A Toolkit for Scrutinizing Neural Network Activations

Abstract

This document introduces diagNNose, an open-source toolkit designed for the examination of activations within deep neural networks. diagNNose offers a diverse collection of interpretability methods, enabling a deeper understanding of the operational dynamics of neural networks. The utility of diagNNose is showcased through an investigation into subject-verb agreement in language models.

1 Introduction

We present diagNNose, a publicly available library for analyzing the behavior of deep neural networks. The diagNNose library equips researchers with tools to gain enhanced understanding of the internal representations formed by these networks, providing a comprehensive suite of established analysis methods. It accommodates a variety of model types, with a particular focus on NLP architectures, such as LSTMs and Transformers.

The availability of open-source libraries has been instrumental in the advancement and wider adoption of NLP technologies. We enhance the open-source ecosystem by integrating several interpretability techniques.

Recent years have witnessed significant interest in enhancing our understanding of the mechanisms by which deep neural networks function. The high-dimensional architecture of these models makes deciphering their internal dynamics a complex endeavor. This complexity has spurred the emergence of a specialized subfield within AI, dedicated to interpretability. diagNNose seeks to consolidate a range of these interpretability techniques into a unified library.

The primary objective of diagNNose is to facilitate the discovery of linguistic knowledge encoded within a model's representations. The library offers abstractions that enable the investigation of recurrent models in a manner similar to Transformer models, using a modular design. It includes a module for extracting model activations. The analysis methods currently implemented in the library include targeted syntactic evaluation tasks, probing with diagnostic classifiers, and feature attributions.

This paper provides a comprehensive overview of the library and illustrates its application in a case study centered on subject-verb agreement within language models. Subsequently, we provide a survey of diagNNose and elaborate on its specific modules. We conclude with the case study.

2 Background

The increasing capabilities of language models have resulted in a vibrant area of research focused on understanding their functionality. Approaches in this field are frequently interdisciplinary. diagNNose facilitates several influential analysis methods.

2.1 Targeted Syntactic Evaluations

Language models have been central to numerous achievements in NLP. These models are trained to predict the probability of upcoming or masked tokens. To achieve success in this task, models must grasp various linguistic aspects, including syntax, semantics, and general domain knowledge. One notable area of research investigating a model's linguistic competence employs targeted syntactic evaluations. This analysis method contrasts a model's outputs on minimally different pairs of grammatical and ungrammatical constructions. If a model assigns a higher probability to the grammatical construction, it suggests an understanding of the relevant linguistic principles.

diagNNose supports a diverse set of syntactic tasks and offers an interface for incorporating new tasks seamlessly.

2.2 Diagnostic Classifiers

Another line of research evaluates a model's comprehension of linguistic properties by training diagnostic classifiers on its representations. This technique, also known as probing, has yielded valuable insights into the internal mechanisms of language models. The activations used for training these classifiers are not limited to the hidden states of a language model at its top layer.

There have been recent discussions regarding the extent to which high accuracy in a diagnostic classifier truly signifies that a property is actively encoded by the model. Several methods have been put forward to address this, such as using control tasks or assessing classifiers based on minimum description length. diagNNose currently supports the training of diagnostic classifiers and control tasks.

2.3 Feature Attributions

While probing helps us identify specific properties embedded in model representations, it does not clarify how a model converts input features into accurate predictions. This can be addressed by calculating the contributions of input features to subsequent outputs. This is a complex task due to the high-dimensional, non-linear nature of deep learning models.

Feature attributions can be calculated in various manners. One common method involves a concept from cooperative game theory, referred to as the Shapley value. Computing Shapley values is computationally intensive, leading to the development of several approximation algorithms. diagNNose currently supports feature attribution computation using Contextual Decomposition and its generalization.

3 Library Overview

3.1 Modules

The library is organized into multiple modules that can be utilized as components for constructing an experimental pipeline.

3.1.1 Core Modules

The foundational modules underpinning the various pipelines that can be built using diagNNose are detailed below.

models: We offer a generalized framework for language models, enabling both recurrent and Transformer models to be accessed through a unified interface. Importing pre-trained Transformer models is accomplished using the transformers library. For recurrent models, we provide an interface that allows access to intermediate activations, including gate activations.

corpus: Corpora are imported as Datasets from the torchtext package. A Corpus can be converted into an iterator for processing. Tokenization can be performed traditionally, token-by-token, or based on subword units, such as byte pair encodings.

extract: The extraction of activations is fundamental to most analysis modules. We provide an Extractor class capable of extracting a model's activations given a corpus. This process is not restricted to the top layer; intermediate (gate) activations can also be extracted. Activations can be dynamically saved to disk to facilitate the extraction from large corpora with limited computational resources.

activations: Extracted activations can be readily accessed using an Activation-Reader, which provides access to activations corresponding to specific subsets of corpus sentences. We also offer functionality for extracting only particular subsets of activations, based on sentence and token information.

config: The pipeline of diagNNose is driven by configuration defined in JSON format. Individual attributes can also be directly set from the command line.

3.1.2 Analysis Modules

We presently offer three primary types of experimental modules.

syntax: The library offers capabilities for a broad range of targeted syntactic evaluation tasks.

probe: We furnish convenient tools for training diagnostic classifiers on extracted activations to probe for linguistic information that may be embedded within them. Our extraction module also enables training diagnostic classifiers on intermediate activations, including gate activations. To address concerns that high probing accuracy does not necessarily indicate that linguistic information is actively encoded, we have incorporated functionality for Control Tasks.

attribute: We offer capabilities for model-agnostic feature attributions, enabling the decomposition of a model's output into a sum of contributions. This is accomplished by implementing a wrapper over PyTorch operations, allowing intermediate feature contributions to be propagated during a forward pass. Our implementation supports various Shapley-based attribution methods and facilitates approximation procedures such as (Generalized) Contextual Decomposition and Shapley sampling values, in addition to the exact computation of propagated Shapley values.

3.2 Requirements

diagNNose can be installed using pip (pip install diagnnose) or cloned directly from the GitHub repository. The library is compatible with Python 3.6 or later, and its primary dependencies are PyTorch (v1.5+), torchtext, and HuggingFace's transformers. diagNNose is released under the MIT License. It operates on both CPUs and GPUs and has been optimized for smaller consumer setups.

The diagNNose codebase is fully typed using Python type hints and formatted using Black. All methods and classes are documented, with an overview available online.

4 Case Study: Subject-Verb Agreement

To exemplify the functionality of diagNNose, we examine subject-verb agreement corpora on a selection of language models. For our experiments, we analyze the following models: BERT, RoBERTa, DistilRoBERTa, and an LSTM language model.

4.1 Corpora

The corpora consist of seven tasks based on template-based syntactic constructions. These constructions feature an "agreement attractor" between the subject and the verb, which may mislead a language model into predicting the incorrect number of the verb. Consequently, a model must possess a robust understanding of sentence structure.

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 encompasses 600 to 900 distinct sentences. Sentences are categorized into multiple conditions based on the number of the subject and the intervening noun phrase.

To assess these corpora on a recurrent model, we initially compute the model's hidden state at the verb's position by feeding it the sub-sentence up to that point. Based on this hidden state, we compute and compare the output probabilities of the verb with the correct number (vc) and the incorrect number (vx):

$P(vc \mid he) > P(vx \mid he)$

For bi-directional masked language models, such as BERT, we cannot compute an intermediate hidden state by passing a sub-sentence because these models also incorporate input from future tokens. To address this, we substitute the verb in each sentence with a <mask> token and evaluate the model's probabilities at this token's position.

Many contemporary language models employ BPE tokenization, which may segment a word into multiple subwords. Therefore, in our experiments, we only compare verb forms where both the plural and singular forms are split into a single token.

4.2 Targeted Syntactic Evaluations

We execute the targeted syntactic evaluation suite on all seven templates. The results of this experiment are presented in Table 1.

Table 1: Results of the targeted syntactic evaluation tasks.

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

It is evident that Transformer language models generally attain higher scores compared to the LSTM model. Notably, the NAMEPP task presents a challenge for all models, with both RoBERTa and DistilRoBERTa scoring lower on this task than the LSTM. Another intriguing observation is the disparity in performance between RoBERTa and DistilRoBERTa on the NAMEPP and NOUNPP tasks. Despite DistilRoBERTa being trained to mimic RoBERTa's behavior, its performance on a downstream task like this differs considerably. These findings can serve as a foundation for more detailed analysis.

4.3 Feature Attributions

To gain a deeper understanding of why language models exhibit particularly poor performance on the NAMEPP corpus, we employ the feature attribution module on these constructions. The results for this experiment are presented below, illustrating the attributions for DistilRoBERTa on an example sentence from the corpus. This highlights the differential impact of the intervening attractor on the verb's number. The score at the top of the attribution represents the model's full logit for that class; these logits are transformed into probabilities using SoftMax. This logit is decomposed into a sum of contributions, indicated at the bottom of each token. It can be verified that the contributions sum to the logit, which is an important characteristic of feature attribution methods, ensuring a degree of faithfulness to the model. A negative value signifies a negative feature contribution to an output class: the influence of that feature diminished the preference for the class. Feature attributions also incorporate the influence of model biases.

In the provided example sentence, DistilRoBERTa produces an incorrect prediction: the logit of the incorrect singular form 'approves' is greater than that of the plural 'approve'. The model's error in predicting the correct verb form arises from the subject 'athletes' not providing sufficient contribution to outweigh the negative contributions from other input features. A model with a comprehensive grasp of subject-verb agreement should assign a larger contribution to the subject when predicting the main verb.

The attribute module is under active development. The exponential complexity of computing Shapley values makes generating these explanations a challenging task.

5 Conclusion

diagNNose offers crucial tools for interpretability research, providing advanced analysis techniques such as diagnostic classifiers and feature attributions. The library's modular architecture enables rapid testing of complex hypotheses and establishes a robust groundwork for the development of new interpretability methods. The library's code is open-source, and contributions are encouraged.