Probing Language Models

Omar Elbaghdadi
12660256
omarelb@gmail.com

Aman Hussain
12667447
aman.hussain@student.uva.nl

1 Introduction

While the field of NLP used to be dominated by recurrent models such as the LSTM (Hochreiter and Schmidhuber, 1997), current state-of-the-art models such as BERT (Devlin et al., 2018) use Transformer architectures (Vaswani et al., 2017) as the basic building block. Transformers do away with recurrence and are based solely on attention mechanisms (Bahdanau et al., 2016) instead. Why attention-based mechanisms perform better than recurrent models is still an open question. Recurrent models don't lend themselves well to parallellization, which constrains them to smaller datasets. It has not yet been tested whether a recurrent model trained on the same amount of data as an attention-based model will yield similar performance gains. While Vaswani et al. (2017) show that Transformers are better able to deal with long-range dependencies between words than recurrent models, humans do process language in an incremental and recurrent fashion.

To better understand the difference between these two paradigms, we need to take a look at how they represent language. It has been shown that natural language processing systems benefit immensely from learning good representations of language (Mikolov et al., 2013; Pennington et al., 2014). While deep models that compute contextualized word representations (Peters et al., 2018; Devlin et al., 2018) have reached state-of-the-art performance, what these representations encode largely remains a mystery. To this end, *probes* have been designed to find out whether learned representations encode linguistic information (Conneau et al., 2018; Hupkes et al., 2018).

To probe sub-sentence level linguistic phenomena such as part-of-speech (POS) tags and coreference, Tenney et al. (2019) introduce edge probing. Hewitt and Manning (2019) go a step further by introducing structural probes, which test whether *entire syntax trees* are embedded in deep representations' vector geometry.

However, probes do not always faithfully show to what degree a representation encodes some linguistic feature. Hewitt and Liang (2019) introduce *control tasks* to differentiate the encoding of linguistic information from the probe's ability to learn *any* supervised task. Using control tasks, we can define a better metric for how much information a representation encodes than predictive accuracy: *selectivity*. Models with higher selectivity better encode the information we're probing for.

Our contribution is to investigate to what extent, and in what way, the representations of recurrent models, in particular Gulordava et al. (2018)'s language model, and attention-based models, in particular a distilled verision of GPT-2 (Radford et al., 2019), encode syntax, using POS tagging probes and structural probes. Here we list our major findings:

- Representations learnt by our recurrent model have better selectivity for POS tag classification task than our attention-based model. This implies that LSTM representations better encode POS tag information than attention-based ones.

- For POS tagging, selectivity increases in deeper layers for both the recurrent and attention-based models. To understand this phenomenon, we propose an information theory based explanation.

- Recurrent models also seem to encode syntax tree information to a higher degree than attention-based models.

2 Methods

2.1 Probes and Control Tasks

To probe a representation for a feature, we generally ask the following question: how predictive is the representation for that feature? The better we are able to predict that feature, the more that feature should be encoded in the representation. To answer that question, we specify a probe or *diagnostic classifier* (Hupkes et al., 2018): a very simple classifier f that predicts a feature $t := f(h)$ from a given word representation h. In the context of POS tagging, t would be a POS tag. We want the classifier to be as simple as possible, to avoid it learning more than the representation encodes.

However, it is still possible for the probe to learn the task from the task supervision itself, without necessarily using that specific feature's information. Control tasks (Hewitt and Liang, 2019) control for this possibility by introducing a baseline: predicting random labels. Control tasks define a control behavior $C(v)$ that assigns a random label to a word type v. Words of the same type do get assigned the same label.

The performance of the probe on the random task indicates its ability to learn any task through supervision. Our new measure for how much a representation encodes, *selectivity*, is then defined as the difference in performance on the real task and the random task. A larger selectivity indicates that the representation encodes the feature to a larger degree.

2.2 Structural Probes

Compared to predicting POS tags, representing entire syntax trees should intuitively require richer syntactic representations, as they include direction and hierarchy. Structural probes (Hewitt and Manning, 2019) test to what degree a network's representation space can represent syntax trees. They do this by trying to find a distance measure on the original space that best corresponds to distance in tree space: the number of edges between tree nodes. The existence of such transformations verifies the presence or encoding of syntax tree structure in the original representation space.

Let $\boldsymbol{h}_i^\ell \in \mathbb{R}^d$ be a hidden representation generated by a model for w_i, word i, in sentence ℓ. Any distance metric in vector space can be parameterized by first projecting h to another space with a rank k linear transformation $B \in \mathbb{R}^{k \times d}$, and taking the inner product in that space. We can then write the (squared) distance between two hidden representations $\boldsymbol{h}_i^\ell$ and $\boldsymbol{h}_j^\ell$ under the inner product induced by B as:

$$d_B(\boldsymbol{h}_i^\ell, \boldsymbol{h}_j^\ell)^2 = (B(\boldsymbol{h}_i^\ell - \boldsymbol{h}_j^\ell))^{\mathrm{T}}(B(\boldsymbol{h}_i^\ell - \boldsymbol{h}_j^\ell)). \quad (1)$$

Through gradient descent, we learn the transformation matrix B so that distances between representations approximate distances between tree nodes:

$$\min_B \sum_\ell \frac{1}{|s^\ell|^2} \sum_{i,j} |d_{T^\ell}(w_i^\ell, w_j^\ell) - d_B(\boldsymbol{h}_i^\ell, \boldsymbol{h}_j^\ell)^2|, \quad (2)$$

where $|s^l|$ is the length of the sentence, $d_{T^\ell}(w_i^\ell, w_j^\ell)$ indicates the number of edges between words w_i^ℓ and w_j^ℓ in the parse tree, and i, j index words in the same sentence.

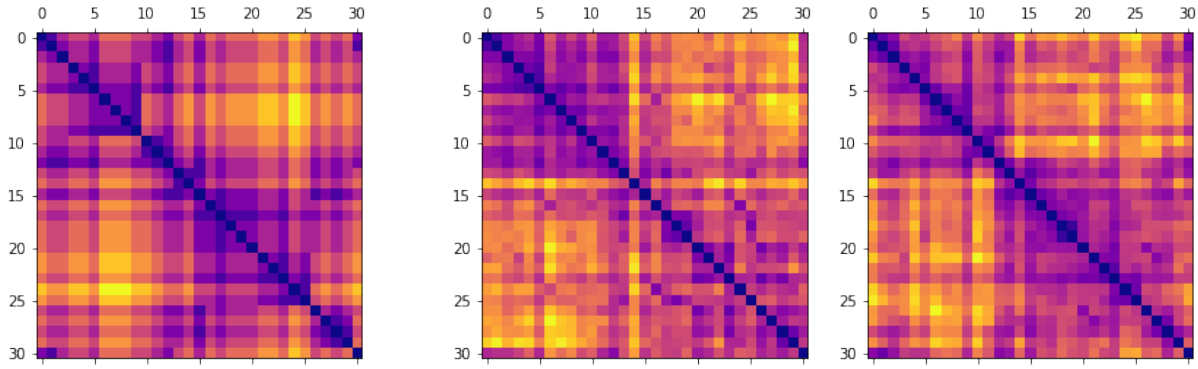
To measure success in this task, undirected unlabeled attachment score (UUAS)—the percent of undirected edges placed correctly against the gold tree—is used. This measure is calculated by first constructing a minimum spanning tree from predicted tree distances.

Besides structural probes for distance between nodes, Hewitt and Manning (2019) also test whether network representation spaces encode tree depth. They again learn a linear transformation B such that the squared norm $\|B\boldsymbol{h}\|^2$ approximates tree depth.

3 Experiments and Results

We compare representations of attention-based model distilGPT-2 (Sanh et al., 2020), a distilled version of GPT-2 (Radford et al., 2019), using Wolf et al. (2020)'s implementation, and Gulordava et al. (2018)'s LSTM language model based on how they encode POS tags and constituency parse trees. For a fairer comparison, we choose the attention-based model GPT-2 as, like the LSTM, it is autoregressive.

Extracting Representations We extract representations of tokens in the Universal Dependencies English Web Treebank (EWT) (Ann Bies et al., 2012), consisting of 254,830 word tokens across 16,624 sentences of web text, with no additional pre-processing. Since GPT-2 constructs representations from word pieces, subword vectors are aligned with EWT tokens, with each EWT token's representation computed as the average of its subword representations. This therefore represents a lower-bound on GPT-2's performance. Each word



(a) Gold Tree Distance (b) DistilGPT-2 Predicted Distance (c) LSTM Predicted Distance

Figure 1: Visualizations of gold standard distance matrix and predicted distance matrices.

Table 1: POS Tag Classification Results. Best selectivity shown in boldface.

	Recurrence	**Attention**
Test Accuracy	89.17%	90.58%
Control Accuracy	62.09%	73.51%
Selectivity	**27.08%**	17.07%

is assigned one of 17 POS tags. Tokens are run through the models sentence by sentence, and representations are given by the models' hidden activations. For each sentence, the LSTM's encoder is initialized using a ". <EOS>" sequence.

For training probing classifiers, we use the Adam optimizer (Kingma and Ba, 2017) with a learning rate of 0.001, and train until the validation loss has stopped improving for 3 epochs.

3.1 POS Tagging

For the POS tagging task, train by mapping each representation h to a probability over POS tags using one linear layer, taking a softmax, and minimizing cross entropy. We use a batch size of 256.

We report POS tag accuracy on the test set for the best layer, according to selectivity, of each model, and their respective control task, in Table 1. We observe that while the attention-based model reaches a higher performance on the task, its selectivity is a lot lower than its counterpart. This is very interesting, as it seems to imply that POS tag information is encoded to a higher degree in the recurrent model's representations than in the attention-based model's representations, even though its performance on the task is (slightly) worse.

Since different layers of the models may encode information differently, we extract representations from each layer. The POS tag accuracy on the test set for each layer of both models, and their respective control tasks, is shown in Figure 2. Notably, we see that control accuracy decreases as layer index increases for both models. We provide possible reasons for this in Section 4. For both models, the last layer yields the highest selectivity. GPT's performance slightly increases in higher layers, whereas the LSTM's performance decreases.

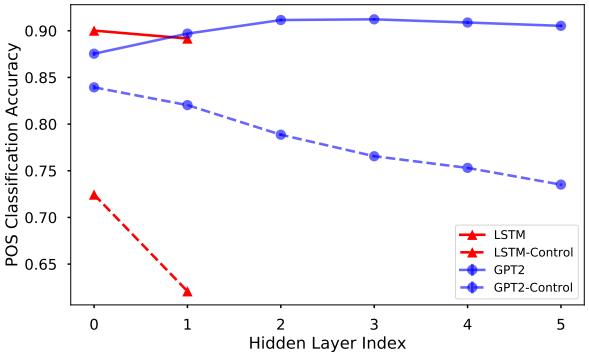


Figure 2: POS tag accuracy across GPT and LSTM layers, including their corresponding control accuracies.

3.2 Structural Edge Probing

For the structural probing task, we map a sequence of representations h^ℓ from a sentence ℓ to a predicted distance for each pair of tokens in the sentence as in Equation (1). We create a minimum spanning tree from these distances to calculate UUAS between the reconstructed tree and the gold parse tree. We minimize the loss given by Equation (2). We also use squared norm of representations to predict parse tree depth, as in Section 2.2. For the depth task, we report Kendall's tau, a tie-adjusted Spearman correlation measure, between real and predicted tree depths. We use a batch size of 24.

Table 2: Structural and Tree Depth Probe Results. Best scores shown in boldface.

	Recurrence	**Attention**
Structure: UUAS	**67.39%**	65.44%
Depth: Kendall's tau	56.20%	**59.80%**

We report UUAS and Kendall's tau on the test set for the final layer of each model in Table 2. We observe that the recurrent model performs better on the structure task and the attention model performs better on the depth task. This is an interesting result, as we would expect a model better at encoding the whole tree to be better at encoding depth as well. However, we found that the correlation measure tends to stay the same even over relatively large differences in error between predicted and real distances, so we are unsure how reliable these kinds of correlation measures are as performance metrics for this task.

To better understand how well these models can reconstruct parse trees, we plot the gold and predicted tree distance matrices in Figure 1. The first matrix shows the actual parse tree distances and the rest shows the parse tree distances as reconstructed by DistilGPT-2 and the LSTM. Although not perfect, we can see that the overall tree structure is captured very well in the reconstructions.

4 Discussion

Related work use probes (Hupkes et al., 2018; Tenney et al., 2019; Blevins et al., 2018) and structural probes (Hewitt and Manning, 2019) to examine learned neural network representations. To our best knowledge, these techniques have not yet been used to explicitly compare recurrent and attention-based network representations.

Qualitative analysis of the POS tagging task predictions reveals some interesting trends. The attention-based model outperforms the recurrent model on rare sequences and tokens such as URLs, timestamps, email headers, internet chat lingo, misspelled words (*"exelent Job"*) and unusual phrases (*"Backdrop stand."*). This is probably due to the nature of the data the models were trained on - DistilGPT-2 was trained on web-crawled text and the LSTM was trained on wikipedia dumps. Since DistilGPT-2 has been trained on larger amounts of data, it can perform better on such outliers.

To our surprise, we find a higher correlation between sentence length and POS tag performance for the recurrent model than for the attention-based model. This goes against common knowledge that recurrent models perform worse on longer sequences, but may also be an artifact of the dataset.

In Figure 2, we see that selectivity increases as we move up the layers for both the recurrent model and the attention-based model. Whereas POS tag classification accuracy does not change very much across the layers, control task accuracy does fall drastically. One plausible answer can be derived from an information theoretic point of view. As the activations travel through the network, entropy in the representations decreases as they become more strongly biased away from a uniform distribution. The presence of biases in deeper layers makes it harder for the classifier to map them to random labels.

Our approach does have some weaknesses. One weakness is that we cannot draw any strong conclusions from our structural probe experiments. While we do find that the recurrent model performs better on this task in terms of UUAS, we do not yet have a control task available for this specific task. Without selectivity, we can't infer whether that information is actually encoded to a larger degree in the LSTM's representations. Also, the metric used to evaluate the quality of tree depth precision is quite unstable, and may not be suitable for this task, which make experiment results for the depth task not as reliable as we would hope.

Another weakness is that our main conclusions are drawn from POS tag experiments. While representations do need to encode syntax for predicting POS tags, it is not the hardest task out there. Ideally, these experiments should be run on a larger set of NLP tasks, such as other syntactic and even semantic tasks.

5 Conclusion

Using probing methods, we compare syntactic representations learned by recurrent and attention-based models. Surprisingly, we find that recurrent models capture POS tag and syntax tree information to a higher degree than attention-based models do. However, these conclusions do not hold as strongly for syntax tree information, as there are no control tasks for structural probes yet, and other NLP tasks may yield different results. Interesting directions for further work are therefore to explore probing for other syntactic and semantic tasks, and to design control tasks for these other tasks.

References

Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English Web Treebank. Technical Report LDC2012T13, Linguistic Data Consortium, Philadelphia.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473 [cs, stat]*. ArXiv: 1409.0473.

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, Melbourne, Australia. Association for Computational Linguistics.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

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)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and Interpreting Probes with Control Tasks. *arXiv:1909.03368 [cs]*. ArXiv: 1909.03368.

John Hewitt and Christopher D. Manning. 2019. A Structural Probe for Finding Syntax in Word Representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, pages 1735–1780.

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. 2018. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *arXiv:1711.10203 [cs]*. ArXiv: 1711.10203.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*. ArXiv: 1412.6980.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *International Conference on Neural Information Processing Systems*, 26:9.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. *Technical report, OpenAI*, page 24.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv:1910.01108 [cs]*. ArXiv: 1910.01108.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. *International Conference on Learning Representations*, page 17.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *arXiv:1706.03762 [cs]*. ArXiv: 1706.03762.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2020. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *arXiv:1910.03771 [cs]*. ArXiv: 1910.03771.