diagNNose: A Library for Neural Activation Analysis

Jaap Jumelet
Institute for Logic, Language and Computation, University of Amsterdam
j.w.d.jumelet@uva.nl

Abstract

In this paper we introduce diagNNose, an open source library for analysing the activations of deep neural networks. diagNNose contains a wide array of interpretability techniques that provide fundamental insights into the inner workings of neural networks. We demonstrate the functionality of diagNNose with a case study on subject-verb agreement within language models. diagNNose is available at https://github.com/i-machine-think/diagnnose.

1 Introduction

We introduce diagNNose, an open source library for analysing deep neural networks. The diagNNose library allows researchers to gain better insights into the internal representations of such networks, providing a broad set of tools of state-of-the-art analysis techniques. The library supports a wide range of model types, with a main focus on NLP architectures based on LSTMs (Hochreiter and Schmidhuber, 1997) and Transformers (Vaswani et al., 2017).

Open-source libraries have been quintessential in the progress and democratisation of NLP. Popular packages include HuggingFace's transformers (Wolf et al., 2019) – allowing easy access to pre-trained Transformer models; jiant (Pruksachatkun et al., 2020) – focusing on multitask and transfer learning within NLP; Captum (Kokhlikyan et al., 2020) – providing a range of feature attribution methods; and LIT (Tenney et al., 2020) – a platform for visualising and understanding model behaviour. We contribute to the open-source community by incorporating several **interpretability** techniques that have not been present in these packages.

Recent years have seen a considerable interest in improving the understanding of how deep neural networks operate (Linzen et al., 2019). The high-dimensional nature of these models makes it notoriously challenging to untangle their inner dynamics. This has given rise to a novel subfield within AI that focuses on interpretability, providing us a peak inside the black box. diagNNose aims to unify several of these techniques into one library, allowing interpretability research to be conducted in a more streamlined and accessible manner.

diagNNose's main focus lies on techniques that aid in uncovering linguistic knowledge that is encoded within a model's representations. The library provides abstractions that allow recurrent models to be investigated in the same way as Transformer models, in a modular fashion. It contains an extensive **activation extraction** module that allows for the extraction of (intermediate) model activations on a corpus. The analysis techniques that are currently implemented include:

- **Targeted syntactic evaluation tasks**, such as those of Linzen et al. (2016) and Marvin and Linzen (2018).
- **Probing** with **diagnostic classifiers** (Hupkes et al., 2018; Adi et al., 2016), and **control tasks** (Hewitt and Liang, 2019).
- **Feature attributions** that retrieve a feature's contribution to a model prediction (Lundberg and Lee, 2017; Murdoch et al., 2018). Our implementation is model-agnostic, which means that any type of model architecture can be explained by it.

In this paper we present both an overview of the library, as well as a case study on subject-verb agreement within language models. We first present a brief overview of interpretability within NLP and a background to the analysis techniques that are part of the library (Section 2). We then provide an overview of diagNNose and expand briefly on its individual modules (Section 3). We

Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, pages 342–350

conclude with a case study on subject-verb agreement, demonstrating several of diagNNose's features in an experimental setup (Section 4).

2 Background

The increasing capacities of language models (and deep learning in general) have led to a rich field of research that aims to gain a better understanding of how these models operate. Approaches in this research area are often interdisciplinary in nature, borrowing concepts from fields such as psycholinguistics, information theory, and game theory. diagNNose provides support for several influential analysis techniques, for which we provide a brief background here.

2.1 Targeted syntactic evaluations

Language models have stood at the basis of many successes within NLP in recent years (Peters et al., 2018; Devlin et al., 2019). These models are trained on the objective of predicting the probability of an upcoming (or masked) token. In order to succeed in this task, these models need to possess a notion of many different linguistic aspects, such as syntax, semantics, and general domain knowledge. One popular line of research that tries to uncover a model's linguistic capacities does this via so-called targeted syntactic evaluations (Linzen et al., 2016; Gulordava et al., 2018; Marvin and Linzen, 2018; Jumelet and Hupkes, 2018). This type of analysis compares a model's output on minimally different pairs of grammatical and ungrammatical constructions. If it assigns a higher probability to the grammatical construction, the model is said to possess a notion of the underlying linguistic principles, such as subject-verb agreement or NPI licensing:

(1) a. The **ladies** near John **walk**.
b. * The **ladies** near John walks.

(2) a. **Nobody** has **ever** been there.
b. * Someone has **ever** been there.

diagNNose supports a wide range of syntactic tasks, as well as an interface that allows new tasks to be added without effort.

2.2 Diagnostic Classifiers

A second line of work tries to assess a model's understanding of linguistic properties – such as part-of-speech tags or number information – by directly training **diagnostic classifiers** on top of its representations (Hupkes et al., 2018; Adi et al., 2016; Belinkov et al., 2017). This type of analysis, also referred to as **probing**, has led to numerous insights into the inner workings of language models (Liu et al., 2019a; Tenney et al., 2019). The activations diagnostic classifiers are trained on are not restricted to just the hidden states of a language model at their top layer: this can, for instance, also be done on the individual gate activations to reveal patterns at the cell-level of a model (Giulianelli et al., 2018; Lakretz et al., 2019).

Recently, it has been a topic of discussion to what extent a high accuracy of a diagnostic classifier signifies that that property is actively being encoded by the model. Several solutions to assess this have been proposed, such as training a diagnostic classifier on a baseline of random labels (called a *control task* (Hewitt and Liang, 2019)), or based on the minimum description length of the classifier, a concept from information theory (Voita and Titov, 2020; Pimentel et al., 2020). diagNNose currently facilitates the training of diagnostic classifiers, as well as training control tasks alongside them.

2.3 Feature Attributions

Although probing allows us to uncover specific properties that are embedded within the model representations, it is unable to explain *how* a model transforms its input features into a successful prediction. This question can be addressed by computing the input **feature contributions** to a subsequent output. This is a challenging task, as the high-dimensional, non-linear nature of deep learning models prevents us from expressing these contributions directly on the basis of the model parameters.

Feature attributions can be computed in different ways. One common approach to this task is based on a concept that stems from cooperative game theory, called the Shapley value (Shapley, 1953). A Shapley value expresses the contribution of a player (in our case an input feature) to the outcome of game (in our the case a model prediction). Computing Shapley values is computationally expensive, and several approximation algorithms have therefore been proposed, such as SHAP (Lundberg and Lee, 2017), and Integrated Gradients (Sundararajan et al., 2017). diagNNose currently facilitates the computation of feature attributions using a technique called Contextual Decomposition (Murdoch et al., 2018), and its generalisation as proposed by Jumelet et al. (2019).

3 Library Overview

3.1 Modules

The library is structured into several modules that can be used as building blocks for an experimental pipeline. We provide an overview of a possible experimental pipeline in Figure 1.

3.1.1 Core modules

The following core modules stand at the basis of the different pipelines that can be build on top of `diagNNose`.

`models` We provide an abstraction over language models, enabling recurrent and Transformer models to derive from the same interface. Importing pre-trained Transformer models is done via the `transformers` library. For recurrent models we provide a wrapper that enables access to intermediate activations, including gate activations. We also provide functionality that allows to set the initial hidden states of recurrent LMs, based on a sentence or corpus.[1]

`corpus` Corpora are imported as a `Dataset` from the `torchtext` package. A Corpus can be transformed into an iterator for processing. Tokenization is done using the `transformers` tokenizers, allowing tokenization to be done in both a traditional token-per-token fashion, or based on subword units, such as byte pair encodings (Sennrich et al., 2016).

`extract` Central to most of the analysis modules is the extraction of activations. We provide an `Extractor` class that can extract the activations of a model given a corpus. Thanks to our model wrappers activation extraction is not restricted to just the top layer of a model; intermediate (gate) activations can be extracted as well. To facilitate the extraction of larger corpora with limited computational resources, activations can be dumped dynamically to disk.

`activations` Extracted activations can easily be retrieved using a `ActivationReader`, providing access to activations that correspond to a specific subset of corpus sentences. We also provide functionality for extracting only a specific subset of activations, based on sentence and token information. This way it is possible, for instance, to only extract the activations at the position of tokens of particular interest.

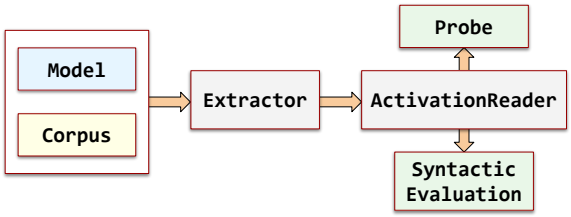

Figure 1: Pipeline stages for conducting syntactic evaluation and probing experiments. Note the modular nature of the pipeline: activations need only to be extracted once, after which the setup of the analysis experiments can be fine-tuned effortlessly.

`config` The pipeline of `diagNNose` is configuration-driven. Configuration is defined in JSON format, but individual attributes can also be set from the command line directly.

3.1.2 Analysis modules

We currently support three main types of experimental modules. We provide a graphical overview of these modules in Figure 2.

`syntax` The library provides functionality for a large suite of targeted syntactic evaluation tasks. Currently we provide support for the following tasks:

- The subject-verb agreement corpus of Linzen et al. (2016), for which we also provide more fine-grained attractor conditions;
- The wide range of linguistic expressions of Marvin and Linzen (2018);
- The subject-verb agreement tasks of Lakretz et al. (2019);
- The NPI corpus of Warstadt et al. (2019);
- The stereotypically gendered anaphora resolution corpus of Jumelet et al. (2019), based on the original WinoBias corpus of Zhao et al. (2018).

Furthermore, the current implementation permits similar types of tasks to be easily added, and we plan on incorporating a larger set of tasks in the near future.

`probe` We provide easy tooling for training diagnostic classifiers (Hupkes et al., 2018; Adi et al., 2016) on top of extracted activations, to probe for linguistic information that might be embedded within them. Our extraction module facilitates

[1] As has been noted by Jumelet et al. (2019), LSTM LMs perform better when initialised with the phrase ". <eos>", instead of zero-valued vectors.

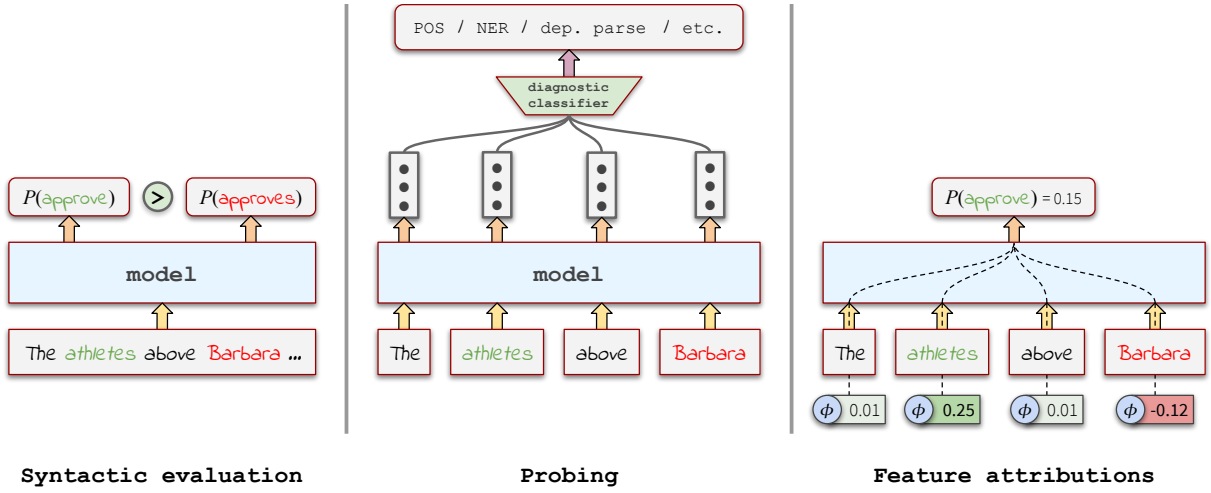

Figure 2: Schematic overview of three different types of experiments that are supported by diagNNose.

training diagnostic classifiers on top of intermediate activations as well, including gate activations. In recent years it has been pointed out that a high probing accuracy does not necessarily imply that linguistic information is actively being encoded by a model. To address this we have incorporated functionality for Control Tasks (Hewitt and Liang, 2019), providing more qualitative insights.

attribute We provide functionality for model-agnostic feature attributions, that allow the output of a model to be decomposed into a sum of contributions. This is achieved by implementing a wrapper over the operations of PyTorch[2], allowing intermediate feature contributions to be propagated during a forward pass in the model. Our implementation provides a basis for many Shapley-based attribution methods, as it allows different approximation methods to be tested easily. We currently facilitate the approximation procedure of (Generalised) Contextual Decomposition (Murdoch et al., 2018; Jumelet et al., 2019), Shapley sampling values (Castro et al., 2009), and the exact computation of propagated Shapley values. Our implementation is the first model-agnostic implementation of Contextual Decomposition: previous implementations were dependent on a fixed model structure.

3.2 Requirements

diagNNose is released on pip and can be installed using `pip install diagnnose`, or directly cloned from the GitHub repository: https://github.com/i-machine-think/diagnnose. The library supports Python 3.6 or later, and its core dependencies are PyTorch (Paszke et al., 2019) (v1.5+), `torchtext`[3], and HuggingFace's `transformers` (Wolf et al., 2019). diagNNose is released under the MIT License (Open Source Initiative, 2020). diagNNose runs both on CPUs and GPUs, and has especially been optimised for smaller consumer setups, due to limited computational resources during development.

The diagNNose code base is fully typed using Python type hints. The code is formatted using *Black*.[4] All methods and classes are documented, and an overview of this documentation can be found on https://diagnnose.readthedocs.io.

4 Case Study: Subject-Verb Agreement

To demonstrate the functionality of diagNNose we will consider the subject-verb agreement corpora of Lakretz et al. (2019) on a set of language models. For our experiments we consider the following models: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), DistilRoBERTa (Sanh et al., 2019), and the LSTM language model of Gulordava et al. (2018).

4.1 Corpora

The corpora of Lakretz et al. (2019) are formed by seven tasks of template-based syntactic constructions. These constructions contain an "agreement attractor" in between the subject and the verb, which might trick a language model into predicting the incorrect number of the verb. A model thus

[2]The wrapper is defined based on the `__torch_function__` functionality that has been introduced in PyTorch 1.5.

[3]https://pytorch.org/text/

[4]https://github.com/psf/black

needs to possess a strong notion of the structure of a sentence: nouns within a prepositional phrase, for instance, should have no impact on the number of the main verb in a sentence.

The seven tasks are defined by the following templates:

SIMPLE
The **athletes approve**

ADV
The **uncle** probably **avoids**

2ADV
The **athlete** most probably **understands**

COADV
The **farmer** overtly and deliberately **knows**

NAMEPP
The **women** near John **remember**

NOUNPP
The **athlete** beside the tables **approves**

NOUNPPADV
The **aunt** behind the bikes certainly **knows**

Each task contains 600 to 900 distinct sentences. Sentences are split up into multiple conditions based on the number of the subject, and the number of the intervening noun phrase. The NOUNPP corpus, for instance, is split up into 4 conditions:

SS: The **athlete** beside the table **approves**

SP: The **athlete** beside the tables **approves**

PS: The **athletes** beside the table **approves**

PP: The **athletes** beside the tables **approves**

To test these corpora on a recurrent model, we first compute the model's hidden state at the position of the verb by feeding it the sub-sentence up till that position. Based on this hidden state we compute the output probabilities of the verb of the correct number ($v^{\checkmark}$), and the incorrect number ($v^{\text{✗}}$), and compare these:

$$P(v^{\checkmark} \mid h_t) > P(v^{\text{✗}} \mid h_t)$$

For bi-directional masked language models, such as BERT, we can not compute a model's intermediate hidden state by passing it a sub-sentence, because these models also incorporate the input of future tokens. To solve this, we replace the verb in each sentence with a `<mask>` token, and assess the model's probabilities at the position of this token.

```
# distilroberta_syntax.json
{
  "model": {
    "model_name": "distilroberta-base",
    "mode": "language_modeling"
  },
  "syntax": {
    "config": {
      "lakretz": {
        "path": "path_to_corpora/"
      }
    }
  }
}

# syntax.py
from diagnnose.config import create_config_dict
from diagnnose.syntax import SyntacticEvaluator
from diagnnose.models import import_model
from diagnnose.tokenizer import create_tokenizer

config_dict = create_config_dict()

model = import_model(config_dict)

tokenizer = create_tokenizer(config_dict)

suite = SyntacticEvaluator(
  model, tokenizer, **config_dict["syntax"]
)

suite.run()
```

(3a) Example setup for running the targeted syntactic evaluation tasks of Lakretz et al. (2019) on DistilRoBERTa.

```
# distilroberta_attribute.json
{
  "model": {
    "model_name": "distilroberta-base",
    "mode": "language_modeling"
  }
}

# attribute.py
from diagnnose.attribute import (
  Explainer, ShapleyDecomposer
)
from diagnnose.config import create_config_dict
from diagnnose.models import import_model
from diagnnose.tokenizer import create_tokenizer

config_dict = create_config_dict()

model = import_model(config_dict)

tokenizer = create_tokenizer(config_dict)

decomposer = ShapleyDecomposer(model)
explainer = Explainer(decomposer, tokenizer)

explainer.explain(
  ["The athletes above Barbara <mask>."],
  ["approve", "approves"]
)
```

(3b) Example setup for creating the feature attributions of DistilRoBERTa on a sentence from the NAMEPP corpus of Lakretz et al. (2019).

Corpus	Condition	BERT	RoBERTa	DistilRoBERTa	LSTM
SIMPLE	S	**100**	**100**	**100**	**100**
	P	**100**	**100**	**100**	**100**
ADV	S	**100**	**100**	**100**	**100**
	P	**100**	**100**	**100**	99.6
2ADV	S	**100**	**100**	**100**	99.2
	P	**100**	**100**	**100**	99.3
COADV	S	**100**	**100**	**100**	98.7
	P	**100**	**100**	**100**	99.3
NAMEPP	SS	93.0	75.7	81.5	**99.3**
	PS	**88.4**	65.9	32.4	68.9
NOUNPP	SS	95.7	88.9	98.1	**99.2**
	SP	**93.3**	84.7	91.1	87.2
	PS	**96.7**	90.6	85.3	92.0
	PP	**100**	**100**	**100**	99.0
NOUNPPADV	SS	99.6	**100**	**100**	99.5
	SP	99.2	99.8	**100**	91.2
	PS	**100**	**100**	**100**	99.2
	PP	**100**	**100**	**100**	99.8

Table 1: Results of the targeted syntactic evaluation tasks of Lakretz et al. (2019).

Modern language models often make use of BPE tokenization that might split a word into multiple sub-words. In our experiments we therefore only compare verb forms for which both the plural and singular form are split into a single token.[5]

4.2 Targeted syntactic evaluations

We run the targeted syntactic evaluation suite on all the 7 templates. An example configuration and script of this experiment in provided in Figure 3a. To run the experiment on a different model, the only configuration that needs to be changed is the `model_name`. The results of the experiment are shown in Table 1.

It can be seen that the Transformer language models generally achieve higher scores than the LSTM model. Interestingly, the NAMEPP task poses a challenge for all models, and both RoBERTa and DistilRoBERTa score lower on this task than the LSTM. A second point of interest is the difference in performance between RoBERTa and DistilRoBERTa on the NAMEPP and NOUNPP tasks. Even though DistilRoBERTa has been trained to emulate the behaviour of RoBERTa, its performance on a downstream task like this differs significantly. These results can provide a starting point for a more fine-grained analysis, such as creating the feature attributions of a model on a specific template.

[5]The RoBERTa tokenizer, for example, splits "confuses" into "conf" + "uses", and "confuse" into "confuse". Comparing the model probabilities for these two forms directly is hence not possible.

4.3 Feature attributions

To gain a better insight into why the language models struggle so strongly with the NAMEPP corpus, we run the feature attribution module on these constructions. An example configuration of this experimental setup is provided in Figure 3b. The results for the experiment are shown in Figure 4.

We show the attributions for DistilRoBERTa on an example sentence from the corpus, which highlights the difference in impact of the intervening attractor on the number of the verb. The results should be interpreted as follows: the score at the top of the attribution denotes the *full logit* of the model for that class, these are the logits that are transformed into probabilities using SoftMax. This logit is *decomposed* in a sum of contributions, which we denote at the bottom of each token. It can be validated that the contributions sum up together to the logit. This is an important property of feature attribution methods – called *efficiency* – that warrants a certain degree of faithfulness of an explanation to the model. A negative value indicates a negative feature contribution to an output class: the impact of that feature led to a decreased preference for the class. Feature attributions also

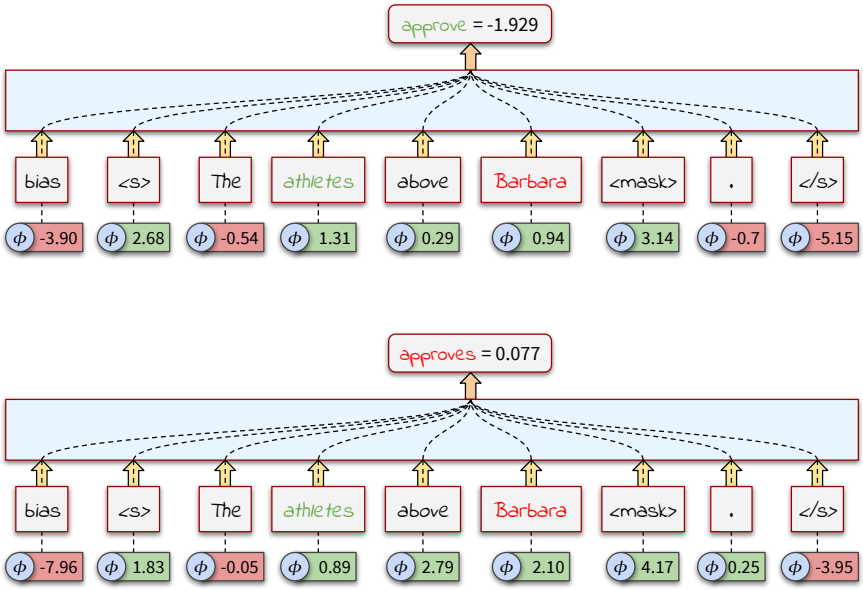

Figure 4: The feature attributions for DistilRoBERTa on an example sentence from the NAMEPP task of Lakretz et al. (2019). The logits of two output tokens, '*approve*' and '*approves*', are decomposed into a sum of contributions.

include the influence of model biases: an aggregate of all information that is statically present within the network such as weight intercepts.

On the presented example sentence, DistilRoBERTa makes an incorrect prediction: the logit of the incorrect singular form '*approves*' is larger than that of the plural '*approve*'. The model's misstep in predicting the correct verb form arrives from the fact that the subject '*athletes*' provided not enough contribution to overrule the negative contributions stemming from other input features. A model that has a thorough understanding of subject-verb agreement should assign a larger contribution to the subject when predicting the main verb: the number signal provided by the subject should be propagated strongly enough to overrule other interfering signals.

The `attribute` module is still in active development. The exponential nature of computing Shapley values makes creating these explanations a challenging task, and we look forward to incorporate other techniques that aim to alleviate the computing costs.

5 Conclusion

`diagNNose` provides essential tools for conducting interpretability research, providing cutting edge analysis techniques such as diagnostic classifiers and feature attributions. The modular design of the library allows complex hypotheses to be tested rapidly, and provides a solid basis for the development of novel interpretability techniques. The library code is open source and welcomes others to contribute: we are eagerly looking forward to collaborate on adding new features to the library.

Acknowledgments

The author gratefully acknowledges the feedback received from Dieuwke Hupkes during the development of the library.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*.

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.

Javier Castro, Daniel Gómez, and Juan Tejada. 2009. Polynomial calculation of the shapley value based on sampling. *Computers Operations Research*, 36(5):1726 – 1730. Selected papers presented at the

Tenth International Symposium on Locational Decisions (ISOLDE X).

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 1 (Long and Short Papers)*, pages 4171–4186.

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. 2018. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 240–248.

Kristina Gulordava, Piotr Bojanowski, Édouard 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.

John Hewitt and Percy Liang. 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)*, pages 2733–2743.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

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.

Jaap Jumelet and Dieuwke Hupkes. 2018. Do language models understand anything? on the ability of lstms to understand negative polarity items. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 222–231.

Jaap Jumelet, Willem Zuidema, and Dieuwke Hupkes. 2019. Analysing neural language models: Contextual decomposition reveals default reasoning in number and gender assignment. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 1–11, Hong Kong, China. Association for Computational Linguistics.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. 2020. Captum: A unified and generic model interpretability library for pytorch. *CoRR*, abs/2009.07896.

Yair Lakretz, German Kruszewski, Theo Desbordes, Dieuwke Hupkes, Stanislas Dehaene, and Marco Baroni. 2019. The emergence of number and syntax units in LSTM language models. 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 11–20, Minneapolis, Minnesota. Association for Computational Linguistics.

Tal Linzen, Grzegorz Chrupała, Yonatan Belinkov, and Dieuwke Hupkes. 2019. Proceedings of the 2019 acl workshop blackboxnlp: Analyzing and interpreting neural networks for nlp. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

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.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019a. Linguistic knowledge and transferability of contextual 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 1073–1094.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

W. James Murdoch, Peter J. Liu, and Bin Yu. 2018. Beyond word importance: Contextual decomposition to extract interactions from lstms. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Open Source Initiative. 2020. The mit license.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca

Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. 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)*, pages 2227–2237.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. *arXiv preprint arXiv:2004.03061*.

Yada Pruksachatkun, Phil Yeres, Haokun Liu, Jason Phang, Phu Mon Htut, Alex Wang, Ian Tenney, and Samuel R Bowman. 2020. jiant: A software toolkit for research on general-purpose text understanding models. *arXiv preprint arXiv:2003.02249*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Lloyd S. Shapley. 1953. A value for n-person games. *Contributions to the Theory of Games*, (28):307–317.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 3319–3328.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.

Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and Ann Yuan. 2020. The language interpretability tool: Extensible, interactive visualizations and analysis for NLP models. *CoRR*, abs/2008.05122.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. *arXiv preprint arXiv:2003.12298*.

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, Sheng-Fu Wang, Jason Phang, Anhad Mohananey, Phu Mon Htut, Paloma Jeretic, and Samuel R. Bowman. 2019. Investigating BERT's knowledge of language: Five analysis methods with NPIs. 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)*, pages 2877–2887, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20.