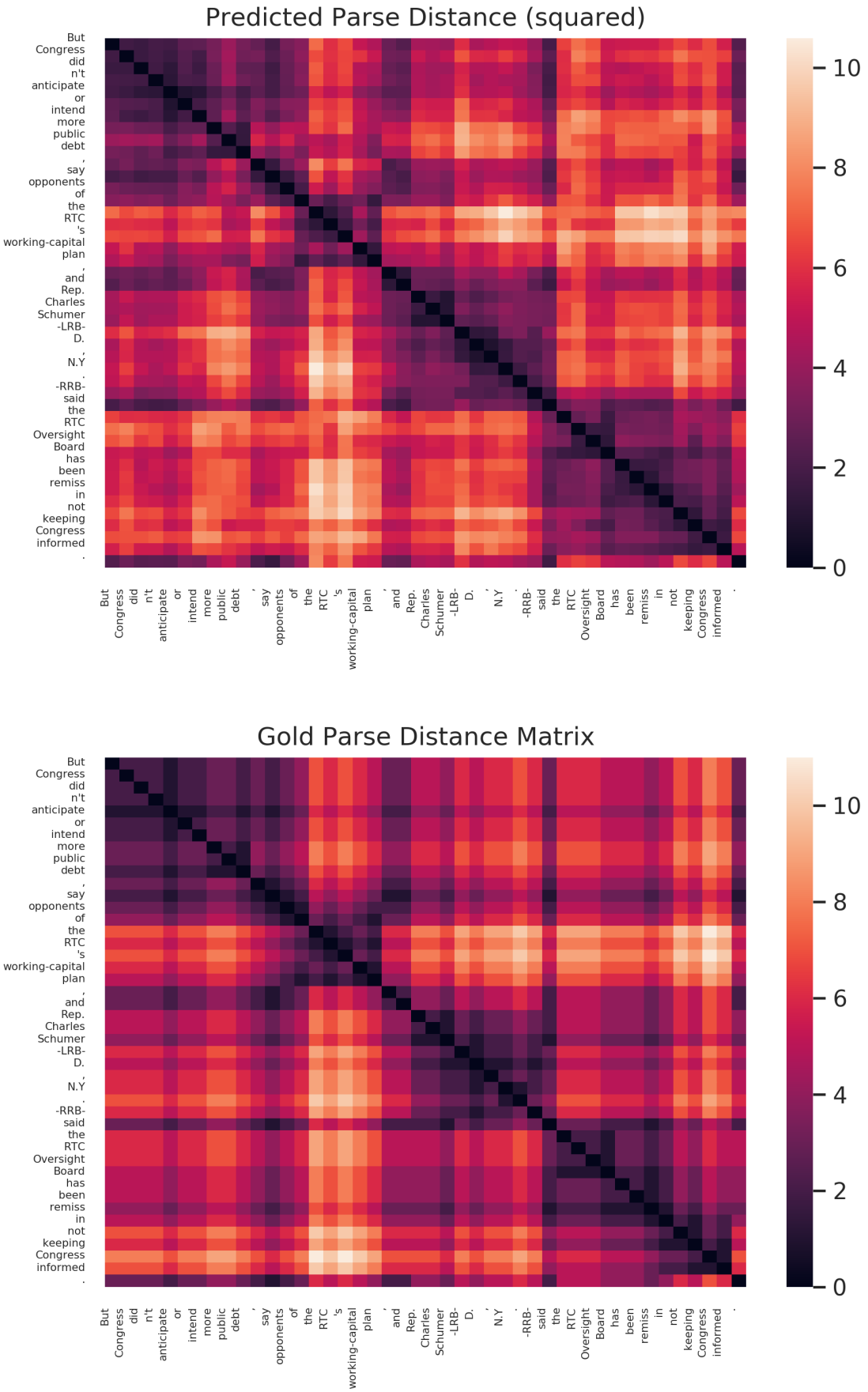


Figure 9: The distance graphs defined by the gold parse distances on a sentence (below) and as extracted from BERTLARGE16 (above, squared).