NLP2 Project: *Probing Language Models*

Teaching assistant: *Jaap Jumelet*

March 31, 2020

1 Introduction

Motivation NLP ultimately aims to develop systems that possess a truly comprehensive understanding of language, an understanding that is able to systematically generalise to new scenarios, based on a fundamental apprehension of the core structures of language. Contemporary approaches adhere to an *unsupervised* paradigm, driven by enormous amounts of data, which has led to a series of breakthroughs in the field. These successes have been supported by an exponential increase in computing power, and by using *deep learning* models such as neural networks. However, employing neural networks comes at a great loss of transparency; while their impressive performance is commendable, it is no longer evident how these models operate. This has given rise to a new line of research that aims to uncover the internal dynamics of these models, in a similar manner to how psycholinguistics attempts to unravel the mysteries of human language processing.

In this assignment, we will focus our investigation on **language models**. These are models that assign a probability to the next token in a sentence, conditioned on a prior context. To do this proficiently, they should be able to thoroughly grasp a sentence on both a syntactic and semantic level: by keeping track of information on the subject, the topic of the sentence, the action being performed, etc. This makes these systems perfectly suited for investigating current NLP systems, as there are few tasks that require such a comprehensive understanding of language as language modelling does.

In recent years, there has been considerable interest into analysing the linguistic capacities of language models. Most of these analyses approach language modelling from a behavioural angle, investigating a model's output behaviour on a specific linguistic phenomenon (e.g. Gulordava et al. (2018) and Marvin and Linzen (2018)). This approach has yielded a substantial understanding of *what* phenomena these models understand, but does not necessarily provide cues on *how* these phenomena are processed.

Assignment The linguistic dynamics of a language model can be assessed in several ways. In this assignment we will focus on using **probing tasks**. These tasks allow us to *probe* a model's representations using *diagnostic classifiers* (Hupkes et al., 2018).[1] Diagnostic classifiers are simple (i.e. often linear) classifiers that are trained on top of the representations of a model. This allows us to uncover (linguistic) properties that are encoded in a representation.

The assignment is divided into 3 sub-tasks:

1. First, you will focus on probing **linguistic properties**, focusing in particular on probing part-of-speech (POS) tags. Stated concretely: you will train a classifier f that maps a model's representation h to a corresponding POS tag t: $f(h) \rightarrow t$.

2. Next, you will focus on **structural probes** (Hewitt and Manning, 2019). Structural probes allow the internal hierarchical structure of a sequence of representations to be uncovered. This makes it possible to retrieve the parse tree of a sentence based only on the intermediate hidden representations (Figure 1).

[1] *Diagnostic classifiers* are also referred to as *probes*, but we will adhere to the former term as it has been proposed by researchers from the UvA itself :-).

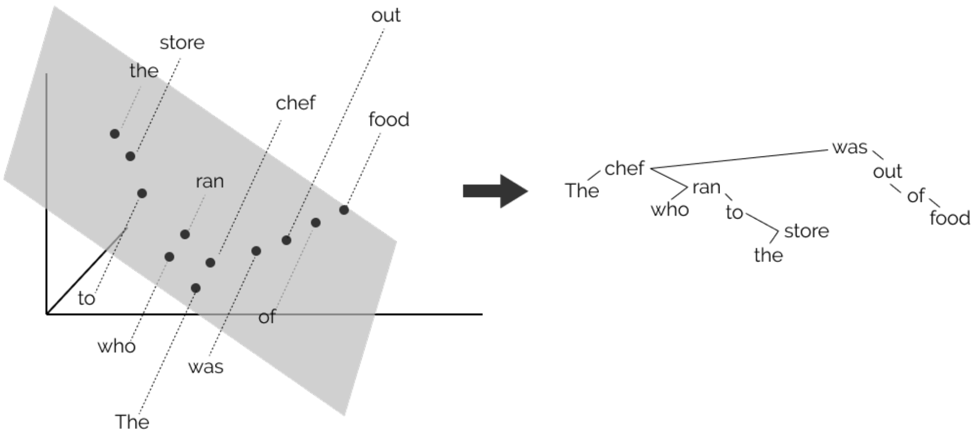

Figure 1: Can we extract a hierarchical ordering based on a language model's high-dimensional representations?

3. Finally, you will use **control tasks** (Hewitt and Liang, 2019) to qualitatively validate your findings. Probing a representation can namely easily lead to retrieving *false positives*: due to their high-dimensional nature it is possible that the diagnostic classifier identifies properties that are not actively being encoded by the model.

Models We will consider 2 types of language models, that will form the core of your main research question. **Recurrent** models process a sentence incrementally, while keeping track of an intermediate hidden state. The most common recurrent architecture is the LSTM (Hochreiter and Schmidhuber, 1997), but other recurrent architectures such as the GRU (Cho et al., 2014) are often used as well. Recent years have seen a drastic surge in the use of **Transformer** architectures (Vaswani et al., 2017), based on the use of *self-attention*. These models are better able to handle long-distance dependencies, and allow for more optimal parallelisation during training. What both architectures have in common, is that their input data is *unstructured*, making it challenging to to process the inherently *hierarchical* nature of natural language. Structural probes provide an excellent opportunity to assess a model's capacities in encoding this hierarchical nature.

We will not be training these language models ourselves, but instead focus on models that have been trained already. **For your assignment you will choose and compare (at least) one recurrent and (at least) one attention-based model.** Transformer models have been made easily accessible by the excellent `transformer` library of Huggingface.[2] We suggest the following Transformer LMs:[3]

- **GPT-2** (Radford et al., 2019), or its *distilled* version (Sanh et al., 2019).[4]
- **XLNet** (Yang et al., 2019).
- **CTRL** (Keskar et al., 2019).
- **Transformer-XL** (Dai et al., 2019).
- **BART** (Lewis et al., 2019).
- **T5** (Raffel et al., 2019).

[2] `https://github.com/huggingface/transformers`

[3] Note that neither BERT nor ELMo have been trained with an auto-regressive language modelling objective, and are therefore unsuited for this assignment.

[4] `https://github.com/huggingface/transformers/tree/master/examples/distillation`

The LSTM model you will investigate is the LSTM model made available by Gulordava et al. (2018). However, you are free to incorporate different LSTM models as well, such as the one of Józefowicz et al. (2016). The assignment is accompanied by a Jupyter notebook that will get you up and running.

Deliverables

1. Jupyter notebook, due April 24, 2020. The notebook should contain the entire pipeline for your experimental setup. Functions or classes are allowed to be defined in Python files externally, as long as the main functionality is listed in the notebook.
2. Short paper due April 24, 2020. The short paper should contain **four** pages (references excluded). A suggested page distribution is as follows:
   (a) **Abstract**: provide a (very) concise overview of your approach, and highlight your key findings.
   (b) **Introduction**: introduce the reader to your research area, summarise your contributions and highlight the relevance of your research (0.6 pages);
   (c) **Related Work**: summarise research papers relevant for your work (i.e. probing, interpretability, language modelling, or whatever you deem relevant to your approach). Be brief, since this is a short paper (0.4 pages);[5]
   (d) **Methods/Approach**: describe your approach, and highlight important research decisions you made along the way (1 page);
   (e) **Experiments and Results**: detail the precise experimental setup used and the numerical results your models achieved (1 page);
   (f) **Discussion**: discuss your results, in an honest and self-critical manner. Provide a future outlook, and highlight your paper's strengths and weaknesses (1 page).
3. Poster presentation, due April 22, 2020. Compress the paper's content into a single-page poster that could be presented at a conference. Support the textual content through visual aids, such as tables and graphs that facilitate fast understanding of the paper's contributions and main results.

   For this project, make sure to highlight the differences you found between recurrent and attention-based models, both quantitatively (UUAS, correlations) and qualitatively (e.g. salient examples).

Useful resources

- John Hewitt's blog post about his paper on structural probes:
  `https://nlp.stanford.edu/~johnhew/structural-probe.html`
- John Hewitt's blog post about his paper on control tasks:
  `https://nlp.stanford.edu/~johnhew/interpreting-probes.html`
- The `transformer` library of Huggingface:
  `https://github.com/huggingface/transformers`
- The `diagnnose` library, written by me (Jaap), that will aid you during your research:
  `https://github.com/i-machine-think/diagnnose`
  `https://diagnnose.readthedocs.io`
- This blog post on Transformers, that will freshen up your mind on what makes these models so powerful:
  `http://jalammar.github.io/illustrated-transformer/`
- This blog post on LSTMs:
  `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

[5] Short papers sometimes place their related work in their discussion section, allowing you to place your contributions more directly in the context of your peers. This is up to you, the structure of (Hewitt and Manning, 2019) might serve as a useful guideline

2 Suggested Schedule

To stay on track, we recommend adhering to the following schedule. The lab sessions will start with a short presentation on the week's topic, containing references to the recommended reads to serve as inspiration and extend your knowledge of interpretability of neural networks.

2.1 Week 1: Linguistic Properties

Reading Read the work Tenney et al. (2019b), who demonstrate a clear approach how diagnostic classifiers can obtain linguistic information that is encoded in a model's representations. In case this paper has heavily peaked your interest, we recommend reading Tenney et al. (2019a) as well, which provides a more in-depth look into these probing methods by applying probes to each individual layer of a BERT model. Have a look at the `transformer` and `diagnnose` libraries.

Coding

- Set up the pipeline for activation extraction.
- Set up the pipeline for training a diagnostic classifier.
- Probe your models to what extent **POS tag** information is encoded in its representations.

I will provide you a Jupyter notebook that will bring you up to speed, and contains more information about training data and model imports.

Writing It might be a good idea to already start on a draft of related work and your introduction. Make sure to get a good understanding of the larger picture (i.e. hierarchical processing in recurrent vs. attention-based models).

2.2 Week 2: Edge Probing

Reading Read the work Hewitt and Manning (2019), and make sure you thoroughly grasp their approach.

Coding

- Set up the pipeline and train your own structural probing classifiers.
- Set up tools to reconstruct a parse tree based on the result of your classifier, and evaluate it on a gold standard parse tree.

Writing Start working on your Methods (or Approach) section, as well as your Experiments section.

2.3 Week 3: Control Tasks

Reading Read the work of (Hewitt and Liang, 2019), and think about how you can incorporate their method into your own research.

Coding

- Set up the pipeline for control tasks.

Writing By now you should have obtained most of your results, and you can start working on your Results section.

2.4 Week 4: Wrapping Up

Wrap up experiments, and finish writing. Think about (and discuss with me) how you can translate your experimental findings towards a clear conclusion, that is aided both quantitatively (UUAS, correlations, etc.) as well as qualitatively: salient examples can strongly support your main conclusions.

References

Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *arXiv preprint arXiv:1406.1078*.

Dai, Zihang, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov (2019). "Transformer-xl: Attentive language models beyond a fixed-length context". In: *arXiv preprint arXiv:1901.02860*.

Gulordava, Kristina, 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, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pp. 1195–1205.

Hewitt, John and Percy Liang (Nov. 2019). "Designing and Interpreting Probes with Control Tasks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 2733–2743.

Hewitt, John and Christopher D. Manning (June 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)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4129–4138.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780.

Hupkes, Dieuwke, Sara Veldhoen, and Willem H. Zuidema (2018). "Visualisation and 'Diagnostic Classifiers' Reveal How Recurrent and Recursive Neural Networks Process Hierarchical Structure". In: *J. Artif. Intell. Res.* 61, pp. 907–926.

Józefowicz, Rafal, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu (2016). "Exploring the Limits of Language Modeling". In: *CoRR* abs/1602.02410.

Keskar, Nitish Shirish, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher (2019). "Ctrl: A conditional transformer language model for controllable generation". In: *arXiv preprint arXiv:1909.05858*.

Lewis, Mike, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer (2019). "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension". In: *arXiv preprint arXiv:1910.13461*.

Marvin, Rebecca and Tal Linzen (2018). "Targeted Syntactic Evaluation of Language Models". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 1192–1202.

Radford, Alec, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever (2019). "Language Models are Unsupervised Multitask Learners". In:

Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu (2019). "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *arXiv preprint arXiv:1910.10683*.

Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf (2019). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv preprint arXiv:1910.01108*.

Tenney, Ian, Dipanjan Das, and Ellie Pavlick (July 2019a). "BERT Rediscovers the Classical NLP Pipeline". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 4593–4601.

Tenney, Ian, 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”. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019.*

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). “Attention is All you Need”. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 5998–6008.

Yang, Zhilin, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le (2019). “Xlnet: Generalized autoregressive pretraining for language understanding”. In: *Advances in neural information processing systems*, pp. 5754–5764.