# Investigating transformers and LSTMS using probing

**Jonathan Mitnik**
10911197
j.mitnik@gmail.com
University of Amsterdam

**Pieter de Marez Oyens**
10002403
oyenspieter@gmail.com
University of Amsterdam

## Abstract

In this report, a study is done to compare the encoding abilities of three state-of-the-art language models. This study is done by means of probing, a technique for mapping high-dimensional features to a simple and interpretable feature-space. The results indicate that the LSTM is better at encoding structural relations, while the DistilGPT-2 transformer encodes lexical properties such as POS better.

## 1 Introduction

Complete apprehension of syntactical and semantic structure of a language is the goal for NLP systems. The first successful NLP systems were mostly based on Chomskyan theories (Chomsky, 1957) and created by handcrafting comprehensible symbolic rules (Laporte, 2005), while modern day successes are garnered using deep learning and neural models. Model architectures that have proven successful in language tasks are RNN- and Transformer-models. Even though these models perform well on different language tasks it is difficult to understand how they "think" and how they come to their decisions.

The field of explainable AI (XAI) focuses on interpreting neural models. In this paper, we partially unravel **language models**. These models predict new word tokens based on words that came before and to do this well syntactical and semantical understanding of language is essential. One way of researching *which* linguistic phenomena these models understand is by **probing**.

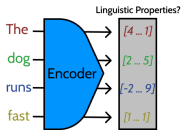


Figure 1: Linguistic properties may be captured in word embeddings.

Probing methods inspect word embeddings of language models, in which theoretically a model encodes the linguistic information (Figure 1) needed to predict a next word. These techniques are however subject to scrutiny: they may find something that is not there. To this end **control tasks** offer a way to analyse probing methods.

Finally, in this paper we implement probing methods and corresponding control tasks. We focus on probing Part-of-speech (POS) and sentence structure. This will be done for different state-of-the-art language model architectures: one LSTM and two Transformers. The results will be two-fold: we provide insight into which model embeds linguistic aspects (better) and why, while providing reproducibility of these tasks by releasing accompanying code.

## 2 Related Work

Core to our task are language models. Since the inception of the NLP-field there have been many advances, especially in recent years with the rise of Deep Neural Language Models. To mention any specific paper would mean not giving credit to others. Instead we reference Otter et al.'s survey that provides an overview of the most important models, advances and applications within the field. Specific to our paper are the GPT2 (Sanh et al., 2020; Radford et al.) and XLM-RoBERTa (Conneau et al., 2020) transformer models. Our LSTM model is by Gulordava et al. and has been trained specifically on structural tasks.

Within the growing field of XAI (Tjoa and Guan, 2019) there is no consensus on explaining methods (Gilpin et al., 2019). However Hupkes et al. has pioneered the probing methods used in this paper, and explores inspecting linguistic properties in word vector spaces. Work that emanated from this paper has focused on uncovering specific linguistic features such as part-of-speech tagging (Tenney et al., 2019a,b), syntax trees (Hewitt and Manning, 2019) among other features (Tenney et al., 2019b). Hewitt and Liang proposed the use of control tasks, enabling grounded reasoning about probing results.

## 3 Methods /Approach

To examine if language models capture linguistic features in word embeddings we use **probing** techniques. Depending on which linguistic feature is probed a different technique is used. To nuance claims about a model's capability of linguistic understanding based on probing results, we perform **control tasks**. A control tasks "corrupts" the golden-labels of data and trains a model on these corrupted labels. It gives rise to a new metric called *selectivity* (Hewitt and Liang, 2019). We perform probing and control tasks on three different models to differentiate between how and why models learn different linguistic features.

For consistency we will use the same mathematical notation as Hewitt and Liang when referring to sentences, words, embeddings, probing and control tasks.

### 3.1 Models

Three models are examined for encoding features: LSTM, DistilGPT-2 (DGPT2) and XLM-RoBERTa (XLMR). For the XLMR model, the base version is considered. Each model produces a word embedding $h_i \in \mathbb{R}^d$, based on the tokenized sentence $\mathbf{x}_{1:T}$. For each model we take the last layer as the *embedding layer* so that a model's full encoding capability is in each embedding. To be consistent with implementations, we use pre-trained models. The Transformer models and their respective tokenizers are provided by the Huggingface library (noa, 2020). The LSTM model code can be found here: (noa). It has been altered so that it only outputs the hidden states for each word.

### 3.2 Probing

Probing methods are simply *diagnostic classifiers* (Hupkes et al., 2018) that are trained on word embeddings. In this research we focus on two probing tasks: POS-tagging and recovering the syntax tree of a sentence.

#### 3.2.1 POS-tagging Probe

For this task the probe tries to correctly predict the correct POS-tag $y_i$, pertaining to word token $x_i$ (Tenney et al., 2019a,b). The probe is implemented as a linear single-layer perceptron as it's performance is on par with more elaborate models while providing *higher* selectivity (Hewitt and Liang, 2019).

#### 3.2.2 Structural Probe

The structural probe attempts to predict the tree distances for each word $x_i$ in a sentence $\mathbf{x_{1:T}}$, given the word embeddings $\mathbf{h}_{1:T}$. By getting these distances it is possible to recreate true tree distances and reconstruct the syntax tree, proving these structures are present in embeddings. Our implementation follows (Hewitt and Manning, 2019), and refer to this article for a detailed explanation.

#### 3.2.3 Dependency Edge Prediction Probe

Like the structural probe, it uncovers whether or not syntax trees are embedded by language models, but is implemented as a classification task like the POS probe. Here, the prediction $y_i$ is the index of the parent of word $x_i$ (Hewitt and Liang, 2019).

### 3.3 Control Tasks

Intuitively, a probe can only perform well on a task if the right linguistic information is encoded in a word embedding. Probes are simple networks tha are capable of learning a task rather than interpreting a representation (Hewitt and Liang, 2019). Control tasks offer us a way to validate if a probe has learned a task or not. A probe is *selective* when a probe performs better on the original task than control task.

#### 3.3.1 POS-tagging Control Task

In POS tagging, we corrupt targets the classifier trains on and assign false POS-tags to each token type (Hewitt and Liang, 2019). We sample these new labels from a uniform distribution over POS-tags. A uniform distribution is the most straightforward to implement (Hewitt and Liang, 2019).

#### 3.3.2 Structural Control Task

This control task was not implemented as the original task is it was difficult there was no literature found on a control tasks for this probe.

#### 3.3.3 Dependency Edge Prediction Control Task

In the same vein as the POS-tagging control task, the targets are corrupted by assigning a word $x_i$ a new parent parent $y_i$. This new parent is sampled uniformly from either itself, first or last token of the sentence it is in(Hewitt and Liang, 2019).

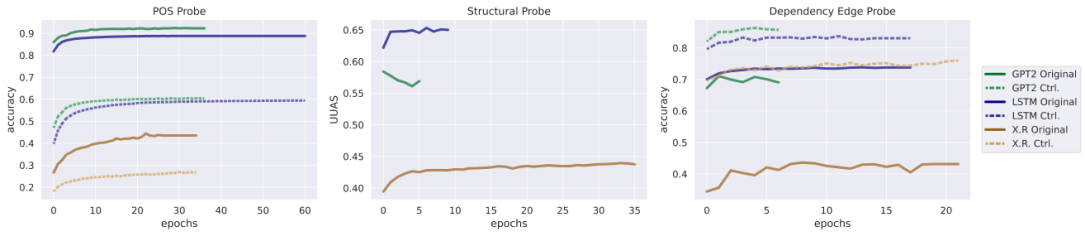


Figure 2: Validation scores during training

## 4 Experiments and Results

### 4.1 Data

We use an English tree-bank which is part of the Universal Dependency Project (Nivre et al.). The data is parsed by the conllu library into the CoNLL-X format (Buchholz and Marsi, 2006) which gives access to linguistic information of words such as POS-tags and dependency relationships, among others. The data is split into a training/validation/test-set with a 75/12/13% ratio.

### 4.2 Metrics

For the tasks of probing POS-tag and dependency edges we measure probe success with the accuracy score:

$$\frac{\#\text{correct classifications}}{\#\text{total classification}}$$

The structural probe's tree prediction is evaluated on undirected attachment score (UUAS)-the percent of undirected edges placed correctly compared to the gold tree (Hewitt and Manning, 2019). Finally, the *selectivity* metric is measured by subtracting the probe's control score from it's original score:

$$\text{selectivity} = \text{original score} - \text{control score}$$

(Hewitt and Liang, 2019). Since we lack a control task for our structural probe, scores equal probe task accuracies.

### 4.3 Probe Models and Hyper-parameters

The POS-tag and dependency edge probes are diagnostic classifiers implemented as linear multiclass neural models. They map a word embedding $h_i$ to a prediction $y_i \sim \arg\max(\text{Softmax}(Ah_i))$, and are trained by minimizing the cross-entropy loss. For the exact model description and training of the structural probe model, we refer to Hewitt and Manning's paper.

The input size for our probing models is equal to the word embedding size of each language model. For the LSTM this is 650, for DGPT-2 and XLMR 768.

The POS model is trained and evaluated using the Skorch library (Tietz et al.) while the structural models are custom trained and evaluated in PyTorch. All models are trained using early stopping with a patience of 4 to ensure models stop training only when converging on a local optimum, on validation. Empirical results for hyper-parameter validation suggest to use a rank for the structural probes of 64.

### 4.4 Quantitative Results

We see the validation scores for each model during training in Figure 2. Probes on all models converge quickly to an optimum on the original task, while control tasks take long to converge. All control tasks perform worse than their original tasks, with GPT2 performing best and XLMR worst.

For the structural probing task, LSTM performers best and XLMR worst. XLMR additionally takes a long time to finish training, inversely of the LSTM and DGPT-2 models.

Contrary to the POS probing tasks, control tasks for dependency edge probing perform better than the original task, this is true for all models. For the original tasks LSTM performs best and again XLMR worst.

In Table 1 we present the evaluation of each trained probe for each model on the test set. We also performed the McNemar's significance test (Dietterich, 1998) on the output of the *POS* task and the *D.E.* task. This test is calculates a contingency table based on the success and negative performances of two classifiers for a particular test-set. For both tasks, the best model is compared with the second-best.

XLMR performs significantly worse than LSTM and DGPT-2 in all tasks. DGPT-2 model has both a higher selectivity and accuracy than the LSTM in the POS task with a p-value of 6.49e-76, indicating statistical significance. The LSTM outperforms

| *Model* | *POS-tagging* | | | *Structural* | | | *D.E. prediction* | | |
|---|---|---|---|---|---|---|---|---|---|
| | **Orig.** | **Ctl** | **Selct.** | **Orig.** | **Ctl** | **Selct.** | **Orig.** | **Ctl** | **Selct.** |
| LSTM | 0.888 | 0.572 | 0.32 | 0.644 | - | - | 0.649 | 0.754 | **-0.104** |
| DGPT-2 | 0.923 | 0.573 | 0.35 | 0.574 | - | - | 0.598 | 0.770 | -0.174 |
| XLMRB | 0.440 | 0.287 | 0.15 | 0.434 | - | - | 0.251 | 0.631 | -0.380 |

Table 1: Probe scores on different linguistic and control tasks on the test set. For the POS-tag and dependency edge task, the scores equal the *accuracy*. For the Structural task, the score equals it's *UUAS* score.

both Transformer models on both the structural and D.E. task. It is notable that the selectivity for all three model-types is negative. The LSTM still has a statistical significant performance over the DGPT-2, with a p-value of 9.99e-17.

### 4.5 Qualitative Results

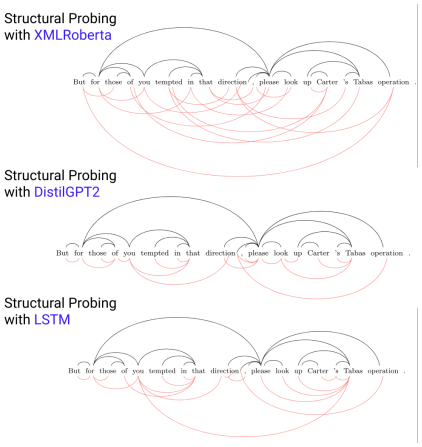


Figure 3: Reconstruction of dependencies by all different language models. Red lines show predicted targets while black show true targets

In Figure 3 we see the same sentence reconstructed by the different models based on probing results after training on the structural task. Empirical evaluations have suggest that the XLMR model more frequently prefers to form long-term connections over short ones. By contrast, the GPT2 model seems to form more connections between sibling words. The LSTM seems to strike a relative middle-ground between forming long and short-term connections.

## 5 Discussion

We first address negative selectivity in the D.E. task: it suggests an ill-formed and over-simplified control task as opposed to the original task. Consider the challenge of classifying a parent node from twenty nodes, over the challenge of classifying one of three potential behaviours. In order to test this tasks more closely, a different control task would have to be considered for the D.E. task, or reconsidered for the standard structural probes. A similarly challenging task would be to assign arbitrary positions in a sentence which are parent-nodes, where each token is preemptively assigned a probability for being a parent. The remainder nodes are randomly assigned to these parent. This allows for the stochastic element of the original task, while maintaining a relation to other nodes.

Secondly, the LSTM outperforming both Transformer model on the structural and D.E. task is surprising due to it being not massively pre-trained. It is however more *specifically* trained on similar structural tasks. As noted in the qualitative results in subsection 4.5, empirical results suggest that the LSTM finds more balanced connections which support this claim. Future research should examine the intermediate layers of the Transformers which may produce different results showing that syntax structure is captured by the model somewhere.

Also notable is the out-performance of GPT2 model on the arguably simpler POS task. This question might also be answered by studying the intermediate layers activation, as it might reveal interesting relations between simple tasks and language translation performances such as what models like the XLMR excel in. To study this better, structural and lexical tasks of different layers would have to be studied in future research to see if the LSTM is limited in its higher performance to specifically the tree-based tasks.

In this paper, the task was to compare new Transformer models with RNNs by means of probing their word embeddings and measuring how well these encode linguistic features. As was found, no model is perfect in one task, but the results suggest that LSTMs encodes syntax-trees better, while GPT-2 POS-tags. We suggest future studies to ask whether the pre-training has an confounding effect by studying other structural tasks, such as co-reference resolution.

## References


facebookresearch/colorlessgreenRNNs. Library Catalog: github.com.

2020. huggingface/transformers. Original-date: 2018-10-29T13:56:00Z.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X Shared Task on Multilingual Dependency Parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City. Association for Computational Linguistics.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton and Co., The Hague.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. *arXiv:1911.02116 [cs]*. ArXiv: 1911.02116.

Thomas G. Dietterich. 1998. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10(7):1895–1923. Publisher: MIT Press.

Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. 2019. Explaining Explanations: An Overview of Interpretability of Machine Learning. *arXiv:1806.00069 [cs, stat]*. ArXiv: 1806.00069.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. *arXiv:1803.11138 [cs]*. ArXiv: 1803.11138.

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.

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.

Eric Laporte. 2005. Symbolic Natural Language Processing. In Lothaire, editor, *Applied Combinatorics on Words*, pages 164–209. Cambridge University Press.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. Universal Dependencies v1: A Multilingual Treebank Collection. page 8.

Daniel W. Otter, Julian R. Medina, and Jugal K. Kalita. 2019. A Survey of the Usages of Deep Learning in Natural Language Processing. *arXiv:1807.10854 [cs]*. ArXiv: 1807.10854.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. 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, Dipanjan Das, and Ellie Pavlick. 2019a. BERT Rediscovers the Classical NLP Pipeline. *arXiv:1905.05950 [cs]*. ArXiv: 1905.05950.

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. 2019b. What do you learn from context? Probing for sentence structure in contextualized word representations. *arXiv:1905.06316 [cs]*. ArXiv: 1905.06316.

Marian Tietz, Daniel Nouri, and Benjamin Bossan. skorch 0.8.0.

Erico Tjoa and Cuntai Guan. 2019. A Survey on Explainable Artificial Intelligence (XAI): Towards Medical XAI. *arXiv:1907.07374 [cs]*. ArXiv: 1907.07374.