Explaining and Evaluating Deep Neural Networks in Natural Language Processing

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Department of Electrical and Computer Engineering

Kaiji Lu

Bachelor of Science in Electrical and Computer Engineering, Rice University
Bachelor of Arts in Cognitive Science, Rice University

Carnegie Mellon University
Pittsburgh, PA

May 2022

Acknowledgements

I always marvel at the elegance of human language for its expression of culture, knowledge and emotion, and I am fortunate to live in a time when we begin to teach machines to understand language, with the guidance of yet another marvel of mankind that is modern science and engineering. Carnegie Mellon has opened this new world to me, a world at times challenging and discouraging, but more often enlightening and fulfilling.

I have many people to thank during the course of my graduate studies. My advisor, Anupam Datta, has taught me how to do research creatively and methodically, how to think broadly and deeply, how to learn from difficulties and negative reviews, and how to communicate effectively to different audiences.

I am also fortunate to be under the guidance of my committee, Prof. Anupam Datta (Chair), Prof. Matt Fredrikson, Dr. Piotr Mardziel and Prof. Pradeep Ravikumar, who have offered me thoughtful feedbacks and enlightening comments during the defense process.

I would also like to extend my gratitude towards my collaborators Dr. Klas Leino, Zifan Wang, Fangjing Wu, Preetam Amancharla, Prof. Anupam Datta, Prof. Matt Fredrikson, Dr. Piotr Mardziel, and other members of the Accountable Systems Lab and CyLab community, in both Silicon Valley and Pittsburgh campus.

The works done in this thesis are developed with the support of NSF grant CNS-1704845 as well as by DARPA and the Air Force Research Laboratory under agreement number FA8750-15-2-0277. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright notation thereon. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of DARPA, the Air Force Research Laboratory, the National Science Foundation, or the U.S. Government. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan V and Titan Xp GPU used for the works in this thesis.

Abstract

Deep Neural Networks such as Recurrent Neural Networks and Transformer models are widely adopted for their superior performance in many natural language processing (NLP) applications, but they remain largely black-box which can lead to unwanted bias and unrobust behaviors. To discover, diagnose and evaluate these issues, it is essential to make NLP systems accountable, reliable and most importantly, explainable. Explainability tools and methods can be leveraged by NLP practitioners to better understand how model make certain predictions, if the models are conceptually sound, and if the predictions are justified.

In this dissertation, we propose new explainability frameworks for NLP models, and show how those frameworks can be used to understand and evaluate NLP models beyond their accuracy metrics. Inspired by prior explainability approaches, our methods are designed to be distinctively suitable for textual data and NLP model architectures.

Our proposed methods can help understand exactly how a linguistic concept is represented in a model by answering two conceptual soundness questions: (1) how does important linguistic information flow from input words to output predictions? (2) how is the relative order of words/phrases encoded? We experiment with both recurrent and Transformer architectures across a variety of NLP tasks and show how our methods precisely and faithfully explain and evaluate the inner workings and order-sensitivity of NLP models. Furthermore, we exemplify generated explanations in key NLP tasks depicting how various linguistic concepts are represented, whether those representations are compatible with the underlying grammatical and linguistic rules, and how deviation from those rules manifests into model weaknesses and conceptual unsoundness.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Motivation. Natural language processing (NLP) has been greatly spurred by deep learning models, from Recurrent Neural Network (RNN) models such as LSTM [Hochreiter and Schmidhuber, 1997], to more recent advancement of transformer-based models [Vaswani et al., 2017], such as BERT [Devlin et al., 2019] and GPT [Brown et al., 2020]. Recurrent networks, in contrast with other architectures such as convolutional neural networks, are designed to model structures of language for its capability to capture syntactic relations within language contexts, as well as to accommodate text inputs with variable lengths. Transformer models, proven to be superior than RNNs in many NLP tasks, have completely transformed the NLP landscape in recent years through their gigantic amount of training data and huge, parallelized architectures that learn contextualized representations.

Both RNNs and transformers, however, are largely inscrutable due to their complicated architectures and large number of parameters [Bender et al., 2021]. Despite theoretical intuitions that deep models create more complicated representations in higher dimensions, the exact inner workings among these representations remain a key roadblock to both models' explainability and transparency, the lack of which hinders these highly performing models to be applied in the real world.

Besides, deep NLP models have also proven to be conceptually unsound [Parkinson, 2011], i.e., they fail to learn the correct linguistic concepts. Despite their success in benchmark datasets in various NLP applications [Wang et al., 2018], they often perform poorly on challenge datasets that are specifically designed to stress test NLP models on the same applications [Marvin and Linzen, 2018, McCoy et al., 2019, Ribeiro

et al., 2020]. However, neither is the creation of those datasets guided by explanations, nor are the models' poor performances on them thoroughly understood, making it difficult to systematically and deductively improve the models or generate new challenge sets.

With the rise in *explainability* research in recent years, model agnostic attribution methods such as Shapley values [Shapley et al., 1953] or Integrated Gradients [Sundararajan et al., 2017] have been proposed to explain the behavior of deep models. Most of these approaches, however, focus on applications with tabular or image data, while few have been, or in fact, are capable of, effectively explaining textual data and NLP models, for the following reasons:

- Most compute input-level feature importance which are often insufficient in excavating the inner workings of NLP models;
- Many NLP model architectures contain structurally interpretable components such as LSTM gates and attention weights, which are often misconstrued to be inherently explainable;
- Understanding NLP models require understanding the role of word orders which current methods do not take into account.

Contributions. Addressing these limitations, this thesis shows that internal explanations of information flow and analysis of order sensitivity are essential for thoroughly understanding and evaluating NLP models. Inspired by existing frameworks of gradient-based or game-theoretic explanation methods, we propose new methods for evaluating NLP models for better explainability, transparency and conceptual soundness, while ensuring the generated explanations are efficient, accurate and faithful. We also demonstrate how each method diagnoses the root causes of conceptual unsoundness in NLP applications: on a higher level, deep models often fail to encode the correct concepts of the English language, instead they impose their information flow mechanism and order-insensitivity which are potentially conceptually unsound. In this thesis, we focus on surfacing and measuring these aspects of conceptual unsoundness leveraging explanations, and leave training robust and conceptually sound models to future work.

In Chapter 2 & 3, we present a model-independent internal explanation method, *influence patterns*, tailored to explain both RNN (*influence paths*) and transformer models. *Influence patterns*, through an underlying searching algorithm, efficiently abstract traditional gradient-based input attributions to those of

specific interpretable components of the computation graph that propagate targeted linguistic signals. We demonstrate that this method effectively abstracts models with respect to certain linguistic concepts.

In Chapter 4, we propose a new method to explain the word orders of sequential data: *Order-sensitive Shapley Values (OSV)*. We conduct an extensive empirical evaluation to validate the method and surface how well various deep NLP models learn word order.

Related Work. Game-theoretic values [Shapley et al., 1953] and gradient-based approaches [Sundararajan et al., 2017] are widely adopted to explain general data-driven systems [Datta et al., 2016, Lundberg and Lee, 2017, Covert et al., 2020], or applied specifically to text classification models [Chen and Jordan, 2020, Zhang et al., 2021, Chen et al., 2018]. However, the explanation power of these methods is limited to computing feature influences in the input-embedding space or treating text data as tabular data. In this thesis, we extend beyond these approaches by characterizing the internal information flow and order sensitivity of NLP models.

A line of related works analyze internal activations such as layer embeddings or self-attention weights [Clark et al., 2019, Vig and Belinkov, 2019, Lin et al., 2019, Ethayarajh and Jurafsky, 2021], which are found to correlate with various linguistic concepts. However, these forward methods for explaining NLP models have been controversial as they equate structural explainability with genuine functional explainability [Serrano and Smith, 2019, Kovaleva et al., 2019, Michel et al., 2019, Voita et al., 2019]. Our proposed methods, on the other hand, are based on axiomatic approaches complemented with rigorous evaluation and comparisons for faithfulness and accuracy.

To allow for broader and non-local testing of model robustness, challenge datasets [Linzen et al., 2016, Belinkov et al., 2017, Ribeiro et al., 2020, Wu et al., 2021, McCoy et al., 2019] are constructed to stress test NLP models, either on benchmarked applications or specific grammatical rules. These tests are becoming more crucial in designing and evaluating the conceptual soundness and robustness of NLP models beyond benchmark performances. In this thesis, we show that generated explanations can verify or extend these analysis into more credible insights on how and why NLP models fail to generalize to these datasets.

1.1 Influence Paths & Influence Patterns

Given any model represented by a computation graph, *influence path* defines influence for a single chain of internal components, while *influence patterns* a subgraph of the entire model. Both methods localize influential components within a model capturing (or hindering) a linguistic concept.

1.1.1 Influence Paths for Explaining SVA of LSTM models

In Chapter 2, we focus our analysis on subject-verb number agreement (SVA), a task widely studied to analyze and evaluate NLP language models for their capabilities of encoding linguistic rules. Previous work either study this phenomena through the lens of causal testing [Linzen et al., 2016] by evaluating models on sentences generated from a template, or quantifying the importance model components through ablation [Lakretz et al., 2019]. Building on the datasets and insights from prior works, *influence paths* [Lu et al., 2020b] enables deeper understanding of RNN models' representation of SVA concept by quantifying the effect of model inputs on model outputs that traverse a particular path through a model. Our approach refines the distributional influence framework of Leino et al. [2018] which instruments a neural model with input a distribution of interest, or *DoI* (e.g. datasets meant to exercise a concept such as subject-verb agreement (SVA) [Linzen et al., 2016]) and an output quantity of interest, or *QoI* (e.g. a measure of SVA) in order to compute a gradient-based measure of input influence. Specifically, we decompose the total influence to path-specific quantities localizing the implementation of the given concept to paths through a model.

We demonstrate that for LSTM models, a single path is responsible for most of the input-output effect defining SVA, despite a variety of surrounding and intervening contexts. We show that attractors (intervening nouns of opposite number to the subject) do not diminish the contribution of the primary subject-verb path, but rather contribute their own influence of the opposite direction along the equivalent primary attractor-verb path, which leads to incorrect number prediction if an attractor's contribution overcomes the subject's. We corroborate and elaborate on existing results localizing subject number to the same two neurons which, in our results, lie on the primary path. We further extend and generalize prior compression/ablation results with a new path-focused compression test which verifies our localization conclusions.

1.1.2 Influence Patterns for Explaining Contextualization of BERT

In Chapter 3, we expand influence paths with a more generalized explanation device: *influence patterns*. Instead of a single path, we generalize the abstraction to a set of nodes within the model. In addition, we propose *guided path refinement (GPR)*, a linear-time greedy algorithm to search for a pattern in a model with the largest influence. We empirically show that GPR identifies succinct model abstractions where linguistic concepts in BERT such as SVA are captured. We discover that significant portion of information flows in BERT travel through skip connections instead of attention heads, indicating that attention weights [Abnar and Zuidema, 2020] alone are not sufficient to characterize information flow. By visualizing the extracted patterns locally, we show how information flow of words interact inside the model and BERT may use grammatically incorrect cues to make predictions. Globally, we also discover that the consistency of influence patterns across instances of a task reflects BERT's performance for that task. Through ablation experiments, we find that influence patterns account for information flows much more accurately than prior attention-based and layer-based explanation methods [Abnar and Zuidema, 2020, Leino et al., 2018, Dhamdhere et al., 2018], respectively.

1.2 Order-sensitive Shapley Values

Previous works discover that NLP models are not sensitive to word orders in the same way as humans [Pham et al., 2021, Sinha et al., 2021a, Clouatre et al., 2021, Alleman et al., 2021, Sinha et al., 2021b]: their performance are often undeterred for completely reshuffled inputs. To understand the root causes of such order-insensitivity, prior approaches such as Shapley values are inadequate as they treat sequential data as tabular data without attributing to the order of features in a sequence.

In Chapter 4, we introduce a new explanation method for sequential data: Order-sensitive Shapley Values (*OSV*), along with a mechanism for separating order features and two modes of intervening on sequence orders, *absolute position* and *relative order*. We conduct an extensive empirical evaluation to validate the method and surface how well various deep NLP models learn word order. Using synthetic data, we first show that *OSV* is more faithful in explaining model behavior than gradient-based methods. Second, applying to the HANS dataset [McCoy et al., 2019], we discover that the BERT-based NLI model uses only the word

occurrences without word orders. Although simple data augmentation improves accuracy on HANS, *OSV* shows that the augmented model does not fundamentally improve the model's learning of order. Third, we discover that not all sentiment analysis models learn negation properly: some fail to capture the correct syntax of the negation construct. Finally, we show that pretrained language models such as BERT may rely on the absolute positions of subject words to learn long-range Subject-Verb Agreement. With each NLP task, we also demonstrate how the order insensitivity uncovered by *OSV* in various models can be leveraged to generate adversarial examples.

1.3 Chapter Contributions

The first two chapters of this dissertation appear in the proceedings of ACL [Lu et al., 2020a] and NeurIPS [Lu et al., 2021], and the final chapter of this thesis is in submission at the time of writing.

Chapter 2

Influence Paths for Explaining RNN Models

2.1 Introduction

Traditional rule-based NLP techniques can capture syntactic structures, while statistical NLP techniques, such as n-gram models, can heuristically integrate semantics of a natural language. Modern RNN-based models such as Long Short-Term Memory (LSTM) models are tasked with incorporating both semantic features from the statistical associations in their training corpus, and structural features generalized from the same.

Despite evidence that LSTMs can capture syntactic rules in artificial languages [Gers and Schmidhuber, 2001], it is unclear whether they are as capable in natural languages [Linzen et al., 2016, Lakretz et al., 2019] in the context of rules such as subject-verb number agreement, especially when not supervised for the particular feature. The incongruence derives from this central question: *does an LSTM language model's apparent performance in subject-verb number agreement derive from statistical heuristics (like n-gram models) or from generalized knowledge (like rule-based models)?*

Recent work has begun addressing this question [Linzen et al., 2016] in the context of *language models*: models tasked with modeling the likelihood of the next word following a sequence of words as expected in a natural language (see Figure 2.1, bottom). *Subject-verb number agreement* dictates that the verb associated with a given subject should match its number (e.g., in Figure 2.1, the verb "run" should match with the subject "boys"). Giulianelli et al. [2018] showed that the subject grammatical number is associated with various gates in an LSTM, and Lakretz et al. [2019] showed that ablation (disabling activation) of an LSTM

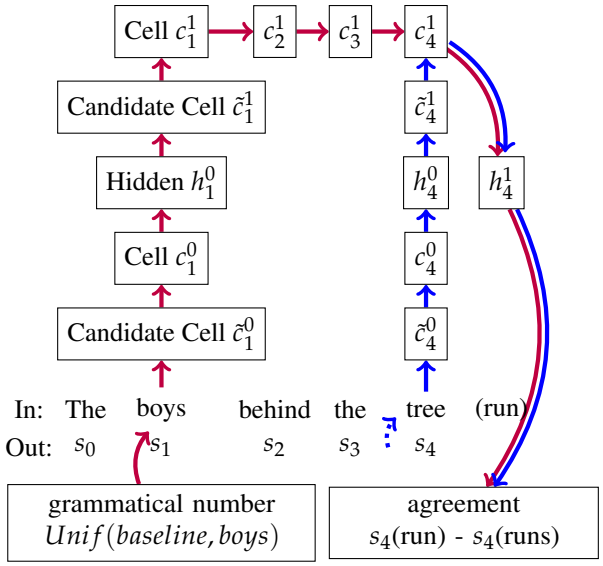


Figure 2.1: Subject-verb agreement task for a 2-layer LSTM language model, and primary paths across various LSTM gates implementing subject-verb number agreement. A language model assigns score s to each word. Agreement is the score of the correctly numbered verb minus that of the incorrectly numbered verb.

model at certain locations can reduce its accuracy at scoring verbs of the correct grammatical number.

Influence offers an alternate means of exploring properties like number agreement. We say an input is *influential* on an outcome when changing just the input and nothing else induces a change on the outcome. In English grammar, the number of a subject is influential on the number of its verb, in that changing the number of that subject while keeping all other elements of a sentence fixed would necessitate a change in the number of the verb. Algorithmic transparency literature offers formal definitions for empirically quantifying notions of influence for systems in general [Datta et al., 2016] and for deep neural networks specifically [Leino et al., 2018, Sundararajan et al., 2017].

The mere fact that subject number is influential on verb number as output by an LSTM model is sufficient to conclude that it incorporates the agreement concept in some way but does not indicate whether it operates as a statistical heuristic or as a generalized rule. We address this question with *influence paths*, which decompose influence into a set of paths across the gates and neurons of an LSTM model. The approach has several elements:

1. Define an input parameter to vary the concept-specific quantity under study (e.g., the grammatical number of a particular noun, bottom-left node in Figure 2.1) and a concept-specific output feature to measure the parameter's effect on (e.g, number agreement with the parameterized noun, bottom-right

node in Figure 2.1).

2. Apply a gradient-based influence method to quantify the influence of the concept parameter on the concept output feature; as per the chain rule, decompose the influence into model-path-specific quantities.

3. Inspect and characterize the distribution of influence across the model paths.

The paths demonstrate where relevant state information necessitated by the concept is kept, how it gets there, how it ends up being used to affect the model's output, and how and where related concepts interfere.

Our approach is state-agnostic in that it does not require *a priori* an assumption about how or if the concept will be implemented by the LSTM. This differs from works on diagnostic classifiers where a representation of the concept is assumed to exist in the network's latent space. The approach is also time-aware in that paths travel through cells/gates/neurons at different stages of an RNN evaluation. This differs from previous ablation-based techniques, which localize the number by clearing neurons at some position in an RNN for all time steps.

Our contributions are as follows:

- We introduce *influence paths*, a causal account of the use of concepts of interest as carried by paths across gates and neurons of an RNN.

- We demonstrate, using influence paths, that in a multi-layer LSTM language model, the concept of subject-verb number agreement is concentrated primarily on a single path (the red path in Figure 2.1), despite a variety of surrounding and intervening contexts.

- We show that attractors (intervening nouns of opposite number to the subject) do not diminish the contribution of the primary subject-verb path, but rather contribute their own influence of the opposite direction along the equivalent primary attractor-verb path (the blue path in the figure). This can lead to incorrect number prediction if an attractor's contribution overcomes the subject's.

- We corroborate and elaborate on existing results localizing subject number to the same two neurons which, in our results, lie on the primary path. We further extend and generalize prior compression/ablation results with a new path-focused compression test which verifies our localization conclusions.

Our results point to generalized knowledge as the answer to the central question. The number agreement concept is heavily centralized to the primary path despite the varieties of contexts. Further, the primary path's contribution is undiminished even amongst interfering contexts; number errors are not attributable to lack of the general number concept but rather to sufficiently influential contexts pushing the result in the opposite direction.

2.2 Background

LSTMs Long short-term memory networks (LSTMs) [Hochreiter and Schmidhuber, 1997] have proven to be effective for modeling sequences, such as language models, and empirically, this architecture has been found to be optimal compared to other second-order RNNs [Greff et al., 2017]. An LSTM cell can be defined as follows:

$$
\begin{aligned}
f_t &= \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \\
i_t &= \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right) \\
o_t &= \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right) \\
\tilde{c}_t &= \tanh\left(W_{\tilde{c}} x_t + U_{\tilde{c}} h_{t-1} + b_{\tilde{c}}\right) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
h_t &= o_t \odot \tanh\left(c_t\right)
\end{aligned}
$$

LSTMs utilize several types of gates and internal states including *forget gates* (f), *input gates* (i), *output gates* (o), *cell states* (c), *candidate cell state* ($\tilde{c}$), and *hidden states* (h). $\odot$ is element wise multiplication and σ is the sigmoid function. W, U and b are learnable parameters associated with each gate. At each time step, a cell state c_t and a hidden state h_t is computed recursively from the sigmoid gates and cell state/ hidden state of the previous time step. The network can be adapted to have multiple layers where the inputs x_t in higher layers are replaced with the hidden states h_t from the previous layer. Each gate is designed to carry out a certain function, or to fix a certain drawback of the vanilla RNN architecture. For example, the forget gate is supposed to determine how much information from the previous cell state to retain or "forget", helping to fix the vanishing gradient problem [Hochreiter, 1998].

Subject-Verb Number Agreement in Language Models The *subject-verb number agreement* (SVA) task, as described by Linzen et al. [2016], is an evaluation of a language model's ability to properly match the verb's grammatical number with its subject. This evaluation is performed on sentences specifically designed for the exercise, with zero or more words between the subject and the main verb, termed the *context*. The task for sentences with non-empty contexts will be referred to as *long-term* number agreement.

"Human-level" performance for this task can be achieved with a 2-layer LSTM language model [Gulordava et al., 2018], indicating that the language model incorporates grammatical number despite being trained only for the more general word prediction task. Attempts to explain or localize the number concept within the model include [Lakretz et al., 2019], where ablation of neurons is applied to locate specific neurons where such information is stored; and Giulianelli et al. [2018], Hupkes et al. [2018], where diagnostic classifiers are trained on gate activations to predict the number of the subject to see which gates or timesteps the number concept exhibits itself. These works also look at the special cases involving *attractors*—intervening nouns with grammatical number opposite to that of the subject (deemed instead *helpful nouns* if their number agrees with the subject)—such as the word "tree" in Figure 2.1. Both frameworks provide explanations as to why attractors lower the performance of SVA tasks. However, they tend to focus on the activation patterns of gates or neurons without justifying their casual relationships with the concept of grammatical number, and do not explicitly identify the exact temporal trajectory of how the number of the subject influences the number of the verb.

Other relevant studies that look inside RNN models to locate specific linguistic concepts include visualization techniques such as Karpathy et al. [2015], and explanations for supervised tasks involving LSTMs such as sentiment analysis [Murdoch et al., 2018].

Attribution Methods *Attribution methods* quantitatively measure the contribution of each of a function's individual inputs to its output. Gradient-based attribution methods compute the gradient of a model with respect to its inputs to describe how important each input is towards the output predictions. These methods have been applied to assist in explaining deep neural networks, predominantly in the image domain [Leino et al., 2018, Sundararajan et al., 2017, Bach et al., 2015, Simonyan et al., 2013]. Some such methods are also axiomatically justified to provide a causal link between inputs (or intermediate neurons) and the output.

As a starting point in this work, we consider *Integrated Gradients* (IG) [Sundararajan et al., 2017]. Given

a *baseline*, x_0, the attribution for each input at point, x, is the path integral taken from the baseline to x of the gradients of the model's output with respect to its inputs. The baseline establishes a neutral point from which to make a counterfactual comparison; the attribution of a feature can be interpreted as the share of the model's output that is due to that feature deviating from its baseline value. By integrating the gradients along the linear interpolation from the baseline to x, IG ensures that the attribution given to each feature is *sensitive* to effects exhibited by the gradient at any point between the baseline and instance x.

Leino et al. [2018] generalize IG to better focus attribution on concepts other than just model outputs, by use of a *quantity of interest* (QoI) and a *distribution of interest* (DoI). Their measure, *Distributional Influence*, is given by Definition 1. The QoI is a function of the model's output expressing a particular output behavior of the model to calculate influence for; in IG, this is fixed as the model's output. The DoI specifies a distribution over which the influence should faithfully summarize the model's behavior; the influences are found by taking an expected value over DoI.

Definition 1 (Distributional Influence)**.** *With* quantity of interest*, q, and* distribution of interest*, D, the influence, χ, of the inputs on the quantity of interest is:*

$$\chi(q, D) = \mathop{\mathbb{E}}_{\vec{x} \sim D} \left[\frac{\partial q}{\partial x}(\vec{x}) \right]$$

The directed path integral used by IG can be implemented by setting the DoI to a uniform distribution over the line from the baseline to $\vec{x}$: $D = \text{Uniform}\left(\overline{\vec{x}_0 \vec{x}}\right)$, for baseline, $\vec{x}_0$, and then multiplying χ by $\vec{x} - \vec{x}_0$. Conceptually, by multiplying by $\vec{x} - \vec{x}_0$, we are measuring the *attribution*, i.e., the contribution to the QoI, of $\vec{x} - \vec{x}_0$ by weighting its features by their *influence*. We use the framework of Leino et al. in this way to define our measure of attribution for SVA tasks in Section 2.3.

Distributional Influence can be approximated by sampling according to the DoI. In particular, when using $D = \text{Uniform}\left(\overline{\vec{x}_0 \vec{x}}\right)$ as noted above, Definition 1 can be computationally approximated with a sum of n intervals as in IG:

$$\chi \approx \sum_{i=1}^{n} \frac{\partial q}{\partial x} \left(\frac{i}{n} \vec{x} + \left(1 - \frac{i}{n}\right) \vec{x}_0 \right)$$

IG is an extension of Aumann Shapley values in the deep neural networks, which satisfies a lot of natural axioms: efficiency, dummy, path-symmetry, etc. [Sundararajan and Najmi, 2020]. Choices of DoIs, besides

the one used in IG, include Gaussian distributions with mean $\mathbf{x}$ [Smilkov et al., 2017] and Uniform distribution around $\mathbf{x}$ [Wang et al., 2020b].

Other related works include Fiacco et al. [2019], which employs the concept of neuron paths based on cofiring of neurons instead of influence, also on different NLP tasks from ours.

2.3 Methods

Our method for computing influence paths begins with modeling a relevant concept, such as grammatical number, in the influence framework of Leino et al. (Definition 1) by defining a quantity of interest that corresponds to the grammatical number of the verb, and defining a component of the input embedding that isolates the subject's grammatical number (Section 2.3.1). We then decompose the influence measure along the relevant structures of LSTM (gates or neurons) as per standard calculus identities to obtain a definition for *influence paths* (Section 2.3.2).

2.3.1 Measuring Number Agreement

For the SVA task, we view the initial fragment containing the subject as the input, and the word distribution at the position of its corresponding verb as the output.

Formally, each instance in this task is a sequence of d-dimensional word embedding vectors, $[\mathbf{w}] \stackrel{\text{def}}{=} \langle \vec{w}_i \rangle_i$, containing the subject and the corresponding verb, potentially with intervening words in between. We assume the subject is at position t and the verb at position $t+n$. The output score of a word, w, at position i will be written $s_i(w)$. If w has a grammatical number, we write w^+ and w^- to designate w with its original number and the equivalent word with the opposite number, respectively.

Quantity of Interest We instrument the output score with a QoI measuring the agreement of the output's grammatical number to that of the subject:

Definition 2 (Number Agreement Measure)**.** *Given a sentence,* $[\mathbf{w}]$*, with verb,* w*, whose correct form (w.r.t. grammatical number) is* w^+*, the* quantity of interest*,* q*, measures the correctness of the grammatical number of the verb:*

$$q\left([\mathbf{w}]\right) \stackrel{\text{def}}{=} s_{t+n}\left(w^+\right) - s_{t+n}\left(w^-\right)$$

In plain English, q captures the weight that the model assigns to the correct form of w as opposed to the weight it places on the incorrect form. Note that the number agreement concept could have reasonably been measured using a different quantity of interest. E.g., considering the scores of *all* vocabulary words of the correct number and incorrect number in the positive and negative terms, respectively, is an another alternative. However, based on our preliminary experiments, we found this alternative does not result in meaningful changes to the reported results in the further sections.

Distribution of Interest We also define a component of the embedding of the subject that captures its grammatical number, and a distribution over the inputs that allows us to sensitively measure the influence of this concept on our chosen quantity of interest. Let $\vec{w}^0$ be the word embedding midway between its numbered variants, i.e., $\frac{\vec{w}^+ + \vec{w}^-}{2}$. Though this vector will typically not correspond to any English word, we interpret it as a number-neutral version of $\vec{w}$. Various works show that linear arithmetic on word embeddings of this sort preserves meaningful word semantics as demonstrated in analogy parallelograms [Mikolov et al., 2013]. Finally, given a sentence, $[\mathbf{w}]$, let $[\mathbf{w}]_t^0$ be the sentence $[\mathbf{w}]$, except with the word embedding $\vec{w}_t$ replaced with its neutral form $\vec{w}_t^0$. We see that $[\mathbf{w}] - [\mathbf{w}]_t^0$ captures the part of the input corresponding to the grammatical number of the subject, $\vec{w}_t$.

Definition 3 (Grammatical Number Distribution)**.** *Given a singular (or plural) noun, w_t, in a sentence, $[\mathbf{w}]$, the distribution density of sentences, $D_{[\mathbf{w}]}$, exercising the noun's* singularity *(or* plurality*) linearly interpolates between the neutral sentence, $[\mathbf{w}]_t^0$, and the given sentence, $[\mathbf{w}]$:*

$$D_{[\mathbf{w}]} \stackrel{\text{def}}{=} \mathit{Uniform}\left(\overline{[\mathbf{w}]_t^0 [\mathbf{w}]}\right)$$

If $\vec{w}_t$ is singular, our counterfactual sentences span $[\mathbf{w}]$ with number-neutral $\vec{w}_t^0$ all the way to its singular form $\vec{w}_t = \vec{w}_t^+$. We thus call this distribution a *singularity* distribution. Were w_t plural instead, we would refer to the distribution as a *plurality* distribution. Using this distribution of sentences as our DoI thus allows us to measure the influence of $[\mathbf{w}] - [\mathbf{w}]_t^0$ (the grammatical number of a noun at position t) on our quantity of interest *sensitively* (in the sense that Sundararajan et al. define their axiom of sensitivity for IG [Sundararajan et al., 2017]).

Subject-Verb Number Agreement Putting things together, we define our attribution measure.

Definition 4 (Subject-Verb Number Agreement Attribution)**.** *The measure of attribution, α, of a noun's grammatical number on the subject-verb number agreement is defined in terms of the DoI, $D_{[\mathbf{w}]}$, and QoI, q, as in Definitions 3 and 2, respectively.*

$$\alpha\left([\mathbf{w}]\right) = \left([\mathbf{w}] - [\mathbf{w}]_t^0\right) \chi(q, D_{[\mathbf{w}]})$$

Essentially, the attribution measure weights the features of the subject's grammatical number by their Distributional Influence, χ. Because $D_{[\mathbf{w}]}$ is a uniform distribution over the line segment between $[\mathbf{w}]$ and $[\mathbf{w}]_t^0$, as with IG, the attribution can be interpreted as each feature's net contribution to the change in the QoI, $q([\mathbf{w}]) - q([\mathbf{w}]_t^0)$, as $\sum_i \chi([\mathbf{w}])_i = q([\mathbf{w}]) - q([\mathbf{w}]_t^0)$ (i.e., Definition 4 satisfies the axiom term *completeness* [Sundararajan et al., 2017]).

In Figure 2.1, for instance, this definition measures the attribution from the plurality of the subject ("boys"), towards the model's prediction of the correctly numbered verb ("run") versus the incorrectly numbered verb ("runs"). Later in this paper we will also investigate the attribution of intervening nouns on this same quantity. We expect the input attribution to be positive for all subjects and helpful nouns, and negative for attractors, which can be verified by the P^+columns of Table 2.1 (the details of this experiment are introduced in Section 2.4).

2.3.2 Influence Paths

Input attribution as defined by IG [Sundararajan et al., 2017] provides a way of explaining a model by highlighting the input dimensions with large attribution towards the output. Distributional Influence [Leino et al., 2018] with a carefully chosen QoI and DoI (Definition 4) further focuses the influence on a concept at hand, grammatical number agreement. Neither, however, demonstrate how these measures are conveyed by the inner workings of a model. In this section we define a decomposition of the influence into paths of a model, thereby assigning attribution not just to inputs, but also to the internal structures of a given model.

We first define arbitrary deep learning models as computational graphs, as in Definition 5. We then use this graph abstraction to define a notion of influence for a path through the graph. We posit that any natural path decomposition should satisfy the following conservation property: *the sum of the influence of each path from the input to the output should equal the influence of the input on the QoI*. We then observe that the chain rule from calculus offers one such natural decomposition, yielding Definition 6.

Definition 5 (Model)**.** *A model is an acyclic graph with a set of nodes, edges, and activation functions associated with each node. The output of a node, n, on input x is $n(x) \stackrel{\text{def}}{=} f_n\left(n_1(x), \cdots, n_m(x)\right)$ where $n_1, \cdots, n_m$ are n's predecessors and f_n is its activation function. If n does not have predecessors (it is an input), its activation is $f_n(x)$. We assume that the domains and ranges of all activation functions are real vectors of arbitrary dimension.*

We will write $n_1 \to n_2$ to denote an edge (i.e., n_1 is a direct predecessor of n_2), and $n_1 \mathrm{a}^* n_2$ to denote the set of all paths from n_1 to n_2. The partial derivative of the activation of n_2 with respect to the activation of n_1 will be written $\frac{\partial n_2}{\partial n_1}$.

This view of a computation model is an extension of network decompositions from attribution methods using the natural concept of "layers" or "slices" [Dhamdhere et al., 2018, Leino et al., 2018, Bach et al., 2015]. This decomposition can be tailored to the level of granularity we wish to expose. Moreover, in RNN models where no single and consistent "natural layer" can be found due to the variable-length inputs, a more general graph view provides the necessary versatility.

Definition 6 (Path Influence)**.** *Expanding Definition 4 using the chain rule, the influence of input node, s, on target node, t, in a model, G, is:*

$$\begin{aligned}\chi_s &= \mathop{\mathbb{E}}_{x \sim D(x)}\left[\frac{\partial t}{\partial s}(x)\right] \\ &= \mathop{\mathbb{E}}_{x \sim D(x)}\left[\sum_{p \in (s \to^* t)} \prod_{(n_1 \to n_2) \in p} \frac{\partial n_2}{\partial n_1}(x)\right]\end{aligned}$$

Note that the same LSTM can be modeled with different graphs to achieve a desired level of abstraction. We will use two particular levels of granularity: a coarse *gate-level* abstraction where nodes are LSTM gates, and a fine *neuron-level* abstraction where nodes are the vector elements of those gates. Though the choice of abstraction granularity has no effect on the represented model semantics, it has implications on graph paths and the scale of their individual contributions in a model.

Gate-level and Neuron-level Paths We define the set of gate-level nodes to include:

$$\left\{f_t^l,\ i_t^l,\ o_t^l,\ c_t^l,\ \tilde{c}_t^l,\ h_t^l\ :\ t < T,\ l < L\right\}$$

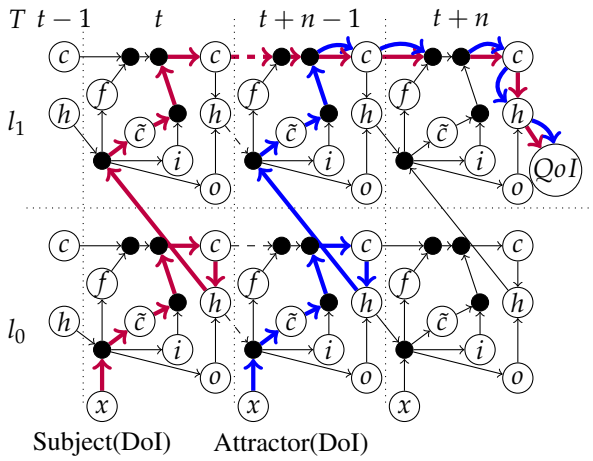


Figure 2.2: Influence path diagram in a SVA task for the 2-layer LSTM model. The red path shows the path with the greatest attribution (the primary path) from the subject; The blue path shows the primary path from the intervening noun (Attractor).

where T is the number of time steps (words) and L is number of LSTM layers. The node set also includes an attribution-specific input node ($[\mathbf{w}] - [\mathbf{w}]_t^0$) and an output node (the QoI). An example of this is illustrated in Figure 2.2. We exclude intermediate calculations (the solid nodes of Figure 2.2, such as $f_t \odot c_{t-1}$) as their inclusion does not change the set of paths in a graph. We can also break down each vector node into scalar components and further decompose the gate-level model into a neuron-level one: $\{f_{ti}^l,\ i_{ti}^l,\ o_{ti}^l,\ c_{ti}^l,\ \tilde{c}_{ti}^l,\ h_{ti}^l \ :\ t < T,\ i < H,\ l < L\}$, where H is the size of each gate vector. This decomposition results in an exponentially large number of paths. However, since many functions between gates in an LSTM are element-wise operations, neuron-level connections between many neighboring gates are sparse.

Path Refinement While the neuron-level path decomposition can theoretically be performed on the whole network, in practice we choose to specify a gate-level path first, then further decompose that path into neuron-level paths. We also collapse selected vector nodes, allowing us to further localize a concept on a neuron level while avoiding an explosion in the number of paths. The effect of this pipeline will be empirically justified in Section 2.4.

2.4 Evaluation

In this section we apply influence path decomposition to the SVA task. We investigate major gate-level paths and their influence concentrations in Section 2.4.2. We further show the relations between these paths and the paths carrying grammatical number from intervening nouns (i.e. attractors & helpful nouns) in Section 2.4.3. In both we also investigate high-attribution neurons along primary paths allowing us to compare our results to prior work.

2.4.1 Dataset and Model

We study the exact combination of language model and SVA datasets used in the closely related prior work of Lakretz et al. [2019]. The pre-trained language model of Gulordava et al. and Lakretz et al. is a 2-layer LSTM trained from Wikipedia articles. The number agreement datasets of Lakretz et al. are several synthetically generated datasets varying in syntactic structures and in the number of nouns between the subject and verb.

For example, *nounPP* refers to sentences containing a noun subject followed by a prepositional phrase such as in Figure 2.1. Each SVA task has subject number (and intervening noun number if present) realizations along singular (S) and plural (P) forms. In listings we denote subject number (S or P) first and additional noun (if any) number second. Details including the accuracy of the model on the SVA tasks are summarized by Lakretz et al. [2019]. Our evaluation replicates part of Table 2 in said work.

2.4.2 Decomposing Number Agreement

We begin with the attribution of subject number on its corresponding verb, as decomposed per Definition 6. Among all SVA tasks, the gate-level path carrying the most attribution is one following the same pattern with differences only in the size of contexts. With indices t and $t+n$ referring to the subject and verb respectively, this path, which we term the *primary path of subject-verb number agreement*, is as follows:

$$x_t(DoI) \cdot \tilde{c}^0 \cdot c^0 \cdot h^0 \cdot \tilde{c}^1 \cdot \left(c^1\right)^* \cdot h^1 \cdot QoI$$

The primary path is represented by the red path in Figure 2.2. The influence first passes through the temporary cell state $\tilde{c}^0$, the only non-sigmoid cell states capable of storing more information than sigmoid gates, since $i, f, o \in (0,1)$ while the *tanh* gate $\tilde{c} \in (-1,1)$. Then the path passes through c^0, h^0, and similarly to c^1

through $\tilde{c}^1$, jumping from the first to the second layer. The path then stays at c^1, through the direct connections between cell states of neighbouring time steps, as though it is "stored" there without any interference from subsequent words. As a result, this path is intuitively the most efficient and simplistic way for the model to encode and store a "number bit."

The extent to which this path can be viewed as *primary* is measured by two metrics. The results across a subset of syntactic structures and number conditions mirroring those in Lakretz et al. [2019] are shown in Table 2.1. We include 3 representative variations of the task. The metrics are:

1. t-value: probability that a given path has greater attribution than a uniformly sampled path on a uniformly sampled sentence.

2. Positive/Negative Share ($\pm$Share): expected (over sentences) fraction of total positive (or negative) attribution assigned to the given positive (or negative) path.

Per Table 2.1 (From Subject, Primary Path), we make our first main observation:

Observation 1. *The same one primary path consistently carries the largest amount positive attribution across all contexts as compared to all other paths.*

Even in the case of its smallest share (nounPPAdv), the 3% share is large when taking into account more than 40,000 paths in total. Sentences with singular subjects (top part of Table 2.1) have a slightly stronger concentration of attribution in the primary path than plural subjects (bottom part of Table 2.1), possibly due to English plural (infinitive) verb forms occurring more frequently than singular forms, thus less concentration of attribution is needed due to the "default signal" in place.

Primary Neurons We further decompose the primary path into influence passing through each neuron. Since only connections between second layer cell states are sparse, we only decompose the segment of the primary path from c_t^1 to c_{t+n}^1, resulting in a total of 650 (the number of hidden units) neuron-level paths. (We leave the non-sparse decompositions for future work). The path for neuron i, for example, is represented as:

$$x_t(DoI) \cdot \tilde{c}^0 \cdot c^0 \cdot h^0 \cdot \tilde{c}^1 \cdot \left(c_i^1\right)^* \cdot h^1 \cdot QoI$$

To compare the attribution of an individual neuron with all other neurons, we employ a similar aforementioned t-value, where each neuron-level path is compared against other neuron-level paths.

Task	C	From Subject						From Intervening Noun					
		P_+	$\|P\|$	Primary Path		Primary Neuron		P_+	$\|P\|$	Primary Path		Primary Neuron	
				+Share	t	t_{125}	t_{337}			$\pm$ Share	t	t_{125}	t_{337}
Simple	S	1.0	16	0.47	1.0	0.99	1.0	-	-	-	-	-	-
nounPP	SS	1.0	6946	0.1	1.0	1.0	1.0	0.82	16	0.31(+)	0.9	0.78	0.98
nounPP	SP	1.0	6946	0.1	1.0	1.0	1.0	0.23	16	0.24(-)	0.23	0.06	0.15
nounPPAdv	SS	1.0	41561	0.07	1.0	1.0	1.0	0.92	152	0.09(+)	0.96	0.85	1.0
nounPPAdv	SP	1.0	41561	0.07	1.0	1.0	1.0	0.32	152	0.09(-)	0.14	0.13	0.01
Simple	P	1.0	16	0.33	0.93	0.97	0.99	-	-	-	-	-	-
nounPP	PS	1.0	6946	0.05	0.91	0.99	1.0	0.06	16	0.28(-)	0.21	0.22	0.12
nounPP	PP	1.0	6946	0.05	0.92	0.99	1.0	0.95	16	0.31(+)	0.9	0.97	0.79
nounPPAdv	PS	1.0	41561	0.03	0.93	0.99	1.0	0.32	152	0.04(-)	0.28	0.41	0.16
nounPPAdv	PP	1.0	41561	0.03	0.92	0.99	1.0	0.83	152	0.07(+)	0.92	0.99	0.84

Table 2.1: Statistics for attribution of primary paths and neurons from the subject/intervening noun: P_+ is the percentage of sentences with positive input attribution. Task and C columns refer to sentence structures in Lakretz et al. [2019]. $|P|$ is the total number of paths; t and $\pm$Share are t-values and positive/negative share, respectively. For calculating t_{125} and t_{337} of primary neurons (125 and 337), we exclude these two neurons to avoid comparing them with each other.

The results of the neuron-level analysis are shown in Table 2.1 (From Subject, Primary Neuron). Out of the 650 neuron-level paths in the gate-level primary path, we discover two neurons with consistently the most attribution (neurons 125 and 337 of the second layer). This indicates the number concept is concentrated in only two neurons.

Comparison with Lakretz et al. [2019] Uncoincidentally, both neurons match the units found through ablation by Lakretz et al., who use the same model and dataset (neurons 988 and 776 are neurons 125 and 337 of the second layer). This accordance to some extent verifies that the neurons found through influence paths are functionally important. However, the t-values shown in Table 2.1 show that both neuron 125 and 337 are influential regardless of the subject number, whereas Lakretz et al. assign a subject number for each of these two neurons due to their disparate effect in lowering accuracy in ablation experiments. One possible reason is that the ablation mechanism used in Lakretz et al. [2019] assumes that a "neutral number state" can be represented by zero-activations for all gates, while in reality the network may encode the neutral state differently for different gates.

Another major distinction of our analysis from Lakretz et al. [2019] regards *simple* cases with no word between subjects and verbs. Unlike Lakretz et al., who claim that the two identified neurons are "long-term neurons", we discover that these two neurons are also the only neurons important for short-term number

agreement. This localization cannot be achieved by diagnostic classifiers used by Lakretz et al., indicating that the signal can be better uncovered using influence-based paths rather than association-based methods such as ablation.

2.4.3 Decomposing from Intervening Nouns

Next we focus on SVA tasks with intervening nouns and make the following observation:

Observation 2. *The primary subject-verb path still accounts for the largest positive attribution in contexts with either attractors or helpful nouns.*

A slightly worse SVA task performance [Lakretz et al., 2019] in cases of attractors (SP, PS) indicates that they interfere with prediction of the correct verb. In contrast, we also observe that helpful nouns (SS, PP) contribute positively to the correct verb number (although they should not from a grammar perspective).

Primary Path from the Intervening Noun We adapt our number agreement concept (Definition 2) by focusing the DoI on the intervening noun, thereby allowing us to decompose its influence on the verb number not grammatically associated with it. In Table 2.1 (From Intervening Noun) we discover a similar primary path from the intervening noun:

Observation 3. *Attribution towards verb number from intervening nouns follows the same primary path as the subject but is of lower magnitude and reflects either positive or negative attribution in cases of helpful nouns or attractors, respectively.*

This disparity in magnitude is expected since the language model possibly identifies the subject as the head noun through the prepositions such as "behind" in Figure 2.1, while still needing to track the number of the intervening noun in possible clausal structures. Such need is comparably weaker compared to tracking numbers of subjects, possibly because in English, intervening clauses are rarer than intervening non-clauses. Similar arguments can be made for neuron-level paths.

Task	C	Compression Scheme						
		$\overline{C_{si}}$	$\overline{C_s}$	$\overline{C_i}$	C_{si}	C_s	C_i	C
nounPP	SS	.66	.77	.95	.93	.71	.77	.95
nounPP	SP	.64	.36	.94	.64	.75	.40	.74
nounPP	PS	.34	.24	.92	.40	.69	.18	.80
nounPP	PP	.39	.66	.91	.76	.68	.58	.97
nounPP	mean	.51	.51	.93	.68	.70	.48	.87
nounPPAdv	SS	.70	.86	.98	.73	.56	.43	1.0
nounPPAdv	SP	.70	.43	.99	.50	.60	.27	.88
nounPPAdv	PS	.38	.22	.98	.76	.79	.56	.96
nounPPAdv	PP	.39	.67	.98	.84	.83	.76	1.0
nounPPAdv	mean	.54	.55	.99	.71	.69	.50	.96

Table 2.2: Model compression accuracy under various compression schemes. C is the uncompressed model.

2.4.4 Model Compression

Though the primary paths are the highest contributors to SVA tasks, it is possible that collections of associated non-primary paths account for more of the verb number concept. We gauge the extent to which the primary paths alone are responsible for the concept with compression/ablation experiments. We show that the computations relevant to a specific path alone are sufficient in maintaining performance for the SVA task. We *compress* the model by specifying node sets to preserve, and intervene on the activations of all other nodes by setting their activations to constant expected values (average over all samples). We choose the expected values instead of full ablation (setting them to zero), as ablation would nullify the function of Sigmoid gates. For example, to compress the model down to the red path in Figure 2.2, we only calculate the activation for gates $\tilde{c}_t^0$ and $\tilde{c}_t^1$ for each sample, while setting the activation of all other $\tilde{c}, f, o, i$ to their average values over all samples. In Table 2.2, we list variations of the compression schemes based on the following preserved node sets:

$$C \stackrel{\text{def}}{=} \left\{ f_t^l, i_t^l, o_t^l, \tilde{c}_t^l : t_{\text{sub}} < t < t_{\text{verb}}, l \in \{0,1\} \right\}$$

$$C_s \stackrel{\text{def}}{=} \left\{ \tilde{c}_{t_{\text{sub}}}^0, \tilde{c}_{t_{\text{sub}}}^1 \right\} \quad C_i \stackrel{\text{def}}{=} \left\{ \tilde{c}_{t_{\text{int}}}^0, \tilde{c}_{t_{\text{int}}}^1 \right\}$$

$$C_{si} \stackrel{\text{def}}{=} C_s \cup C_i$$

For example, column C_{si} in Table 2.2 shows the accuracy when the compressed model only retains the primary path from both the subject and the intervening noun while the computations of all other paths are set

to their expected values; while in $\overline{C_{si}}$, all paths but the paths in C_{si} are kept.

We observe that the best compressed model is $\overline{C_i}$, where the primary path from the intervening noun is left out; it performs even better than the original model; the increase comes from the cases with attractors (PS, SP). This indicates that eliminating the primary path from the attractor improves the model. The next best models apart from C are C_s and C_{si}, where primary paths are kept. Compressed models without the primary subject-verb path ($\overline{C_{si}}$, $\overline{C_s}$, C_i) have performances close to random guessing.

Observation 4. *Accuracy under path-based model compression tests corroborate that primary paths account for most of the subject number agreement concept of the LSTM.*

By comparing the SP and PS rows of $\overline{C_{si}}$, $\overline{C_s}$, C_s, and C_i, we observe the effect of attractors in misguiding the model into giving wrong predictions. Similarly, we see that helpful nouns (SS, PP) help guide the models to make more accurate predictions, though this is not grammatically justified.

2.5 Conclusions

The combination of finely-tuned attribution and gradient decomposition lets us investigate the handling of the grammatical number agreement concept attributed to paths across LSTM components. The concentration of attribution to a primary path and two primary cell state neurons and its persistence in a variety of short-term and long-term contexts, even with confounding attractors, demonstrates that the concept's handling is, to a large degree, general and localized. Though the heuristic decisioning aspect of an LSTM is present in the large quantities of paths with non-zero influence, their overall contribution to the concept is insignificant as compared to the primary path. Node-based compression results further corroborate these conclusions.

We note, however, that our results are based on datasets exercising the agreement concept in contexts of a limited size. We speculate that the primary path's attribution diminishes with the length of the context, which would suggest that at some context size, the handling of number will devolve to be mostly heuristic-like with no significant primary paths. Though our present datasets do not pose computational problems, the number of paths, at both the neuron and the gate level, is exponential with respect to context size. In the next Chapter, we explore new approaches for finding similar model abstractions in larger Transformer architectures with much better algorithmic scalability.

We also note that our method can be expanded to explore number agreement in more complicated sentences with clausal structures, or other syntactic/semantic signals such as coreference or gender agreement.

Chapter 3

Influence Patterns for Explaining Transformer Models

3.1 Introduction

Previous works show that transformer models such as BERT [Devlin et al., 2019] encode various linguistic concepts [Lin et al., 2019, Tenney et al., 2019a, Goldberg, 2019], some of which can be associated with internal components of each layer, such as internal embeddings or attention weights [Lin et al., 2019, Tenney et al., 2019a, Hewitt and Manning, 2019, Reif et al., 2019]. However, exactly how information flows through a transformer from input tokens to the output predictions remains an open question. Recent attempts to answer this question include using attention-based methods, where attention weights are used as indicators of flow of information [Clark et al., 2019, Abnar and Zuidema, 2020, Wu et al., 2020], or layer-based approaches [Hewitt and Manning, 2019, Hewitt and Liang, 2019, Reif et al., 2019], which identify important network units in each layer. In this chapter, we examine the information flow question through the lens of gradient-based attribution methods. In the previous chapter, we introduce *influence paths* which decompose the attribution to path-specific quantities localizing the implementation of the given concept to paths through a model. We also discover that a single path is responsible for most of the input-output effect defining SVA. Directly applying individual paths to transformer-based models like BERT, however, results in an intractable number of paths to enumerate due to the huge number of computation edges in BERT.

In this chapter, we introduce *influence patterns*, abstractions of sets of gradient-based paths through a transformer's entire computational graph. We also introduce a greedy search procedure for efficiently and effectively finding patterns representative of concept-critical information flow. Figure 3.1 provides an example of an influence pattern in BERT. We conduct an extensive empirical study of influence patterns for several NLP tasks: Subject-Verb Agreement (SVA), Reflexive Anaphora (RA), and Sentiment Analysis (SA). Our findings are summarized below.

- A significant portion of information flows in BERT go through skip connections and not attention heads, indicating that attention weights [Abnar and Zuidema, 2020] alone are not sufficient to characterize information flow. In our experiment, we show that on average, important information flow through skip connections approximately three times more often than attentions.
- By visualizing the extracted patterns, we show how information flow of words interact inside the model and BERT may use grammatically incorrect cues to make predictions.
- The consistency of influence patterns across instances of a task reflects BERT's performance for that task.
- Through ablation experiments, we find that influence patterns account for information flows in BERT on average 74% and 25% more accurately than prior attention-based and layer-based explanation methods [Abnar and Zuidema, 2020, Leino et al., 2018, Dhamdhere et al., 2018], respectively.

3.2 Background

We begin this section by introducing notations and architecture of transformer models in Section. 3.2.1. We then introduce two ways of describing the computational graph of Transformer models in Section. 3.2.2.

3.2.1 Transformer Models

Throughout the paper we use $\mathbf{x}$ to denote a vector and $|S|$ to denote the cardinality of a set S or number of nodes in a graph S. We begin with the basics of the Transformer encoder architecture [Vaswani et al., 2017] such as BERT [Devlin et al., 2019] (presented in Fig. 3.1). Let L be the number of layers in a transformer

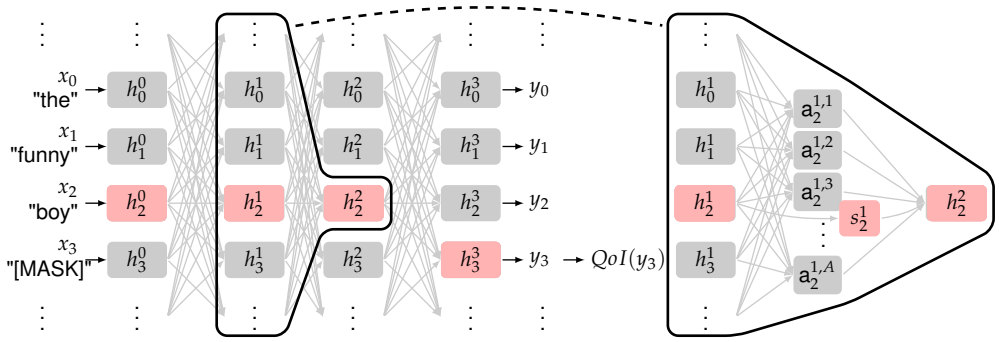


Figure 3.1: BERT architecture (left) and details of a transformer layer (right) for an instance of the SVA task, which evaluates whether the model chooses the correct verb form `is` over `are` for `[MASK]` to agree with the subject. An example of a pattern is highlighted with red nodes.

encoder, H the hidden dimension of embeddings at each layer, and A the number of attention heads. The list of input word embeddings is $\mathbf{x} \stackrel{\text{def}}{=} [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N], \mathbf{x}_i \in \mathbb{R}^d$. We denote the output of the l-th layer as $\mathbf{h}^l_{1:N}$. First layer inputs are $\mathbf{h}^0_{1:N} \stackrel{\text{def}}{=} \mathbf{x}_{1:N}$.We use $a^{l,i}_j$ to denote the j-th attention head from the i-th embedding at l-th layer and s^l_j to denote the skip connection that is "copied" from the input embedding from the previous layer then combined with the attention output. The output logits are denoted by $\mathbf{y}$.

3.2.2 Computation Graph of Transformer models

A deep network can be viewed as a computational graph $\mathcal{G} \stackrel{\text{def}}{=} (\mathcal{V}, \mathcal{F}, \mathcal{E})$, a set of nodes, activation functions, and edges, respectively. In this paper, we assume the graph is directed, acyclic, and does not contain more than one edge per adjacent pair of nodes. A path p in $\mathcal{G}$ is a sequence of graph-adjacent nodes $[p_1, p_2, \cdots, p_t]$; p_t is the output node. Thus, the Jacobian passing through a path p evaluated at input $\mathbf{x}$ is $\prod_{i=1}^{-1} \partial p_i(\mathbf{x}) / \partial p_{i-1}(\mathbf{x})$ as per chain rule. We further denote the Jacobian of the output of node n_i w.r.t the output of connected (not necessarily directly) predecessor node n_j evaluated at $\mathbf{x}$ as $\partial n_j(\mathbf{x}) / \partial n_i(\mathbf{x})$.

For computational and interpretability reasons, an ideal graph would contain as few nodes and edges as possible while exposing the structures of interest. For transformer models we propose two graphs: *embedding-level graph* $\mathcal{G}_e$ corresponding to the nodes and edges shown in Figure 3.1 (left) to explain how the influence of input embeddings flow from one layer of internal representations to another and to the eventual prediction; and *attention-level graph* $\mathcal{G}_a \supset \mathcal{G}_e$ that additionally includes attention head nodes and skip connection nodes as in Figure 3.1 (right), a finer decomposition to demonstrate how influence from the input embedding flows

through the attention block (or skip connections) within each layer.

3.3 Tracing Influence Flow with Patterns

To explain how different concepts in the input flow to final predictions in BERT, it is important to show how the information from each input word flows through each intermediate layer and finally reaches the output embedding of interest, e.g. `[MASK]` for pretraining or `[CLS]` for fine-tuned models. Prior approaches have use the attention weights to build directed graphs from one embedding to another [Abnar and Zuidema, 2020, Wu et al., 2020]. These approaches use heuristics to treat high attention weights as indicators of important information flow between layers. However, as more work starts to highlight the axiomatic justifications of gradient-based methods over attentions weights as an explanation approach [Treviso and Martins, 2020], we therefore explore an orthogonal direction in applying distributional influence in BERT to trace the information flow. We use the similar definition of distribution influence introduced in Section. 2.2 of the previous chapter. Specifically, in a MLM setting such as Figure 3.1, we can define the quantity of interest $q(f(x)) = f(x)_{\texttt{IS}} - f(x)_{\texttt{ARE}}$ where $f(x)_*$ is the logit output of class $*$ at the position of `[MASK]`.

Tracing Influence by Patterns By viewing BERT as a computation graph ($\mathcal{G}_e$ or $\mathcal{G}_a$ in 3.2.1), we restate the problem: given a source node s and a target node t, we seek a significant pattern of nodes from s to t that shows how the influence from s traverses from node to node and finally reaches t. An exhaust way to rank all paths by the amount of influence flowing from s to t is possible in smaller networks, as is introduced in Section. 2.3 of the previous chapter. However, the similar approach lacks scalability to large models like BERT. Therefore, we propose a way to greedily narrow down the searching space from all possible paths to specific *patterns*. That is, our approach is two-fold: 1) we employ abstractions of sets of paths as the localization and influence quantification instrument; 2) we discover influential patterns with a greedy search procedure that refines abstract patterns into more concrete ones, keeping the influence high. We begin with the formal definition of a *pattern*.

Definition 7 (Pattern)**.** *A pattern π is a sequence of nodes $[\pi_1, \pi_2, \cdots, \pi_{-1}]$ such that for any pair of nodes π_i, π_{i+1} adjacent in the sequence (not necessarily adjacent in the graph), there exists a path from π_i and π_{i+1}.*

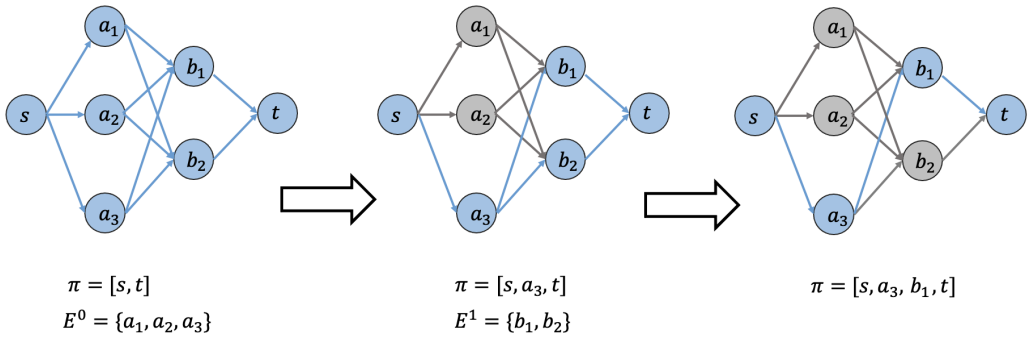


Figure 3.2: A visual illustration of Guided Pattern Refinement (GPR) in a toy example. We start with a pattern $\pi = [s, t]$ containing only the source and the target node. At each step we define a guided set E^0 and E^1, respectively and find the node in the guided set that maximize the pattern influence (Def. 8). GPR finally returns a pattern $\pi = [s, a_3, b_1, t]$ that abstracts a single path.

A pattern π abstracts a set of paths, written $\gamma(\pi)$ that follow the given sequence of nodes but are free to traverse the graph between those nodes in any way. Interpreting paths and patterns as sets, we define $\gamma(\pi) \stackrel{\text{def}}{=} \{p \subseteq \mathcal{P} : \pi \subseteq p\}$ where $\mathcal{P}$ is the set of all paths from π_1 to π_{-1}. If every sequence-adjacent pair of nodes is directly connected then the pattern abstracts a single path. To quantify the amount of information that flows from the input node to the target node over a particular pattern, we propose *Pattern Influence*, motivated by distributional influence.

Definition 8 (Pattern influence)**.** *Given a computation graph and a user-defined distribution $\mathcal{D}$, the* influence *of an influence pattern π, written $\mathcal{I}(\mathbf{x}, \pi)$ is the total influence of all the paths abstracted by the pattern:* $\mathcal{I}(\mathbf{x}, \pi) \stackrel{\text{def}}{=} \sum_{p \in \gamma(\pi)} \mathbb{E}_{\mathbf{z} \sim \mathcal{D}(\mathbf{x})} \prod_{i=1}^{-1} \frac{\partial p_i(\mathbf{z})}{\partial p_{i-1}(\mathbf{z})}$.

Proposition 1 (Chain Rule)**.** $\mathcal{I}(\mathbf{x}, \pi) = \mathbb{E}_{\mathbf{z} \sim \mathcal{D}(\mathbf{x})} \prod_{i=1}^{-1} \frac{\partial \pi_i(\mathbf{z})}{\partial \pi_{i-1}(\mathbf{z})}$ *for any distribution* $\mathcal{D}(\mathbf{x})$.

Prop. 1 (proof in Appendix 6.1) simplifies the evaluation of the pattern influence by specifying the exact set of internal nodes through which influence flows in a computational graph. We hereby introduce a greedy way of finding the most influential pattern in both $\mathcal{G}_e$ and $\mathcal{G}_a$.

Guided Pattern Refinement(GPR) Starting with source and target nodes s and t along with a initialized pattern $\pi^0 = \{s, t\}$ representing all paths between s and t, we construct π^1 by adding one[1] sequence-adjacent node e^0 (there is a direct path between s and e^0) from a *guiding set* E^0 that maximizes the influence of the

[1]The algorithm can be easily adapted to include more nodes per layer. However, we found one node per layer retain a reasonable proportion of influence for the tasks evaluated in this paper (See Sec.3.5.4).

Algorithm 1: Guided Pattern Refinement in the Embedding-level Graph (GPR-e)

Result: Significant Path π^e
initialization;
$\mathbf{x} \sim$ Input Tokens, $f \leftarrow$ BERT ;
$\mathcal{G}_e \leftarrow$ GetEmbeddingGraph(f), $L \leftarrow$ GetNumberOfLayers(f);
$N \leftarrow$ GetNumberOfTokens(f), $m \leftarrow$ GetIndex(`[MASK]`), ;
$n_q \leftarrow$ GetQoINode$(m, \mathcal{G}_e)$, $n_w \leftarrow$ GetClfNode$(\mathcal{G}_e)$;
$\pi^e \leftarrow$ OrderedSet(), $\mathcal{C} \leftarrow \{\}$;
$j \leftarrow$ GetStartingIndex(); `// The word we start the search with`
$n_j^0 \leftarrow$ GetNode$(\mathcal{G}_w, \mathbf{h}_j^0)$; `// Find the corresponding node`
$\pi^e \leftarrow$ Append(π^e, n_j^0);
for $l \in \{1, ..., L-2\}$ **do**
  $\mathcal{C} \leftarrow \{\}$;
  for $i \in \{0, ..., N-1\}$ **do**
    $n_i^l \leftarrow$ GetNode$(\mathcal{G}_w, \mathbf{h}_i^l)$;
    $\pi_t \leftarrow$ Append(π, n_i^l);
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{\pi_t\}$;
  end
  $\pi^e \leftarrow \arg\max_{\pi' \in \mathcal{C}} \mathcal{I}(\mathbf{x} | \pi', \pi'_{-1} \rightarrow^{\mathcal{P}} n_q)$
end
$n_m^L \leftarrow$ GetNode$(\mathcal{G}_w, \mathbf{h}_m^{L-1})$;
$\pi^e \leftarrow$ Append(π^e, n_m^L);
$\pi^e \leftarrow$ Append(π^e, n_w);
$\pi^e \leftarrow$ Append(π^e, n_q);

resulting pattern such that:

$$e^0 = \arg\max_{e \in E^0} \{\mathcal{I}(\mathbf{x}, [s, e, t]), \pi^1 = [s, e^0, t]\,, \tag{3.1}$$

This procedure is iterated until we exhaust the last guiding set. We show an example of GPR in a toy graph in Fig. 3.2. For an embedding-level graph $\mathcal{G}_e$, each guiding set E^l includes all embeddings $\mathbf{h}_{1:N}^l$ at layer l. For the attention-level graph $\mathcal{G}_a$, we refine on the embedding-level pattern π^e by only expanding π^e from addition nodes in $\mathcal{G}_a$ compared with $\mathcal{G}_e$: we perform GPR iterations between $\pi_l^e, \pi_{l+1}^e \in \pi^e$ with the guiding set E_a^l which includes A attention heads $\mathbf{a}^l$ and the skip node $\mathbf{s}^l$(Fig. 3.1 Right), until we reach the same last node in π^e. The returned attention-level pattern π^a thus abstracts a single path from the source to the target in $\mathcal{G}_a$. As the attention-level analysis refines the embedding-level analysis, the produced attention-level pattern π^a abstracts a strict subset of the paths of the attention-level graph that the embedding-level pattern π^e abstracts. That is, $\pi^e \subset \pi^a$ while $\gamma(\pi^a) \subset \gamma(\pi^e)$. The detailed algorithm for computing π^e and π^a is

Algorithm 2: Guided Pattern Refinement in the Attention-level Graph (GPR-a)

Result: Significant Path π^a
initialization;
$\mathbf{x} \sim$ Input Tokens, $f \leftarrow$ BERT ;
$\mathcal{G}_e \leftarrow \text{GetEmbeddingGraph}(f)$, $\mathcal{G}_a \leftarrow \text{GetAttentionGraph}(f)$;
$m \leftarrow \text{GetIndex}(\texttt{[MASK]})$;
$n_q \leftarrow \text{GetQoINode}(m, \mathcal{G}_e)$;
$L \leftarrow \text{GetNumberOfLayers}(f)$, $N \leftarrow \text{GetNumberOfTokens(f)}$;
$\pi^e \leftarrow \text{GPR-e}(\mathbf{x}, \mathcal{G}_e)$; `// Find embedding-level path first`
$\mathcal{C} \leftarrow \{\}$;
for $i \in \{0, ..., |\pi^e| - 2\}$ **do**
  if *ExistAttentionBlock*(π^e_i, π^e_{i+1}); `// Check if this is a Transformer Layer`
  then
    $\mathcal{A}_i \leftarrow \text{GetHeadsBetween}(\mathcal{G}_a, \pi^e_i, \pi^e_{i+1})$; `// Get all attention heads and the skip connetion node`
    $\pi^e_{head} \leftarrow \text{Slice}(\pi^e_0, \pi^e_i)$; `// Take a slice between two nodes`
    $(a^*_i, c^*_i) \leftarrow \arg\max_{(a_i, c_i) \in \mathcal{A}_i} \mathcal{I}(\mathbf{x} | \pi^e_{head} \cup \{a_i, c_i\}, c_i \rightarrow^{\mathcal{P}} n_q)$;
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{(a^*_i, c^*_i)\}$;
  else
    continue;
  end
end
$\pi^a \leftarrow \text{InsertNode}(\pi^e, \mathcal{C})$; `// Insert attention nodes into the embedding-level path at the corresponding place`

included in Algorithm 1 and 2, respectively. Although GPR is not guaranteed to be optimal, we included a detailed analysis of its optimality in Appendix 6.2.1.

3.4 Baseline Patterns

In this section, we introduce baseline patterns for comparison. We include those generated by attention weights [Abnar and Zuidema, 2020, Wu et al., 2020], and layer-based gradient approaches [Leino et al., 2018, Dhamdhere et al., 2018]. Both groups of approaches compute either input influences or influences to internal neurons without an explicit way to quantify an end-to-end influential pattern. However, we define natural augmentations of each baseline to compute the pattern capturing the most influence, layer by layer, while aligning the total number of extracted nodes with that of an influence pattern.

3.4.1 Baseline: Attention-based patterns

We introduce the implementation details of attention-based patterns, inspired by Abnar and Zuidema [2020] and Wu et al. [2020]. Consider a BERT model with L layers, between adjacent layers l and $l+1$, the attention matrix is denoted by $M_l \in \mathbb{R}^{A\times N\times N}$ where A is the number of attention heads and K is the number of embeddings. Each element of $M_l[a,i,j]$ is the attentions scores between the i-th embedding at layer l and the j-th embedding at layer $l+1$ of the a-th head such that $\sum_i M_l[a,i,j] = 1$. We average the attentions scores over all heads to lower the dimension of M: $\tilde{M}_l \stackrel{\text{def}}{=} \frac{1}{A}\sum_a M_l[a]$. We define the baseline attention path as one where the product of each edge in this path is the maximum possible score among all paths from a given source to the target.

Definition 9 (Attention-based Pattern)**.** *Given a set of attention matrices $\tilde{M}_0, \tilde{M}_1, ..., \tilde{M}_{L-1}$, a source embedding x and the quantity of interest node q, an attention-based pattern Π_{attn} is defined as*

$$\Pi_{attn} \stackrel{\text{def}}{=} \{x, h_*^1, h_*^2, ..., h_*^{L-2}, q\}$$

where

$$h_*^1, h_*^2, ..., h_*^{L-2} = \arg\max_{j_1, j_2, ..., j_{L-2}} P(h_{j_1}^1, h_{j_2}^2, ..., h_{j_{L-2}}^{L-2})$$
$$P = \tilde{M}_0[s, j_1]\tilde{M}_{L-1}[j_{L-2}, t]\prod_{l=1}^{L-2}\tilde{M}_0[j_l, j_{l+1}]$$

Abnar and Zuidema [2020] considers an alternative choice $\hat{M}_l \stackrel{\text{def}}{=} 0.5I + 0.5\tilde{M}_l$ to account for the skip connection in the attention block where I is an identity matrix to represent the identity transformation in the skip connection, which we use for GPR in Sec. 3.5, we use $\hat{M}_l$ to replace $\tilde{M}_l$ when returning the attention-based pattern. Dynamic programming can be applied to find the maximum of the product of attentions scores and back-trace the optimal nodes at each layer to return $h_*^1, h_*^2, ..., h_*^{L-2}$.

3.4.2 Baseline: Conductance and Internal Influence

We find patterns consisting of nodes that maximizes *condutance* [Dhamdhere et al., 2018] and *internal influence* [Leino et al., 2018] to build baseline methods Π_{cond} and Π_{inf}, respectively. These two measurements are defined follows:

Definition 10 (Conductance [Dhamdhere et al., 2018]). *Given a model $f : \mathbb{R}^d \to \mathbb{R}^n$, an input* $\mathbf{x}$*, a baseline input* $\mathbf{x}_b$ *and a QoI* q*, the conductance on the output* $\mathbf{h}$ *of a hidden neuron is defined as*

$$c_q(\mathbf{x}, \mathbf{x}_b, \mathbf{h}) = (\mathbf{x} - \mathbf{x}_b) \circ \sum_i \mathbf{1}_i \circ \mathbb{E}_{\mathbf{z}\sim\mathcal{U}}[\frac{\partial q(f(\mathbf{z}))}{\partial \mathbf{h}(\mathbf{z})} \frac{\partial \mathbf{h}(\mathbf{z})}{\partial z_i}] \tag{3.2}$$

where $\mathbf{1}_i$ *is a vector of the same shape with* $\mathbf{x}$ *but all elements are 0 except the* i*-th element is filled with 1.* $\mathcal{U} := Uniform(\overline{\mathbf{x}_b\mathbf{x}})$.

Definition 11 (Internal Influence [Leino et al., 2018]). *Given a model $f : \mathbb{R}^d \to \mathbb{R}^n$, an input* $\mathbf{x}$*, a QoI* q *and a distribution of interest* $\mathcal{D}$*, the internal influence on the output* $\mathbf{h}$ *of a hidden neuron is defined as*

$$\chi_q(\mathbf{x}, \mathbf{h}) = \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[\frac{\partial q(f(\mathbf{z}))}{\partial \mathbf{h}(\mathbf{z})}] \tag{3.3}$$

To build a pattern using the layer-based methods, we select the most influential node in each layer and concatenate them to form the most influential pattern Π_{cond} and Π_{inf}, respectively.

3.5 Evaluation

Experiments in this section demonstrate our method as a tool for interpreting end-to-end information flow in BERT. Specific visualizations exemplify these interpretations including the importance of skip connections and BERT's encoding of grammatical concepts are included in Section. 3.5.2. Section. 3.5.3 explores the consistency of patterns across instances and template positions and how they relate to task performances and influence magnitudes in. Sec 3.5.4 demonstrates two advantages of patterns in explaining the information flow of BERT over baselines: 1) abstracted patterns, with much fewer nodes compared to the whole model, carry sufficient information for the prediction. That is, without the information outside the refined pattern, the model shows no significant performance drop; 2) At the same time, patterns are sparse but concentrated in BERT's components. The code for the experiments conducted in this section, including the core GPR algorithm, is publicly available at `https://github.com/caleblu/influence-patterns`.

3.5.1 Setup

Tasks. We consider two groups of NLP tasks: (1) *subject-word agreement (SVA)* and *reflexive anaphora (RA)*. We explore different forms of sentence stimuli (subtask) within each task: object relative clause

(Obj.), subject relative clause (Subj.), within sentence complement (WSC), and across prepositional phrase (APP) in SVA [Marvin and Linzen, 2018]; number agreement (NA) and gender agreement (GA) in RA [Lin et al., 2019]. Both datasets are evaluated using masked language model (MLM) as is used in Goldberg [2019]. We sample 1000 sentences from each subtask evenly distributed across different sentence types (e.g. singular/plural subject & singular/plural intervening noun) with a fixed sentence structure; (2) *sentiment analysis(SA)*: we use 220 short examples (sentence length$\leq$ 17) from the evaluation set of the 2-class GLUE SST-2 sentiment analysis dataset [Wang et al., 2018]. For linguistics tasks, example sentences and accuracy can be found in Table 3.1. First five of the linguistic tasks are sampled from Marvin and Linzen [2018], the last task is sampled from dataset in Lin et al. [2019], all datasets are constructed as an MLM task according to Goldberg [2019]). In order to compute quantitative results, we sample sentences with a fixed length from each subtask.

Models. For linguistic tasks, We evaluate our methods with a pretrained BERT($L = 6, A = 8$) [Turc et al., 2019], referred hereby as $\text{BERT}_{\text{SMALL}}$. In Table 3.1, we include the accuracy of a larger BERT model ($\text{BERT}_{\text{BASE}}$) which are comparable in performance with the smaller $\text{BERT}_{\text{SMALL}}$. For SST-2 we fine-tuned on the pretrained $\text{BERT}_{\text{BASE}}$ [Devlin et al., 2019] with $L = 12, A = 12$. The models are comparable in sizes compared to the transformer models used in Abnar and Zuidema [2020].

Implementation of GPR. Let the target node for SVA and RA tasks be the output of the QoI score $q(\mathbf{y}) \stackrel{\text{def}}{=} y_{correct} - y_{wrong}$ (similar to the QoI used in Def. 2 of Chapter 2). For instance, $y_{\texttt{IS}} - y_{\texttt{ARE}}$ for the sentence `she [MASK] happy`. Similarly, we use $y_{\texttt{positive}} - y_{\texttt{negative}}$ for sentiment analysis. We choose an uniform distribution over a linear path from $\mathbf{x}_b$ to $\mathbf{x}$ as the distribution $\mathcal{D}$ in Def. 8 where the $\mathbf{x}_b$ is chosen as the the input embedding of `[MASK]` because it can viewed a word with no information, similar to the QoI used in Def. 3 of Chapter 2[2]. For a given input token $\mathbf{x}_i$, we apply GPR differently depending on the sign of distributional influence $g(\mathbf{x}; q, \mathcal{D})$: if $g(\mathbf{x}; q, \mathcal{D}) \geq 0$, we maximize the pattern influence towards $q(\mathbf{y})$ at each iteration of the GPR otherwise we maximize pattern influence towards $-q(\mathbf{y})$. We use π_i as the extracted patterns for individual input word i. When explaining the whole input sentence, we collect all refined patterns for each word and use $\Pi = \bigcup_i \pi_i$. Further, we denote Π_+ as the set of patterns for

[2]Two reasons for using a fixed baseline of `[MASK]` as opposed to the midpoint in the previous chapter is that (1) we also trace the influence of other non-noun words so we choose one consistent baseline for all words. (2) The vocabulary of LSTM models does not contain a neutral word like `[MASK]`.

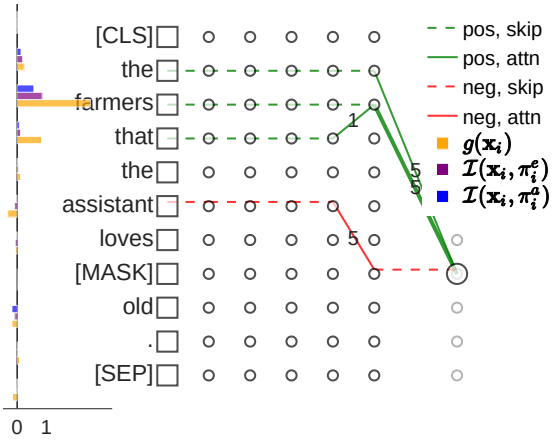
Task	Type	Example	BERT Small	BERT Base
SVA				
Object Relative Clause	SS	the author that the guard likes [MASK(is/are)] young	1	1
	SP		0.92	0.96
	PS		0.9	0.98
	PP		1	1
Subject Relative Clause	SS	the author that likes the guard [MASK(is/are)] young	1	1
	SP		1	0.96
	PS		1	0.98
	PP		1	1
Within Sentence Complement	SS	the mechanic said the author [MASK(is/are)] young	1	1
	SP		1	1
	PS		1	1
	PP		1	1
Across Prepositional Phrase	SS	the author next to the guard [MASK(is/are)] young	1	0.99
	SP		1	0.98
	PS		0.98	0.98
	PP		1	1
Reflexive Anaphora				
Number Agreement	SS	the author that the guard likes hurt [MASK(himself/themselves)]	0.66	0.6
	SP		0.66	0.74
	PS		0.83	0.83
	PP		1	0.96
Gender Agreement	MM	some wizard who can dress our man can clean [MASK(himself/herself)]	0.78	1
	MF		0.32	0.96
	FF		1	0.9
	FM		0.8	0.66

Table 3.1: Example of each agreement task and their performance on two BERT models.

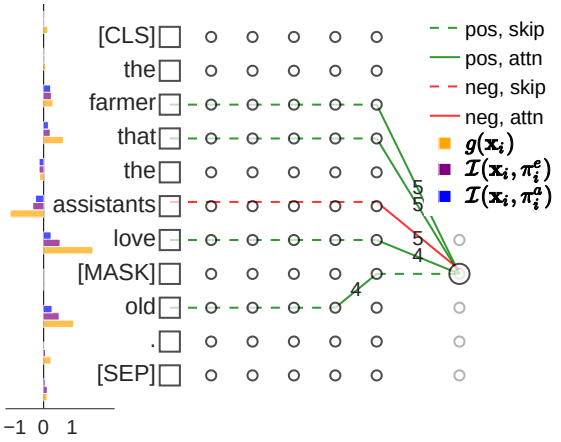
all positively influential words. Both terms may be further decorated by a or e to denote attention-level or embedding-level results. All experiments are done with a Titan V on a machine with 64 GB of RAM and implemented with tensorflow [Abadi et al., 2015]. On average GPR for attention-level and embedding-level take around half a minute to run for each instance in SVA and RA, and around 1 min for SA, with 50 batched samples to approximate influence. The whole quantitative experiment in Section. 3.5.4 across all tasks takes around 1 days cumulatively.

3.5.2 Visualizing Influence Patterns

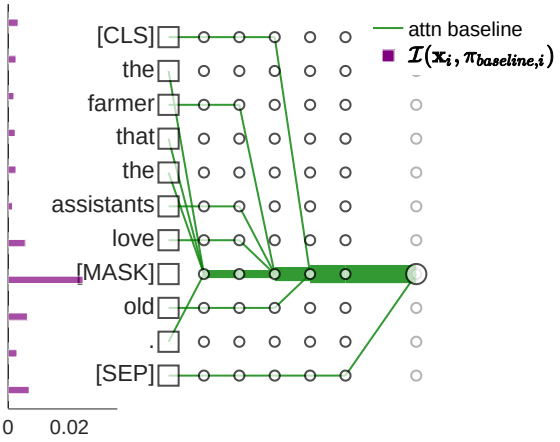
This section does not serve as an evaluation but an exploration of insightful results and succinct conclusions an human user can learn from our proposed technique. We visualize the information flow identified by



((a)) plural subject+singular attractor(PS)



((b)) singular subject+plural attractor(SP)

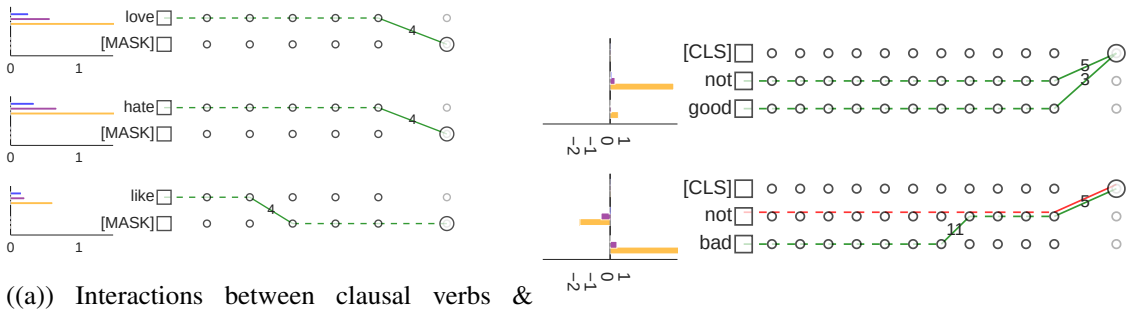


((c)) attention-based baseline Π_{attn}

Figure 3.3: (a)(b) Patterns for two instances of *SVA-Obj*.(c)Baseline pattern Π_{attn}. For each plot: Left: bar plots of the distributional influence $g(\mathbf{x}_i)$(yellow), $\mathcal{I}(\mathbf{x}_i, \pi_i^e)$ (purple) and $\mathcal{I}(\mathbf{x}_i, \pi_i^a)$ (blue) (or $\mathcal{I}(\mathbf{x}_i, \pi_{baseline,i})$) for each word at position i. Right: Extracted patterns Π from selective words. Square nodes and circle nodes denote input and internal embeddings, respectively. In (a) and (b), influence flowing through skip connections is represented by dashed lines and attention heads in solid lines; the edges are are marked with the corresponding attention head number (ranging from 1 to A) in Π^a . Line colors represent the sign of influence (red as negative and green as positive).

patterns found by GPR, and compare with those generated by the baseline method Π_{attn}.

We first focus on instances of the subtask *SVA across object relative clauses (SVA-obj)*, which are generated from the template: `the SUBJECT that the ATTRACTOR VERB [MASK](is/are) ADJ.`. We observe in Figure 3.3 that the subject words exert positive input influences on the correct choice of the verb, and the intervening noun (attractor) exerts negative influence, which is true for both $\mathcal{I}(\mathbf{x}_i, \pi_i^e)$



((a)) Interactions between clausal verbs & `[MASK]` for *SVA-Obj*

((b)) Interactions between *not* & adjectives in SA

Figure 3.4: (a) Patterns on three clausal verbs from SP, *SVA-obj*.(b) Patterns for two instances in the SA task. Legend follows Figure 3.3.

and $\mathcal{I}(\mathbf{x}_i, \pi_i^a)$ (blue and purple bars in Figure 3.3(a) and 3.3(b)). While Π_{attn}(Fig. 3.3(c)) can not distinguish between positively influential words and negative ones. Our other main findings are discussed as follows.

Finding I: Skip Connection Matters. Horizontal dashed lines in Figure 3.3 indicate that influence can flow through layers at the same word position via skip connections, which is not isolated as separate nodes Π_{attn} (shown in Fig. 3.3(c)). Fig. 3.3(a) and 3.3(b) also show that the influence from subject words `farmer[s]` travels through skip connections across layers before it transfers into the attention head 5 in the last layer, indicating the "number information" from the subject embedding flows directly to the output. Interestingly, this also explains why attentions can be pruned effectively without compromising the performance [Kovaleva et al., 2019, Michel et al., 2019, Voita et al., 2019] as important information may not flow through attentions at all. Namely, they are simply "copied" to the next layer through the skip connection. In fact, attention heads are traversed far less often than skip connections, which account for 75.4% of nodes in Π_a across all tasks evaluated.

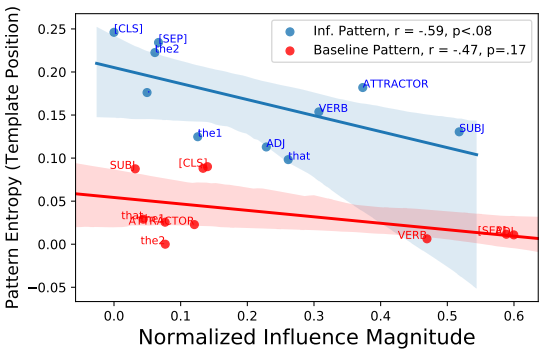
Finding II: `that` as an Attractor. Fig. 3.3(a) and 3.3(b) shows that the singular subject is less influential than the plural subject, especially when compared to the large negative influence from the attractor. Besides, the word `that` behaves like a singular pronoun in the singular subject case (Figure 3.3(b)), flowing through the same pattern as the subject (skip connections + attention head 5), whereas `that` is more like a grammatical marker (relativizer) in the plural subject case (Figure 3.3(a)): the pattern from `that` converge to the subject in the second to last layer via a different attention head 1. An explanation to the observed difference is that `that` can either be used as a singular pronoun in English, or a marker that encodes the syntactic boundary

of the clause to help identify the subject and ignore the attractor. This finding reveals that although the latter one is always the correct usage for `that` across all instances, BERT may resort to the "easier"(while ungrammatical) encoding of `that` as a pronoun when the number happens to be singular.

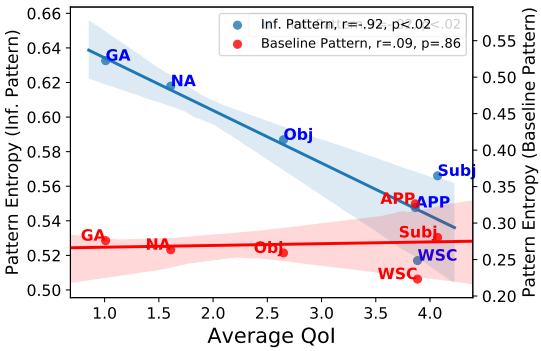
Finding III: `love`, `hate` and `like` Figure 3.4(a) shows the pattern across three instances containing different clausal verbs by replacing `love` in Figure 3.3(b) with `like` or `hate`. We observe that `hate` and `love` are more influential, with a distinct and more concentrated pattern compared to that of the word `like`, while one comparable to that of the subject or noun attractor. Therefore, `hate` and `love` are treated more like singular nouns than `like`, contributing positively to the correct prediction of the verb's number (but for the wrong grammatical reason). This discrepancy also corroborates different accuracy for instances in the three subsets: `hate`: .99; `love`: 1.0; `like`: .82. In fact, the number of misclassifications containing `like` accounts for more than 33% of all misclassifications in *SVA-Obj(SP)*.

Finding IV: Interactions with `not`. Two instances of sentiment analysis which BERT classifies correctly (negative for `not good.` and positive for `not bad.`) in Fig. 3.4(b) shows that: 1) in `not good.`, `not` and `good` does not interact internally and `not` carries much more positive influence than `good` towards the correct class (negative sentiment); 2) in `not bad.`, on the contrary, `not` contributes negatively to the correctly class (positive sentiment) and `bad` interacts with the `not` in an internal layer and exhibit a large positive influence. This disparity shows the model may treat `not` as a word with negative sentiment by default, and only when the subsequent adjective is negative, can it be used as a negation marker to encode the correct sentiment. Similar polarity of negation cues are also observed in prior works [Chen and Jordan, 2020, Zhu et al., 2014, Barnes et al., 2021], while influence pattern provides a more precise analytical approach to surface it.

The asymmetries and oddities of patterns of Finding II to IV show how BERT may use ungrammatical and conceptually unsound correlations for its predictions (We include more examples of patterns in Appendix 6.3). However, one may ask, what are the negative implications of these results? If a model is able to learn the correct correlations for a certain concept as demonstrated by the performances of these tasks, why does it matter that the model learns a different rule from humans? Our answers to these hypothetical questions is two-fold: first, a conceptual sound model should always follow the grammatical rules as they are the root causes for constructing a sentence in a certain way rather than the other; second, although we do not include



((a)) Pattern Entropy vs. Average Influence Magnitude for template positions in *SVA-Obj*



((b)) Pattern Entropy vs. Average QoI for linguistic tasks (SVA & RA)

Figure 3.5: Relationship between Task Performance, Influence Magnitude and Pattern Entropy

counterfactuals evaluation or adversarial examples in this work, we speculate that there exist scenarios where failing to learn a consistent grammatical rule can cause models to fail on instances outside the training distribution. We have demonstrated similar findings in the next Chapter (Section. 4.6), through the creation of adversarial examples leveraging explanations.

3.5.3 Consistency of Patterns Across Instances

The prior section offered examples building connections between information flow and patterns. Since patterns are computed for each word of each instance individually, a global analysis of consistency/variation of patterns across instances can help us make deeper insights and make more general conjectures. We quantify this variation with *pattern entropy* on the linguistic tasks (SVA & RA) with fixed templates, where for each instance, each template position are chosen from a group of words. Interpreting the collection of instances, or template positions, in a task as a distribution, pattern entropy is the entropy of the binary probability that a node is part of a pattern, averaged over all nodes (minimally this is 0 if all patterns incorporate the exact same set of nodes). With a node n in graph G, a set of K instances, π_k as the pattern of instance k by abuse of notation and H as the entropy function: pattern entropy of a pattern π is defined as:

$$\hat{H}(\pi, G) = \frac{\sum_{n \in G} H(p_n(\pi))}{|G|}; p_n(\pi) = \frac{\sum_k \mathbb{1}(n \in \pi_k)}{K}$$

Figure 3.5(a) shows an inverse linear relation between influence magnitude and the pattern entropy of each template position(π_i) in *SVA-Obj*. For example, the subject position has on average high influence

magnitude and low pattern entropy, while grammatically unimportant template positions such as [CLS] and the period token has relatively low influence and high entropy. Note that this fit is not perfect: attractors, for example, have high influence magnitude but also high entropy's due to their disparate behaviors among instances shown in Sec 3.5.2.

Figure 3.5(b) demonstrates a more global inverse relation between the pattern entropy of Π_+ and task-specific performance(average QoI) for 6 linguistic subtasks studied. This indicates that the more consistent a pattern is for a concept, i.e, the more consistent how a model locates the positively influential signals, the better the model is at capturing that concept. In both figures, the baseline attention-based patterns (Π_{attn}) does not indicate this relation. Specially, the patterns computed with attention flows have consistently low entropy across tasks, indicating they generate similar explanations regardless of how well a task is learned. Together with Finding II to IV in Sec. 3.5.2, it demonstrates that patterns may offer alternative insights in formalizing and answering high-level questions such as to what extent does BERT generalize to correct and consistent grammatical rules or use spurious correlations which varies across instances [McCoy et al., 2019, 2020, Ribeiro et al., 2020]. This observation also relates to recent studies linking explainability to model robustness: robust models tend to be explainable [Datta et al., 2021], since the more robust a model is, the less likely it relies on some highly inconsistent and spurious correlations that we do not understand.

3.5.4 Evaluating Influence Patterns

This section serves as a formal evaluation of our proposed methods against existing baselines in tracing information flows. Except the attention baseline Π_{attn}, Π_{inf} and Π_{cond} defined in Sec. 3.5.2, we also include the baseline, Π_{rand}, which describes patterns with randomly sampled nodes.

We compare various aspects of extracted patterns against the following baselines, including: (1) ablation experiments showing how influence patterns account for model performance; (2) the sparsity of the patterns in the computational graph. Finally, we also discuss the relation between patterns and attention-weights.

Ablation. We extend the commonly-used ablation study for evaluating explanation methods [DeYoung et al., 2020, Dabkowski and Gal, 2017, Ancona et al., 2018, Leino et al., 2018, Lu et al., 2020b] from input features to internal patterns as a sanity check of our method. That is, we ablate BERT down to a simpler model: we only retain the nodes from Π^e_+ (or Π^a_+) while replacing other nodes by zeros. The retained and replaced

Patterns	SVA				RA		SA
	Obj.	Subj.	WSC	APP	NA	GA	
Π^e_+(ours)	**.96**	**.99**	**1.0**	**.96**	**.98**	**.99**	**.97**
Π^e_{rand}	.55±.017	.60±.014	.55±.016	.55±.016	.55±.015	.56±.014	.52±.030
Π^e_{attn}	.61	.55	.49	.63	.56	.65	.47
Π^e_{cond}	.93	.91	.88	.90	.71	.95	.94
Π^e_{inf}	.66	.71	.50	.56	.68	.50	.49
Π^a_+(ours)	**.70**	**.62**	**.56**	**.65**	**.78**	**.89**	**.86**
Π^a_{rand}	.50±.010	.50±.008	.50±.009	.50±.011	.50±.011	.50±.008	.49±.022
$\Pi^a_{\text{repl\_skip}}$	.50	.58	.54	.52	.55	.50	.48
original	.96	1.0	1.0	1.0	.83	.73	.92

Table 3.2: Ablated accuracies of Π_+ and baseline patterns, with $* \in \{e, a\}$, denoting embedding & attention-level metrics.

Metrics	SVA				RA		SA
	Obj.	Subj.	WSC	APP	NA	GA	
conc.(Π^e_+)	.33	.30	.27	.33	.31	.22	.07
conc.(Π^e_-)	.29	.30	.38	.31	.31	.23	.05
share(Π^e_+)	.06	.09	.03	.06	.04	.02	.01
conc.(Π^a_+)	.21	.18	.16	.18	.18	.14	.01
conc.(Π^a_-)	.17	.16	.25	.15	.17	.15	.01
share(Π^a_+) ($\cdot 10^{-6}$)	1.8	1.7	1.6	1.5	1.3	1.3	3e-2
Alignment Rate	.58	.57	.57	.59	.46	.53	.34

Table 3.3: Sparsity metrics; conc.($\Pi^*_\pm$): positive/negative *concentration*; share(Π^*): *path share* of pattern Π^*. , with $* \in \{e, a\}$, denoting embedding & attention-level metrics.

nodes together are forward passed to the next layer with the original model parameters until a new set of nodes are retained or replaced. *Ablated Accuracy* denotes the accuracy of the ablated model. We show the results for our methods Π^*_+ and other baseline patterns Table 3.2. We run Π_{rand} 50 times per task and average the ablated accuracy, finding with no significant variation.

Each baseline retains the same number of nodes as the corresponding Π_+. Although this process seems to be highly invasive, the model ablated with Π^*_+ achieved the highest ablated accuracy uniformly over all tasks. All baseline patterns except Π^e_{cond} produce random guesses. The relatively higher ablated accuracy of Π_{cond} is expected: pattern influence (Def. 8) using our settings of $\mathcal{D}$ and QoI, with exactly one internal node, reduces to conductance [Dhamdhere et al., 2018], therefore picking most influential node per layer using conductance is likely to achieve similar patterns. However, this gap in ablated accuracy between Π_{cond} with Π^e_+, albeit small, shows the utility of the GPR algorithm over a comparable layer-based approach. For the

attention-level graph Π_a, we also add a counterfactual pattern $\Pi^a_{\text{repl\_skip}}$ where we replace each skip node in Π^a_+ with its all A corresponding attention heads. However, with much fewer nodes ablated, $\Pi^a_{\text{repl\_skip}}$ still produces random ablated accuracies. This corroborates Finding I in Sec. 3.5.2 that the skip connections relaying important information directly which attention block cannot replace. In summary, patterns refined by GPR account for the model's information flow more sufficiently compared to all baselines.

Sparsity of Patterns. A good explanation should not only account for the model performance, but also be sparse relative to the entire model semantics so as to be interpretable to humans. In this section, we quantify the sparsity of extracted patterns using two metrics: *path share* and *concentration*. With on average more than 10 words per sentence, the embedding and attention-level graphs contain at least 10^6 and 10^{11} individual paths, respectively. The totality of these paths represent the entire semantics of the BERT model. *Path share*(i.e. $\text{share}(\Pi^*_+) \stackrel{\text{def}}{=} |\gamma_*(\Pi^*_+)|/|\mathcal{P}^*|$), is defined as the number of paths in a pattern over the total number of paths in the entire computational graph. *concentration*, on the other hand, is defined as the proportion of negative/positive pattern influence over the total positive/negative influence. It represents the average ratio between the blue/purple bars and the orange bars in Figure 3.3.

Table 3.3 shows that the abstracted patterns contain only a small share of paths while accounting for a large portion of both positive and negative influence. In linguistic tasks, the embedding level abstracted pattern has a *concentration* around 0.3 ($\text{conc.}(\Pi^e_\pm)$), indicating that the input concept flows through single internal embeddings in each layer, instead of distributed to many words. Zooming in on the attention-level graph, a relative high concentration $\text{conc.}(\Pi^a_+)$, suggests that between the internal embeddings of adjacent layers, influence is also more concentrated to either one attention head or the skip connection. We speculate the much lower $\text{conc.}(\Pi_+)$ in SA is due to (1) the much larger model for the SA task with more layers and attention heads (2) sentiment information may be more diverse and complex than the information needed to encode syntax agreements, therefore input information may flow in a more distributed way. However, despite low concentration of pattern influence, the extracted patterns for SA are still effective in capturing the model performance as shown in Table 3.2.

Attention heads in Π^a vs. Attention weight value. Another observation is that the attention head nodes in Π^a found by GPR aligns, to greater extent than random ($1/|A|$), with the head of the largest attention weight along the corresponding embedding-level edge, as shown by *alignment rate* in Table 3.3. This partial

alignment is expected since the Jacobian between two nodes correlates with the coefficients (weights) in the linear attention mechanism(product rule), while the opposite is also expected since (1) attention weights themselves are not fixed model parameters thus part of the gradient flow, (2) model components other than attention blocks come between two adjacent layers (e,g. dense layer & skip connections). In other words, attention weights are correlated, but not equivalent to gradient-based methods as explanations.

3.6 Related Work

Previous work has shown the encoding of syntactic dependencies such as SVA in RNN Language models [Linzen et al., 2016, Hupkes et al., 2018, Lakretz et al., 2019, Jumelet et al., 2019]. More extensive work has since been done on transformer-based architectures such as BERT. Probing classifiers has shown BERT encodes many types of linguistic knowledge [Elazar et al., 2021, Hewitt and Liang, 2019, Tenney et al., 2019a,b, Jawahar et al., 2019, Klafka and Ettinger, 2020, Liu et al., 2019a, Lin et al., 2019, Reif et al., 2019, Hewitt and Manning, 2019]. Goldberg [2019] discovers that SVA and RA in complex clausal structures is better represented in BERT compared to an RNN model.

A line of related works analyze self-attention weights of BERT [Clark et al., 2019, Vig and Belinkov, 2019, Lin et al., 2019, Ethayarajh and Jurafsky, 2021], where attention heads are found to have direct correspondences with specific dependency relations. Abnar and Zuidema [2020] and Wu et al. [2020], with which we compare extensively in this paper, propose using attentions to quantify information flows in BERT. Attention weights as interpretation devices, however, have been controversial [Serrano and Smith, 2019] and empirical analysis has shown that attention can be perturbed or pruned while retaining performance [Kovaleva et al., 2019, Michel et al., 2019, Voita et al., 2019]. Our work demonstrates that attention mechanisms are only part of the computation graph, with each attention block complemented by other model components such as dense layer and skip connections; axiomatically justified influence patterns, however, can attribute to the whole computation graph. The strong influence passing through skip connections also corroborates the findings of Brunner et al. [2020] which finds input tokens mostly retain their identity. Besides pruning attentions, other works [Prasanna et al., 2020, Sanh et al., 2019, Jiao et al., 2019] also show that BERT is overparametrized and can be greatly compressed. Our work to some extent corroborates that point by pointing to the sparse gradient flow, while employing ablation studies only to verify the sufficiency of the

extracted patterns.

A closely related and concurrent work [Dong et al., 2021] also shows the importance of skip connections and dense (MLP) layers by decomposing and analyzing forward-pass computations in self-attention modules. Our work, in comparison, introduces a gradient-based method that can be generalized to any model as long as they are differentiable. We also focus on how influential patterns can help us understand information flow of specific NLP tasks.

3.7 Conclusion

We demonstrated influence patterns for explaining information flow in BERT. We highlighted the importance of skip connections and BERT's potentially mishandling of various concepts through visualized patterns. We inspected the relation between consistency of patterns across instances with model performance and quantitatively validated pattern's sufficiency in capturing model performance.

Chapter 4

Order-sensitive Shapley Values for Evaluating Conceptual Soundness of NLP Models

4.1 Introduction

Recent works discover that NLP models are not sensitive to word orders in the same way as humans [Pham et al., 2021, Sinha et al., 2021a, Clouatre et al., 2021, Alleman et al., 2021, Sinha et al., 2021b]: although randomly shuffled sentences are often incomprehensible to humans, they result in very small performance drop for the state-of-the-art Transformer models such as BERT [Devlin et al., 2019], for both syntactic and semantic classification tasks. The underlying hypothesis behind such insensitivity is that the occurrences or co-occurrences of words are often sufficient for learning the tasks [Malkin et al., 2021]. However, relying on such heuristics without word order is not *conceptually sound* [Parkinson, 2011], i.e. the models fail to learn the correct linguistic concepts. One aspect of conceptual soundness, which is the focus of this paper, is *order-sensitivity*. A model may ignore order-sensitive concepts and thus become susceptible to adversarial examples. One notable example is HANS [McCoy et al., 2019], which discovers that Natural Language Inference (NLI) models wrongly predict entailment between hypothesis and premise sentences with overlapping words but different ordering, such as *The doctors visited the lawyers* entailing *The lawyers*

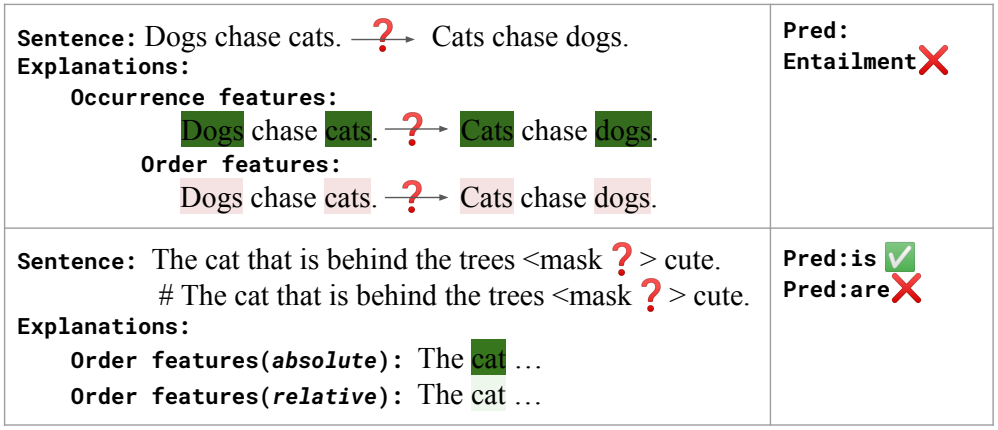


Figure 4.1: Two examples of Order-sensitive explanations. Two colors represent opposite signs of attributions while darker shade represents larger magnitude. Top: In an NLI model, attributions to the entailment class are small and negative for order features (light pink) but large and positive for occurrence features (dark green), resulting the wrong prediction of entailment between inverted sentences; Bottom: To learn subject verb agreement: a language model relies on absolute positions of the subject word (larger attribution to absolute order features than relative ones), resulting in an error when the subject's position is shifted by prepending the token #.

visited the doctors. To systemically understand how and to what extent the models capture order information, a method specially designed to explain sequence orders is needed.

Game theoretic approaches such as Shapley values [Shapley et al., 1953] are widely adopted to explain the predictions of deep models by assigning influences to features [Strumbelj and Kononenko, 2010, Datta et al., 2016, Lundberg and Lee, 2017, Sundararajan and Najmi, 2020, Covert et al., 2020]. Prior work applying Shapley values to textual data or other sequential data, however, treat sequential data as tabular data without attributing to the order of features in a sequence.

In this work, we propose a new model-agnostic explanation framework, *Order-sensitive Shapley Values* (*OSV*), for attributing to the *order* of features along with *occurrence* of features in a sequence. We propose a mechanism for separating these two groups of features and two modes of intervening on sequence orders, *absolute position* and *relative order*. Most importantly, we show that *OSV* as an explanation device can help answer the questions of *conceptual soundness* of NLP models: **does the model learn the correct order-sensitive syntax, or is the model exploiting mere occurrences of words that are possibly spurious correlations?** We conduct an extensive empirical study applying *OSV* to synthetic datasets and several NLP tasks: Natural Language Inference (NLI), Sentiment Analysis (SA) and Subject Verb

Agreement (SVA). For each task, we demonstrate how *OSV* is the appropriate explanation device to precisely diagnose the root cause of spurious correlations by constructing adversarial examples based on the generated explanations. Two examples of *OSV* applied to NLI and SVA are shown in Figure 4.1. Our contributions and findings are summarized below:

- We propose Order-sensitive Shapley Values (*OSV*) by expanding the standard feature set to include *order features* (Section. 4.3.1). Using synthetic data, we show that *OSV* is more faithful in explaining model's behavior than gradient-based methods (Section. 4.5);
- Applying *OSV* to the HANS dataset [McCoy et al., 2019], we discover that BERT uses only word occurrences without word orders. Data augmentation [Min et al., 2020] improves accuracy on HANS, but *OSV* shows that the augmented model does not fundamentally improve the model's learning of order features (Section. 4.6.1);
- We discover that sentiment analysis models, such as BERT, do not learn negation properly: they fail to capture the correct syntax of negators preceding adjectives. Interestingly, RoBERTa [Liu et al., 2019b] and StructBERT [Wang et al., 2019] produce more conceptually sound explanations than BERT [Devlin et al., 2019] (Section. 4.6.2);
- We show that pretrained language models such as BERT rely on the absolute positions of subject words for capturing long-range Subject-Verb Agreement (Section. 4.6.3).
- In each NLP task, we show how insights drawn from *OSV* can be exploited to generate adversarial examples.

4.2 Background

4.2.1 Shapley value as an attribution method

A large group of explanation methods, summarized by Covert et al. [2021], quantify the impact of individual features (or groups of features) by measuring how removing those features change the outcome of model predictions. In this paper, we focus on one axiomatic method inspired by coalition game theory, Shapley

values [Shapley et al., 1953]. Given a set of n players represented by integers $N = \{0, 1, \ldots, n-1\}$ and a value function $v : 2^n \mapsto \mathbb{R}$, the *Shapley value* of player i is:

$$\phi_v(i) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!} \Big(v(S \cup \{i\}) - v(S) \Big) \tag{4.1}$$

In the machine learning setting, to apply Shapley values for a classification model f with input features $x = \{x_0, x_1, \ldots, x_{n-1}\}$, x are treated as individual players and a function $f^y(x) : X \mapsto \mathbb{R}$ related to the model output $f(x)$ is treated as the value function. For example, $f^y(x)$ can be the output probability of the predicted class for local explanations [Lundberg and Lee, 2017], a loss function for global explanations [Covert et al., 2020], or any user-defined function of the output logits [Datta et al., 2016].

Intervention on removed features A distribution $g(x_{\bar{S}}|x_S)$ for intervening on removed features ($\bar{S} = N \setminus S$) is required to compute $f^y(x)$ for feature subset x_S, The choice of g, however, has been a subject of debate in recent studies [Chen et al., 2020, Kumar et al., 2020, Catav et al., 2021, Frye et al., 2020, Hase et al., 2021]. Chen et al. [2020] in particular describes a choice between either conditional or interventional expectation for g: the former respects the joint distributions of features to explain the *data*, while the latter breaks the correlation among features to explain the *model*. In this paper, we adopt the latter: specifically, we inspect model's response to reordered sequences of words that may deviate from the underlying data distribution, allowing for a thorough understanding of the model's mechanism of encoding orders in sequences.

4.3 Method

In this section, we present the extension of Shapley Values (*SV*) into Order-sensitive Shapley Values (*OSV*) by expanding the feature set with *order features*. Next, we explore the connections between the two and propose two mechanisms of intervention on order features, attributing to the *absolute* or *relative* positions of elements in a sequence, respectively. Finally, we bridge the axiomatic properties of Shapley Values in the context of *OSV*.

4.3.1 Set Representation of a Sequence

We start by defining a sequence as a *set* of features. A finite ordered sequence $[\mathbf{w}]$ can be represented by a union of two sets $w = x \cup z$ where the *occurrence feature* set $x = \{x_0, x_1, \ldots, x_{n-1}\}$ and the integer *order feature* set $z = \{z_0, z_1, \ldots, z_{n-1}\} = N$, with a bijective mapping $h : z \mapsto x$, $h(z_i) = x_i, \forall i \in N$. The sequence is simply:

$$[\mathbf{w}]_{z_i} = x_i, \forall i \in N$$

We use $\mathbf{w}$, $\mathbf{x}$ and $\mathbf{z}$ to represent sets with ordered assignment of values for elements in otherwise orderless sets w, x and z: $\mathbf{w} = (x_0, x_1, \ldots, x_{n-1}, z_0, z_1, \ldots, z_{n-1})$ and similarly for $\mathbf{x}$ and $\mathbf{z}$. $[\mathbf{w}]$ can also be represented from w by enumerating elements of x indexed by the corresponding elements in z: $[\mathbf{w}] = \pi_{\mathbf{z}}(\mathbf{x})$. In the opposite direction of the mapping, since many combinations of x and z represent the same sequence $[\mathbf{w}]$, we assume by default $\mathbf{z} = (0, 1, \ldots, n-1) = \mathbf{N}$ when mapping $\mathbf{w}$ to w. We use $\mathbf{w}$, w and $[\mathbf{w}]$ interchangeably when referring to the same sequence.

4.3.2 Order-sensitive Shapley Value for Sequential Inputs

A sequence $[\mathbf{w}]$ of length n can be treated as a game with $2n$ players with the feature set w. Attributions to w measure how important it is for a feature to be present, while attributions to z measure how important it is for a feature to be in the correct position, they are computed together by plugging in Equation. 4.1:

Definition 12 (Order-sensitive Shapley Value)**.**

$$\phi^z_{f^y}(w_i) = \sum_{S \subseteq N' \setminus i} \frac{|S|!(2n - |S| - 1)!}{(2n)!} m(S) \tag{4.2}$$

$$\begin{aligned}
\textit{with } m(S) &= f^y(\mathbf{w}_{S \cup i}) - f^y(\mathbf{w}_S) \\
f^y(w_S) &= \mathbb{E}[f^y(\mathbf{w}) | \mathbf{w}_S] \\
&= \mathbb{E}_{\mathbf{w}_{\bar{S}} | \mathbf{w}_S}[f^y(\mathbf{w}_S)] \\
&\approx \mathbb{E}_{\mathbf{z}_{\bar{S}_z} | \mathbf{z}_{S_z}} \mathbb{E}_{\mathbf{x}_{\bar{S}_x} | \mathbf{x}_{S_x} \cup \mathbf{z}_{S_z}}[f^y(\mathbf{x} \cup \mathbf{z}_{S_z})] && (4.3) \\
&\approx \mathbb{E}_{\mathbf{z} \sim q(\cdot | \mathbf{z}_{S_z})} \mathbb{E}_{\mathbf{x} \sim g(\cdot | \mathbf{x}_{S_x} \cup \mathbf{z})}[f^y(\pi_{\mathbf{z}}(\mathbf{x}))] && (4.4)
\end{aligned}$$

$$
\begin{aligned}
\textit{where: } N' &= \{0, 1, \ldots, 2n-1\} \\
S_x &= \{i | \mathbf{w}_i \in x \wedge i \in S\}, \overline{S_x} = N \backslash S_x \\
S_z &= \{i - n | \mathbf{w}_i \in z \ \wedge i \in S\}, \overline{S_z} = N \backslash S_z
\end{aligned}
$$

$S \cup i$ abbreviates $S \cup \{i\}$. Choices of g is discussed in Section. 4.2.1 and choices of q is to be discussed in Section. 4.3.3. In Equation. 4.3, we assume features in z are independent from those in x, in other words, how occurrence features can be ordered into a sequence does not depend on the value of the occurrence features themselves. In Equation. 4.4, we assume that the intervention of occurrence features is only done in the context of the original sequence. With these two assumptions, we align *OSV* with *SV* through the following remarks:

Remark 1. *when $q(\cdot) = \mathbf{N}$, ϕ^z reduces to the (order-insensitive) Shapley values ϕ as follows:*

$$
\phi^z_{f^y}(x_i) = \phi_{f^y}(x_i), \forall x_i \in x
$$

$$
\phi^z_{f^y}(z_i) = 0, \forall z_i \in z
$$

Proof:

An alternative definition [Datta et al., 2016] for $\phi^z_{f^y}(i)$ is:

$$
\begin{aligned}
\forall x_i \in x, \phi^z_{f^y}(x_i) &= \frac{1}{(2n)!} \sum_{\sigma \subseteq \Pi(N')} m_i(\sigma) \\
&= \frac{1}{(2n)!} \cdot \frac{(2n)!}{n!} \sum_{\sigma \subseteq \Pi(N)} m_i(\sigma) \\
&= \frac{1}{n!} \sum_{\sigma \subseteq \Pi(N)} m_i(S) \\
&= \phi_{f^y}(x_i) \\
\textit{where}: m_i(\sigma) &= m\left(P_i(\sigma)\right) \\
P_i(\sigma) &= \{j \mid \sigma_j < \sigma_i\}
\end{aligned}
$$

P_i is the set of i's predecessors in σ. $\forall z_i \in z$, according to the dummy axiom, since z is omnipresent, $f^y(S \cup i) = f^y(S)$, so $\phi^z_{f^y}(z_i) = 0, \forall z_i \in z$. Remark 1 states that if we assume the omnipresence of order features, *OSV* for order features are all zero and *OSV* for occurrence features reduce to corresponding order-insensitive Shapley values (*SV*).

Remark 2. *If X is the set of intervened sequences used for computing* SV*, and Z for computing the corresponding* OSV*, then $Z \subseteq \{z | z \in \Pi(x), \forall x \in X\}$. where Π is a function mapping to all permutations of elements in X: $\Pi(\mathbf{w}) = \{\pi_{\mathbf{z}}(\mathbf{w}) : \forall \mathbf{z}\}$.*

Remark 2 stipulates that the space of intervened sequences evaluated for ϕ^z is encompassed by permutating those evaluated for the corresponding ϕ. The intervention on order features is therefore orthogonal to the choice of g, enabling the potential expansion of order features to other framework besides Shapley values, such as Banzhaf values [Banzhaf III, 1964] or LIME [Ribeiro et al., 2016].

Remark 3. *A model f^y is* completely order-sensitive *if $f^y(\mathbf{w}_S) = y_\varnothing, \forall S_z \neq N$:*

$$\phi^z_{f^y}(x_i) = \sum_{S_x \subseteq N \setminus i} \frac{(|S_x| + n)!(n - |S_x| - 1)!}{(2n)!} m(S_x) \tag{4.5}$$

$$= \sum_{S_x \subseteq N \setminus i} p(S_x) \frac{|S_x|!(n - |S_x| - 1)!}{n!} m(S_x) \tag{4.6}$$

$$\textit{where: } p(S_x) = \frac{(2n - |S_x| - 1)!n!}{(n - |S_x| - 1)!(2n)!}$$

$$\forall z_i, \phi^z_{f^y}(z_i) = \sum_{S_x \subseteq N} \frac{|S_x + n|!(n - |S_x| - 1)!}{(2n)!} [f^y(\mathbf{x}_{S_x}) - y_\varnothing)]$$

Remark 3 describes a hypothetical, *completely order-sensitive model* f^y undefined on all reordered sequences (those that miss any feature from z), thus output a constant ($y_\varnothing$)). For example, the joint distribution of words in a sentence for most natural languages is so sparse that almost all reordered sentences are out of distribution. Equation. 4.6 shows the relation between the attribution to ϕ^z and the corresponding ϕ: each term in the sum of Equation. 4.1 is reweighted by $p(S_x)$. Attributions to order features ($\phi(z)$) are all equal since missing any order feature results in $y_\varnothing$. For longer sequences, $p(S_x)$ becomes extremely small, making $\phi(x)$ negligible compared to $\phi(z)$. In other words, under a *completely order-sensitive model* f^y, *OSV* should be dominated by order features. However, in the real world, neither a human nor an NLP model is completely order-sensitive. Some tasks inherently tolerate reordered sentences than others: For a sentiment analysis model, for instance, even though *film the good very is* is technically not a valid sentence thus no sentiment shall be assigned, most models (and human) would still justifiably label it with positive sentiment, while more syntactic tasks such as language modeling should be far more order-sensitive, where model's response to incoherent sentences should be more neutral therefore $\phi(z)$ should be larger. This hypothesis is confirmed by various NLP tasks evaluated in Section. 4.6.

4.3.3 Intervention on order features

Assuming the independence of order features from occurrence features, the only missing piece left is to choose an appropriate q for sampling $\mathbf{z} \sim q(\cdot|\mathbf{z}_{S_z})$. A key premise is that any q should guarantee a valid sequence, i.e, $q(\cdot|\mathbf{z}_{S_z}) = N$. We propose the following two q, evaluating two distinctive notions of ordering: *absolute position* and *relative order*.

Definition 13 (Absolute-position order intervention)**.**

$$q^a(\cdot|\mathbf{z}_{S_z}) = \mathit{Unif}(\Pi(\mathbf{z}_{\overline{S_z}}); \mathbf{z}_{S_z})$$

For example, for a sequence $[\mathbf{w}] = (a, b, c, d)$ and $S_x = \{0, 1, 2, 3\}$, $S_z = \{0, 1\}$, then $q^a(\cdot|\mathbf{z}_{S_z}) = \text{Unif}\{(a, b, c, d), (a, b, d, c)\}$.

Definition 14 (Relative-order order intervention)**.**

$$q^r(\cdot|\mathbf{z}_{S_z}) = \mathit{Unif}(\Pi(\mathbf{z}_{\overline{r(S_z)}}); \mathbf{z}_{r(S_z)})$$

where $r(\cdot)$ does a random non-wrapping shifting of the feature positions for unintervened features.

With the setting of the previous example, $q^r(\cdot|\mathbf{z}_{S_z}) = \text{Unif}\{((a, b, c, d), (c, a, b, d), (c, d, a, b), (a, b, d, c), (d, a, b, c), (d, c, a, b)\}$.

We denote *OSV* computed with q^a and q^r as ϕ^a and ϕ^r, respectively. It is apparent from Def.13 and 14 that $Supp(q^a) \subseteq Supp(q^r)$. With ϕ^a, we consider an order feature as important if deviating from its absolute position cause a large change in $f^y(x)$. With ϕ^r, on the other hand, an order feature is only considered as important if breaking its relative ordering with other features affects $f^y(x)$. For natural language or other variable-length data, ϕ^r can help assess if the model captures the order of n-gram features. Similar notions of shuffling n-gram features have been explored in Sinha et al. [2021a] and Alleman et al. [2021], while q^r summarize all n-gram coalitions and attribute them to individual order features. q^a, on the other hand, follows a stricter and more general notion of *order* where the position of unintervened order features will not change. q^a may also work well with sequential data with a meaningful starting position such as time-series data. The difference between q^a and q^r also functions as a diagnostic tool to see if a model relies on the absolute position of features when it should not: Section. 4.6.3 explore an NLP example on SVA.

4.3.4 Axioms of Shapley Values

Note that all axioms of Shapley values [Shapley et al., 1953, Datta et al., 2016] extend to *OSV*, in this section, we provide some intuitions on how the feature set expanded with order features can be interpreted in the context of each axiom. The axiomatic properties of Shapley values are first introduced in Shapley et al. [1953] and extensively discussed in the context of machine learning in Lundberg and Lee [2017] and Datta et al. [2016]. Shapley values defined in Equation. 4.1 is the only value satisfying all following axioms. The names in parenthesis for each axiom are alternative names used in literature.

Symmetry Axiom $i, j \in N$ are *symmetric* if $v(S \cup i) = v(S \cup j)$ for all $S \subseteq N \backslash i, j$. A value ϕ satisfies symmetry if $\phi(i) = \phi(j)$ whenever i and j are symmetric.

Extending to order features, if two order features are symmetric, meaning that their positions are always interchangeable, then their attributions should equal each other. Though theoretically possible, order features are unlikely to be symmetric with occurrence features.

Dummy Axiom (Null Effects/Missingness) A player is a dummy or null player if $v(S \cup i) = v(S)$ for all $S \subseteq N$. A value ϕ satisfies the dummy axiom if $\phi(i) = 0$ whenever i is a dummy or null player.

An order feature is a dummy: (1) if the model doesn't care where it is in a model, such as a Bag-of-Words model; (2) if the order feature is omnipresent as in Remark 1.

Completeness Axiom (Efficiency or Local Accuracy) A value ϕ satisfies completeness if $\sum_i^N \phi(i) = v(N)$. This axiom is important as it makes Shapley value an *attribution value* in that it *attributes* and *allocates* the output of the function to individual inputs. With the augmented feature set, the total output is allocated to both the order features and occurrence features.

Monoticity Axiom (Consistency) A value ϕ satisfies monoticity if $m(S, v_1) \geq m(S, v_2)$ for all S implies that $\phi_{v_1}(i) \geq \phi_{v_2}(i)$. The theorem of monoticity states that if one features' contribution is greater for one model than another model regardless of all other inputs, that input's attribution should also be higher. It can be extended directly to order features.

Table 4.1: Tasks and performance of LSTM models. Acc.: mean test accuracy for each model over 5 random seeds, Acc-1: mean test accuracy for $[\mathbf{W_1}]$.

Ind.	Condition for y=1	Acc.	Acc-1.
1	Sequence beginning with duplicate	1.00± 0.00	1.00± 0.00
2	Adjacent duplicate in the sequence	1.00± 0.00	1.00± 0.00
3	Any duplicate in the sequence	0.99± 0.00	1.00± 0.00

4.4 Approximating Shapley values

One disadvantage of Shapley value is that it is often computationally intractable, which could be exacerbated by the doubling of feature size for *OSV*. However, many approaches have since been proposed to address this issue such as SampleShapley [Strumbelj and Kononenko, 2010, Datta et al., 2016], SHAP [Lundberg and Lee, 2017], and DASP [Ancona et al., 2019], all of which are extendable to *OSV*. In Section. 4.6 and Section. 4.5, we use *OSV* as global explanations [Covert et al., 2020] for faster approximation, especially when we are only interested in overall feature importance.

To make sure that the global explanations truly represent the global contribution of each feature towards the prediction, we use a convergence factor of $t = 0.005$ for all global explanations, meaning the variance of the Shapley value is less than 0.5% of the difference between the maximum and minimum $\phi(i)$ for all i (a looser $t = 0.01$ is sufficient for convergence according to Covert et al. [2020]). For each S, we use a sample size of 4 for q and a sample size of 5 for g when g samples from templates (total of 20 intervened sentences per instance per S). We find this setting generates stable explanations for the synthetic data (low variance in Table 4.3). Moreover, since the stopping condition is based on t and the algorithm for computing global explanations may iterate through the whole data multiple times to reach that convergence, the actual sample size of q and g in principle should not matter with a fixed t. We include an comparative analysis of the computational efficiency between *OSV* and baseline gradient-based methods in Section. 4.5.

4.5 Synthetic Data Experiment

Since it is often difficult to evaluate the faithfulness [Jacovi and Goldberg, 2020] of an explanation method [Ju et al., 2021, Zhou et al., 2021], we first experiment with synthetic data where ground truth explanations

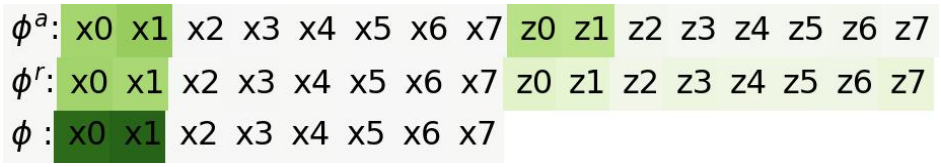
Ind.	Condition for y=1	Acc.	Acc-1.
1	Begins with duplicate	1.00± .00	1.00± .00
2	Adjacent duplicate	0.98± .00	0.99± .01
3	Any duplicate	1.00± .00	1.00± .00

Table 4.2: Tasks Description and Model performances for Synthetic Data Experiments for transformer models. Acc. are test accuracies for each model over 5 random seeds, Acc-1 are accuracies for $[\mathbf{W_1}]$.

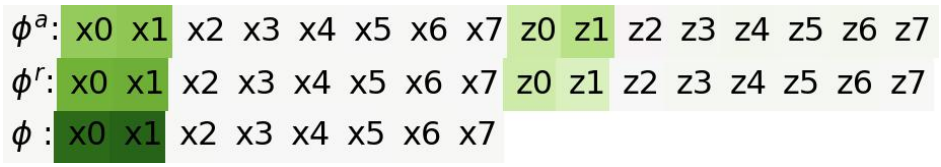
are available. Our data and models are inspired by a similar experiment by Lovering et al. [2020]. We will publicly release the code for this section and Section. 4.6 once the paper is published.

Models and Data We experiment with two model architectures: (1) an RNN with 4-layer LSTM [Hochreiter and Schmidhuber, 1997] and an MLP with 1 hidden layer and ReLU activation. The embedding dimension E and hidden size H of LSTM are both 512. (2) a Transformer encoder model [Vaswani et al., 2017] with 2 self-attention layers, 8 attention heads and same E, H and MLP layer as the RNN model. We train 5 randomly seeded models for three k-length binary sequence classification tasks, with a symbolic vocabulary V of size 200 containing integer symbols $(0 \cdots |V|-1)$ and $k = 8$. The datasets each contain 400k sequences with a 99/1 split between training and testing. The description of the tasks and performance of the LSTM models are included in Table 4.1 and those of the Transformer models in Table 4.2. All models have a test accuracy of at least 98%, ensuring that the models truly generalize to the tasks.

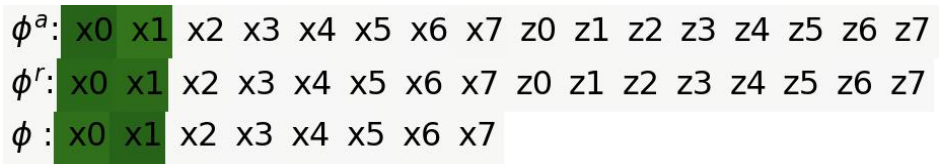
For each model, we compute global Shapley explanations [Covert et al., 2020], both order-insensitive(ϕ) and order-sensitive(ϕ^a, ϕ^r), with a uniform discrete distribution over all symbols for g. We use 5 random seeds for each explanation to control for the randomness from g and q (total of 25 random seeds per task). We compute ϕ, ϕ^a and ϕ^r on 1000 1-labeled sequences of the test-set of Task 1, $[\mathbf{W_1}]$. $[\mathbf{W_1}]$ satisfies the condition of "sequence beginning with a duplicate"(ex. $[11234567]$). We choose $f^y(x)$ as the difference between the predicted probability of the correct class ($y = 1$) and that of the incorrect class ($y = 0$), as used in previous chapters. Similar $f^y(x)$ is used in the rest of this chapter. Since the condition of Task 1 is sufficient for that of Task 2 and Task 3, Model 2 and Model 3 can also accurately classify $[\mathbf{W_1}]$, as shown by *Acc-1* of Table 4.1.



((a)) Model 1: Begins with duplicate



((b)) Model 2: Adjacent duplicate

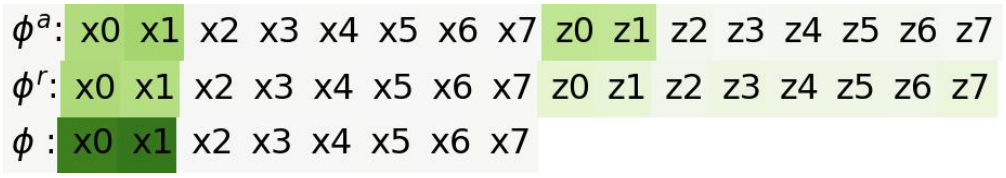


((c)) Model 3: Any duplicate

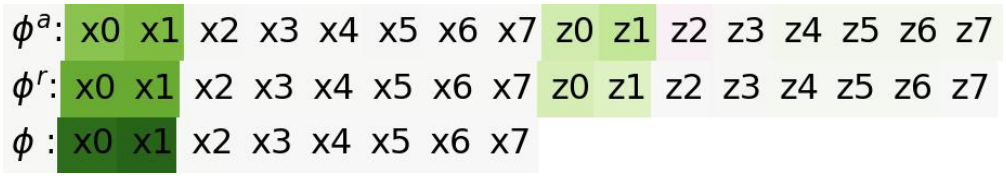
Figure 4.2: Explanations(ϕ^a, ϕ^r and ϕ) of $[\mathbf{W_1}]$ by three LSTM models solving tasks in Table 4.1. Positive explanations are green and Negative explanations are pink. Darker shade represents larger magnitude.

Result Figure 4.2 shows explanations of $[\mathbf{W_1}]$, for three models, respectively. Applying *SV* which only attributes to occurrence features x, we obtain identical explanations across three models: x_0 and x_1 are equally influential while other features have zero attribution. *OSV*, on the other hand, faithfully recovers the importance of sequence order by also attributing to index features z_0 and z_1 for Model 1 and 2, but not for Model 3.

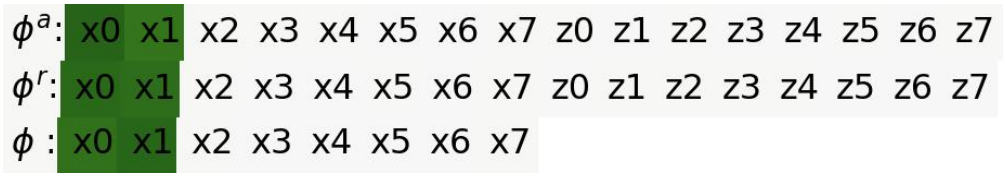
Varying q, we observe that q^r for Model 1 does not attribute exclusively to z_0 and z_1 as q^a does: z_0 and z_1 does not function as a 2-gram feature, as they are only influential in their respective absolute positions. Instead, the attributions distribute evenly to all order features, as intervening on any order feature almost always cause $[\mathbf{w}]_0$ and $[\mathbf{w}]_1$ to change positions. On the other hand, ϕ^r and ϕ^a are similar for Model 2, as $[\mathbf{w}]_0$ and $[\mathbf{w}]_1$ can be treated as a 2-gram feature, which is influential as long as their relative position is retained. The results for Transformer models are highly similar to Table 4.1 and Figure 4.2, which is depicted in Figure 4.3.



((a)) Model 1: Begins with duplicate



((b)) Model 2: Adjacent duplicate



((c)) Model 3: Any duplicate

Figure 4.3: Explanations(ϕ^a, ϕ^r and ϕ) of $[\mathbf{W_1}]$ by three Transformer models solving tasks in Table 4.2

	LSTM		Transformer	
	p_a	p	p_a	p
ϕ^a	0.97±0.02	0.99 ±0.01	0.96±0.03	0.99±0.01
ϕ^r	0.90±0.08	0.99±0.00	0.89±0.08	0.99±0.00
ϕ	0.76±0.16	0.99±0.01	0.76±0.16	0.99±0.01
$\phi_{saliency}$	-	-	0.40±0.54	0.35±0.61
ϕ^0_{IG}	-	-	0.05±0.36	0.55±0.55
ϕ^p_{IG}	-	-	0.12±0.27	0.52±0.61

Table 4.3: Correlations with the ground truth for various explanation methods.

Comparison with baselines Since our paper is the first to define attributions for order features, there are no model-agnostic baseline methods to directly compare with *OSV*. However, Transformer models separately encode orders with positional embeddings and occurrences of words with word/token embeddings [Vaswani et al., 2017], analogous to order features and occurrence features. As a result, we compare *OSV* with three baselines across two gradient-based methods, Saliency map [Simonyan et al., 2013] and Integrated Gradients (IG) [Sundararajan et al., 2017] by attributing to two embeddings separately. For IG, we use the zero baseline for the word embeddings and experiment with two baselines for position embeddings: (1) zero

embeddings (ϕ^0_{IG}); (2) a dynamic baseline (ϕ^p_{IG}) which randomly permutes the position ids used to create position embeddings (equivalent to shuffling a sequence), and iterate through all permutations or until it converges. For IG, we use 1000 as the number of steps for approximation(the authors of IG recommends 20 to 1000 steps).

To compare the explanations with the ground truth (Ex. $[1, 1, 0, 0, 0, 0, 0, 0]$ for both $\phi^a(x)$ and $\phi^a(z)$ for Model 1), we compute two Pearson correlations (1) p_a between ϕ, ϕ^r, ϕ^a and the ground truth(16 features). We set the missing $\phi(z)$ values of ϕ to be zero. (2) p between ϕs and the ground truth, for the occurrence features($\phi(x)$) alone(8 features). As shown in Table 4.3, explanations using *OSV* are more accurate to recover the order-sensitive ground truth (p_a) for both LSTM and Transformer models, while retaining the same accuracy for occurrence features (p). Neither ϕ^0_{IG} and ϕ^p_{IG} recovers attributions to both features or even to the occurrence features.

There are three potential reasons for the poor faithfulness of gradient-based methods, especially IG, for attributing to position embeddings: (1) Intervention using gradients such as $\phi_{saliency}$ is not a faithful intervention for "removing order features"; (2) Despite axiomatic properties of IG, a fixed baseline for positional embeddings result in invalid sequences while the dynamically permutated baseline does not easily converge; (3) Both IG baselines intervene continuously on the sparse positional embeddings space, resulting in potentially invalid sequence representations. Most importantly, not all models explicitly encode order features as positional embeddings, making IG fundamentally inapplicable for non-Transformer models such as RNN. In comparison, *OSV* is model agnostic.

Computational Efficiency Table 4.4 shows the average number of evaluations (number of intervened instances) per instance for computing global explanations shown in Figure 4.2 and 4.3. For sequences of length 8, ϕ^z needs less than 8 times more evaluation for convergence, which is comparably much slower than *SV*. However, considering *SV* disregards order features altogether, while IG with positional embeddings requires at least 1000 evaluations per instance and still fail to recover the true attributions to order features, *OSV* offers a reasonable middle ground. In this paper, we look at applications with shorter input length (around 10 to 20 words per input), and all experiments finish with a reasonable amount of time even with the tight convergence setting. For example, on average, each template in Section. 4.6.1 takes around 5 min.

	LSTM	Transformer
ϕ^a	2425	2299
ϕ^r	2485	2445
ϕ	330	331

Table 4.4: Average (across 25 seeds) number of evaluations per instance for synthetic experiments

4.6 Natural Language Experiment

In this section we apply *OSV* to a variety of natural language tasks. Specially, we show how *OSV* offers richer and deeper insights in explaining models' behavior compared to prior methods such as *SV*, and how order-insensitivity shown by *OSV* precisely uncovers model's weakness to potential adversarial manipulations.

4.6.1 Hans

Models and Data HANS Challenge set [McCoy et al., 2019] is constructed to test the syntactic understanding of NLI models against spurious heuristics of overlapping words between the premise and the hypothesis sentences. For example, the sentence *The doctors visited the lawyers* should not entail *The lawyers visited the doctors* even though they contain the same words. While BERT fine-tuned on the MNLI corpus [Williams et al., 2018] achieves high accuracy on its test set, its accuracy on the non-entailment subset of HANS is less than 20%. To understand the role of word order play in a task such as NLI, we apply *OSV* to the original BERT model fine-tuned on MNLI, M_{ori} and two models designed to be robust against HANS dataset:

- M_{aug} using data augmentation. Min et al. [2020] augments the training dataset of MNLI with instances in the form of " $A \not\rightarrow$ inverted A", such as *This small collection contains 16 El Grecos* $\not\rightarrow$ *16 El Grecos contain this small collection*, where A are the hypothesis sentences of the original MNLI corpus. M_{aug} scores 67% on HANS.

- M_{for} using BoW Forgettables. Yaghoobzadeh et al. [2021] retrains BERT on instances unlearnable by a simple Bag-of-Words model. It enables the model to focus on "forgettable examples" which interestingly contains instances with the same heuristics as those targeted by HANS. M_{for} scores 72% on HANS.

	$\sum_i \phi(x_i)$	$\sum_i \phi(z_i)$	Acc-HANS	Acc-HANS*
M_{ori}	1.01	-0.60	0.58	1.00
M_{aug}	0.36(↓0.65)	-0.58(↑0.02)	0.67	0.64
M_{for}	0.37(↓0.64)	-0.69(↓0.09)	0.72	0.94

Table 4.5: Global Statistics across all templates and accuracy on HANS & HANS*

Both models have exactly the same architecture (BERT-Base [Devlin et al., 2019]) as M_{ori}, and neither makes use of secondary datasets including HANS itself, making their explanations more comparable. For the original model, we use models available from the Huggingface [Wolf et al., 2019] repository and fine-tuned each model with 3 epochs using the default parameters of HuggingFace Trainer class. We use the official implementation of M_{aug}[1] and M_{for}[2]. We do not make extensive efforts to tune the hyperparameters since the goal of this paper is not to use the best models but to show the utility of an explanation device. Similar to Section. 4.5, we compute global explanations for all sentences of a same template, for all templates used to construct HANS, and intervene with a discrete uniform distribution of words at each template position.

Results Figure 4.4 shows explanations and accuracy for instances with template *The [Noun A] [Verb] the [Noun B]* $\xrightarrow{?}$ *The [Noun B] [Verb] the [Noun A]*, for three studied models, respectively. In the original model, we observe that the overlapping nouns exert large positive attributions ($\phi(x)$), driving the model towards the wrong direction of predicting entailment. Meanwhile, the small negative attributions from the order features ($\phi(z)$) are insufficient to counteract $\phi(x)$. In M_{aug} which scores 100% on this template, we observe that $\phi(x)$ shifts to negative, while $\phi(z)$ remains the same: the model pays no more attention to word orders compared to M_{ori}, other than simply reducing the correlations between overlapping words and entailment by shifting the decision boundary. In contrast, M_{for} produces more conceptually sound explanations: $\phi(z)$ almost doubles, while $\phi(x)$ is still positive, albeit smaller in magnitude. However, *SV* shows no such distinctions: both M_{aug} and M_{for} produces identical explanations, which may be misconstrued as both models successfully overcoming the spurious correlations.

We demonstrate more general quantitative result in Table 4.5, by computing the sum of $\phi(x)$ and $\phi(z)$ averaged across all non-entailment-labeled templates: while both M_{for} and M_{aug} suppress the effect of the

[1] https://github.com/Aatlantise/syntactic-augmentation-nli
[2] https://github.com/sordonia/hans-forgetting

Figure 4.4: Global explanations for instances with a template of HANS where nouns are swapped between the premise and the hypothesis, and the ground truth is non-entailment. Features suffixed by (x) and (z) are occurrence features and order features, respectively. Words in brackets ([]) are one example filling in variable template positions. Attributions are computed for $f^y(x) = p_{entailment} - p_{non-entailment}$, so positive attribution values indicate the features drive the model towards entailment while negative values indicates the features drive the model towards non-entailment. For easier visualization, attributions of *the* are combined with that of the following nouns.

overlapping words, M_{for} sees a sizable (15%) decrease of $\phi(z)$ in contrast to M_{aug}, which even increases a little. More result are included in Appendix 7.1.

In contrast to standard *SV* explanations, *OSV* shows that the more conceptually sound model M_{for} recognizes the difference in word orders between the premise and hypothesis as an indicator of non-entailment, while shifting of decision boundary without the true learning of order in M_{aug} may lead to overfitting [Lu et al., 2020c, Jha et al., 2020] and catastrophic forgetting [Kirkpatrick et al., 2017]. To test this hypothesis, we create a simple "adversarial" dataset, HANS*, by replacing the hypothesis

Dataset	DistilBERT	BERT	RoBERTa	StructBERT
REG	1.0	1.0	1.0	1.0
NEG	0.99	1.0	1.0	1.0
REG*(+)	0.85	0.85	1.0	1.0
REG*(-)	0.96	0.91	0.99	1.0
REG*,(+)	0.96	0.99	1.0	1.0
REG*,(-)	1.0	1.0	1.0	1.0
REG*B(+)	0.99	1.0	1.0	1.0
REG*B(-)	1.0	1.0	1.0	1.0

Table 4.6: Model performances of sentiment analysis for various datasets.

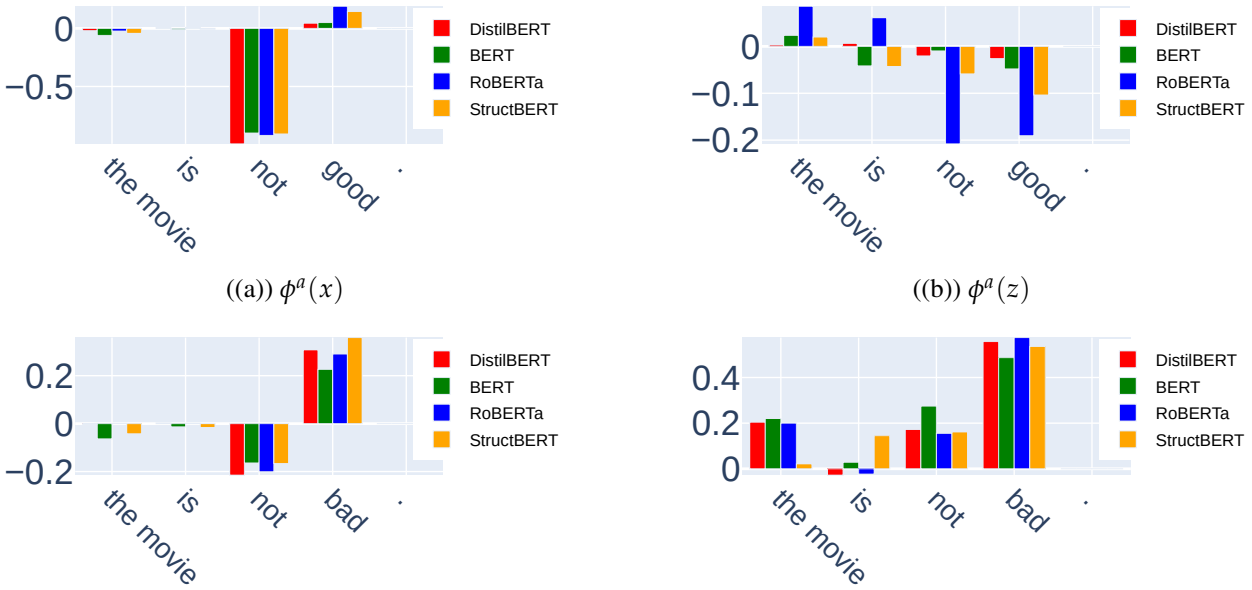


Figure 4.5: Explanations for two example sentences with negation across four models with $f^y(x) = p_{positive} - p_{negative}$. $\phi^a(x)$ and $\phi^a(z)$ are attributions to occurrence and order features, respectively.

in HANS with the same sentences as the premise, effectively creating instances such as *The doctors visited the lawyers.* $\xrightarrow{?}$ *The doctors visited the lawyers.*, for which the ground truth is always entailment. According to Table 4.5, M_{ori} scores 100% and M_{for} maintains a high accuracy of 94% on HANS*, while M_{aug} scores only 64%, corroborating the overfitting hypothesis.

4.6.2 Negation in Sentiment Analysis

Previous works find that word order hardly matters to Transformer-based sentiment analysis models. Mostly notably, Pham et al. [2021] discovers that "more than 60% of instances can be correctly predicted by the top-1 salient word." As a result, we focus on one syntactical relation that is both order-sensitive and essential for classifying sentiment: negation. In particular, we look at the construct of negation cues (ex. *no*, *not*, *never*) preceding an argument word/phrase (*not good* or *not a good movie*). Order is important in these constructs: for instance, *not* should only reverse the sentiment when directly preceding an argument.

Models and Data We evaluate four models, DistilBERT [Sanh et al., 2019], BERT-Base [Devlin et al., 2019], RoBERTa [Liu et al., 2019b], and StructBERT [Wang et al., 2019]. For StructBERT [Wang et al., 2019] and $M_{for}$of Section. 4.6.1, we use the implementation from the official repository[3]. All other models are trained on 2-class GLUE SST-2 sentiment analysis dataset [Wang et al., 2018] using the Huggingface API [Wolf et al., 2019]. Due to the lack of instances in SST-2 with the targeted negation construct, we construct a simple synthetic dataset to surgically evaluate the model's learning of negation. A total of 3072 sentences such as those in Figure 4.5 are generated from the template *[Det(The)] [Noun(movie)] [Verb(is)] [NEG/REG] [Adjective(good)][(Punc).]* where *[REG/NEG]* is evenly distributed between negation cues (ex.*not*) and regular adverbs (ex.*very*) or blank. The adjectives are evenly distributed between common positive and negative sentiment adjectives(See Appendix 7.2 for more details). All models perform really well on both the regular (REG) and negated (NEG) half of the dataset, as shown by the first two rows of Table 4.6.

Results Figure 4.5 shows local $OSV(\phi^a)$ with marginal intervention with a single token *[MASK]*, for all four models on two example instances. We observe an interesting distinction between the two: High $\phi(z)$ compared to $\phi(x)$ on *not* and *bad* across all models in Figure 4.5(c) indicates that it is essential for *not* to precede *bad* to correctly reverse the sentiment. However $\phi(z)$ on *not good* (Figure 4.5(a)), is comparably smaller in magnitude for all models, particularly for DistilBERT and BERT, where the attributions to order features are negligible. This discrepancy is mostly likely caused by the inherent polarity of the word *not*, hence models do not have to encode order for negating positive adjectives while order is essential for negating

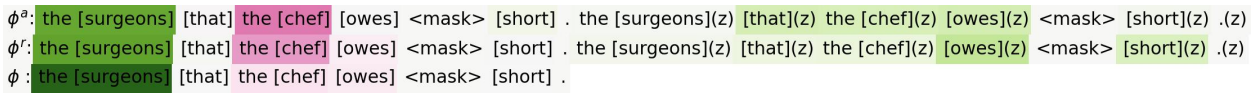
[3] https://github.com/alibaba/AliceMind

	SVA-Obj			SVA-Subj		
	$\Delta\phi^z_{su}$	Acc.*	Acc.	$\Delta\phi^z_{su}$	Acc.*	Acc.
DistilBERT	.04	.86(↓.05)	.91	.12	.79(↓.14)	.92
BERT-B	-.03	.93(↓.00)	.93	.11	.94(↓.04)	.99
BERT-L	-.05	.81(↓.00)	.81	.09	.92(↓.04)	.96
RoBERTa	-.04	.71(↓.00)	.71	.09	.64(↓.08)	.72

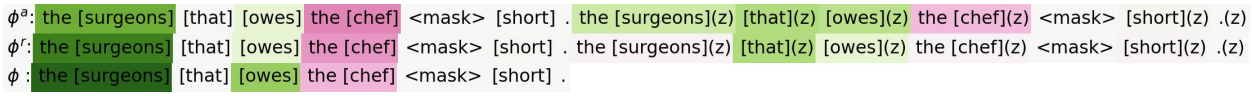
Table 4.7: Absolute-relative order discrepancy and Model performances for SVA.

negative adjectives. Similar polarity of negation cues has been discovered in Section. 3.5.2 of Chapter 3. One possible reason for the model-wise difference is that RoBERTa and StructBERT benefit from their improved training procedures over BERT/DistilBERT. StructBERT, for example, is specially designed to be sensitive to word orders during pretraining.

To demonstrate the potential weakness of such order insensitivity of BERT and DistilBERT, we construct an adversarial dataset (REG*) by appending one of five neutral phrases containing negation words(*not as expected*, *not like the other*, *not gonna lie*, *never gonna lie*, *never as expected*), to the end of sentences of REG, effectively creating sentences such as *The movie is good not gonna lie*, which happens to contain the flipped negation construct(*good not*) and *not* in this context should not convey any sentiment. This dataset of size 7680 is evenly distributed between negative and positive sentiments. The result is shown in Table 4.6. Mirroring the difference in order-sensitivity shown by $\phi(z)$, DistilBERT and BERT both suffer a 15% decrease for sentences with positive sentiments (REG*(+)) while the accuracy is either unchanged for RoBERTa and StructBERT, or decreases slightly for sentences with negative sentiments (REG*(-)) for all models. To further ensure the drop is not merely caused by the inherent negative sentiment of the word *not* (apart from the same $\phi(x)$ for *not* in Figure 4.5(a)), we test two alternative datasets: (1) REG*, inserts a comma between the appended phrase and the original sentences, ex.: *The movie is good, not gonna lie.* (2) REG*B appends the phrases to the beginning of the sentences instead. Interestingly, all four models are robust to both alternatives. This distinction implies that the models are capable of disregarding the added phrases if negation cues are not directly proximate to the adjective. ***OSV*, unlike *SV*, effectively surfaces local order-insensitivity in the negation construct, which is validated by BERT & DistilBERT's failure on REG*(+).**



((a)) SVA: across Object Relative Clause



((b)) SVA: across Subject Relative Clauses

Figure 4.6: Global explanations for two SVA tasks for DistilBERT. Words in brackets ([]) are one example filling in variable template positions. Features suffixed by (z) are order features. $f^y(x) = p_{correct\_verb} - p_{incorrect\_verb}$. Positive attributions are in green and negative in pink. Darker shade represents larger attribution magnitude.

4.6.3 Subject-Verb Agreement(SVA)

While fine-tuned classification models may not learn word order because they don't have to, a pretrained language model should, in theory, be more sensitive to order, especially when evaluated on syntactic tasks such as Subject-Verb Agreement. The task of SVA evaluates whether a language model prefers the correct verb form to match with the subject. For example, a pretrained masked language model should assign higher probability to *is* over *are* for the sentence *The cat <mask> cute*.

How language models learn SVA is widely studied in prior works [Linzen et al., 2016, Lu et al., 2020b, Wei et al., 2021]. In this section, we focus on two complicated long-range formulations of SVA where Transformers excel over simple RNN models [Goldberg, 2019]: SVA across subject relative clauses (*SVA-subj*) and SVA across object relative clauses (*SVA-obj*). For example, test instances for *SVA-obj* are generated from the template: *the [Subject] that the [Attractor] [Verb] <mask?(is/are)> [Adjective].*(ex.Figure 4.6(a)). A language model needs to correctly parse the sentence to identify the true subject. In particular, when *Attractor* is of an opposite number to the *Subject*, a model that fails to parse the sentence will be distracted by the more proximate attractor. Prior works find that Transformer models succeed in long-range SVA and overcome such distraction because they learn composite syntactic structures [Jawahar et al., 2019, Hewitt and Manning, 2019].

Models and Data We use dataset from Marvin and Linzen [2018] and the cloze-style set up used in Goldberg [2019]. Specifically, we look at sentences with oppositely numbered attractors. We evaluate four pretrained

langauge models: DistilBERT [Sanh et al., 2019], BERT-Base & BERT-Large [Devlin et al., 2019], and RoBERTa [Liu et al., 2019b]. All MLM models are directly sourced from the Huggingface API [Wolf et al., 2019]. The accuracy of each task is found in Table 4.7, where all models perform quite well except for RoBERTa, which still scores more than 70%.

Result Figure 4.6 show the global explanations for *SVA-Obj* and *SVA-Subj* with $f^y(x) = p_{correct\_verb} - p_{incorrect\_verb}$ and the same g as Section. 4.6.1. We observe that the subject word exerts positive attribution on the correct choice of the verb, and the intervening noun (attractor) exerts negative attribution. In contrast to fine-tuned classification models such as NLI and SST, attributions to order features are high: they are indeed more important for syntactical tasks. Zooming in on the explanations, we obtain two other findings:

First, we observe that the relativizer word *that* is important as an order feature, but not as an occurrence feature. In other words, in order to predict the correct verb form, *that* itself may not contain the number signal, nevertheless it is regarded by the language model as an essential syntactic boundary.

Second, there is a big difference ($\Delta\phi^z_{su}$) between ϕ^a and ϕ^r for the subject word in *SVA-Subj.*, but not for *SVA-Obj.*, i.e, the model seems to rely on absolute position of the subject word in *SVA-Obj.* much more than the relative position of the subject word. ϕ^a of the attractor is also much more negative than ϕ^r, indicating the model is distracted by the proximity of the attractor words (along with their occurrences). These distinctions point to a hypothesis that even though they contain exactly the same words, the model parses *SVA-Obj.* better than *SVA-Subj.*, most likely due to the proximity of the attractor with <*mask*> in *SVA-Subj*. **The higher accuracies of *SVA-Obj.* may be attributed to the models' reliance on the absolute position of the subject words in the beginning of sentences, surfaced by the two modes of intervention on order features(q^a & q^r in Section. 4.3.3). Without attributing to order features, however, *SV* cannot uncover this subtle discrepancy.**

To test this hypothesis, we create a simple adversarial set by prepending each of 23 punctuation and symbols[4] such as "." or <unk> to the beginning of sentences and see if such shifting of the subject from its absolute position impact the predictions. In Table 4.7, Acc* shows all models suffer more from this adversarial perturbation for *SVA-Subj* than *SVA-Obj*, mirroring the distinction from $\Delta\phi^z_{su}$. This indicates that the models' learning of long-range SVA is not always conceptually sound, as is in the case of *SVA-Subj.*.

[4] !"#&'*+,-./:<=>?@^_|~;<unk>

4.7 Related Work and Discussion

Recent works [Pham et al., 2021, Sinha et al., 2021a, Clouatre et al., 2021, Alleman et al., 2021, Sinha et al., 2021b, Gupta et al., 2021] show NLP models' insensitivity to word orders. Our work corroborates such insensitivity by showing often insignificant attributions to order features in classification models. Meanwhile, a systematic explanation device such as *OSV* offers more precise and measurable attributiosn to the importance of order.

Shapley Values [Shapley et al., 1953] is widely adopted as an model-agnostic explanation method [Strumbelj and Kononenko, 2010, Datta et al., 2016, Lundberg and Lee, 2017, Sundararajan and Najmi, 2020, Covert et al., 2020] and also applied to explain text classification models [Chen and Jordan, 2020, Zhang et al., 2021, Chen et al., 2018]. However, the explanation power of these methods is limited by treating text data as tabular data. *OSV*, as demonstrated by numerous examples, allows us inspect the effect of "word occurrences" along side the effect of "word orders". As order sensitivity is an essential metric of conceptual soundness, explanations incorporating order are more faithful in reflecting the true behavior of models.

We believe that explanations should not only function as a verification for plausibility [Jacovi and Goldberg, 2020], but also as a tool for diagnosing conceptual soundness. **Instead of *confirming* the human understanding or the correct linguistic rules, explanations are equally, if not more, insightful when *deviating* from them.** One popular approach to surface such deviation is adversarial examples [Zhang et al., 2020], and many algorithms for finding them in NLP tasks [Cheng et al., 2020, Ebrahimi et al., 2018, Wallace et al., 2019] use gradient-based attributions such as saliency maps to guide adversarial perturbations. However, the use of orderless explanations is limited to local perturbations, such as swapping words into their synonyms. To allow for broader and non-local testing of model robustness, challenge datasets [Belinkov et al., 2017, Ribeiro et al., 2020, Wu et al., 2021, McCoy et al., 2019, Naik et al., 2018] are constructed to stress test NLP models. These tests are becoming more crucial in designing and evaluating the conceptual soundness and robustness of NLP models beyond benchmark performances. However, rarely is the creation of those datasets guided by explanations methods, making it difficult to systematically and deductively discover non-local adversarial examples. In this paper, we show an initial attempt to bridge the gap between the two: we show that *OSV* not only informs on *which*, but *how* adversarial perturbations break models, enabling a deeper understanding of model weaknesses than orderless explanations. More broadly, the findings in

Section. 4.6 echos Lovering et al. [2020] and Mangalam and Prabhu [2019], which discover that models learn "easier features"(e.g. occurrence features in HANS, *not* in SA and absolute order features in SVA) before learning "harder features" that may actually be the conceptually sound ones.

4.8 Conclusion

We propose *OSV* for explaining word orders of NLP models. We show how *OSV* is an essential extension of Shapley values and introduce two mechanisms for intervening on word orders. We highlight how *OSV* can precisely pinpoint if, where and how a model uses word order. Using adversarial examples guided by *OSV*, we demonstrate how order insensitivity, harmless as it seems, result in conceptually unsound models.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis, we use explanations to understand and evaluate NLP models and applications. Our approaches explore two essential directions: how information flows from input to output in sequence models, and how word orders influence the predictions of textual data. We explore both discrete Shapley values and gradient-based explainability frameworks and propose essential extensions within frameworks for explaining NLP data and model architectures. We make sure our explanations are faithful to the models true behavior, efficient in computation, and interpretable and succinct in its presentation to potential end users.

These explanations serve to enhance deeper understanding of NLP models and surface conceptual unsoundness issues in capturing grammatical, syntactic or semantic concepts. For example, we discover that neither RNN nor Transformer models learn the full scope of syntactic structures necessary for long-range agreement tasks, or the order-sensitive features to overcome the spurious correlations in HANS, a challenge set for natural language inference.

Our generated explanations and insights pave way for future research effort to address these weaknesses or to uncover more, to apply and expand the proposed frameworks for future applications, and to inspire the building of reliable, transparent and robust models.

5.2 Future Work

In this section, we discuss three directions for future work. First, we can create more faithful and interpretable explanations for more complicated NLP concepts and applications; Second, we can use explanations to inform and guide model improvements for NLP applications; third, we can leverage explainability research in NLP to make real and tangible social impacts and to address outstanding ethical issues.

5.2.1 More Faithful Explanations for NLP

Distributed representations The generated explanations in Chapter 2 and 3 described highly concentrated and sparse abstractions such as a singular path with two neurons for capturing SVA in RNN models. However, we speculate that some concepts may require distributed, compositive or interactive representations. Although our methods can be potentially extended to account for the trade off between sparsity and faithfulness such as including more nodes per layer in the GPR algorithm (Section. 3.3), more sophisticated abstraction and interpretation mechanisms may be required to explain more complicated linguistic concepts.

Intrinstic interpretability In contrast to post-hoc analysis, recent works have began to explore mechanistic interpretability which aims to *reverse engineer* the detailed computations of each model component, especially attention modules [Chiang and Cholak, 2022, Weiss et al., 2021, Kim et al., 2021, Elhage et al., 2021]. In addition, representational capacities of LSTM and attention modules are studied in the context of approximating finite state machines [Ross et al., 2021] or learning formal languages [Bhattamishra et al., 2020]. In both categories of intrinsic approaches, explainability comes *a priori*: if one proactively understands how and to what extent a given model architecture and parameters operate even before plugging in the training data, then the model is intrinsically interpretable.

Exciting and promising as this research direction is, intrinsic approaches have yet completely incorporated the training dynamics of natural language models, especially for pretrained transformer models which are often trained on large text corpus. Thus a complementary post-hoc analysis may still be necessary to analyze real-world models until we completely decipher the whole training pipeline from static model architectures to learning dynamics.

Similar argument can be made for explainability methods that explains the data rather than the model,

including tracing influence back to the training data [Yeh et al., 2018, Koh and Liang, 2017, Guo et al., 2021], or explanation through counterfactual examples [Ross et al., 2021, Wu et al., 2021, Jacovi et al., 2021]. However, it is often equally important to directly surface and address the conceptual soundness issues hidden in the training data such as spurious correlations [Lovering et al., 2020] or distribution shift [Michel, 2021].

5.2.2 From explanations to model improvement

Sparse or distilled models As transformer models evolve in size to the scale of trillions of parameters [Fedus et al., 2021], so does environmental cost of training, maintaining and deploying them. In Chapter 2 and 3, we discovered that many linguistic concepts are sparsely encoded in both RNN and Transformer models, which brings up the question, can we construct equivalent but much smaller models without compromising utility? Recent work on compressing Transformer models employ optimization techniques such as model distillation [Hinton et al., Tsai et al., 2019, Prasanna et al., 2020, Sanh et al., 2019, Jiao et al., 2019] or sparse attention modules [Zaheer et al., 2020, Wang et al., 2020a, Tay et al., 2020, Child et al., 2019]. However, how to use model abstractions generated from explanations to inform on more systematic training of smaller models in the first place remains an exciting direction for future work.

Training robust NLP models A natural next step from this thesis is to tackle the various model weaknesses and conceptual unsoundness uncovered by our explanations. The analysis in Section. 4.6.2 of Chapter 4 on StructBERT [Wang et al., 2019], for instance, surfaced how explanations justify certain modeling choices: training with order-sensitivity in StructBERT indeed facilitates order sensitivity for downstream applications. However, what remains to be an essential path forward for further work is to make more tangible connections between explainability and robustness in NLP. Such connections have already been explored in image data [Datta et al., 2021, Wang et al., 2020b], where robust models are found to generate more interpretable explanations. However, due to discrete input space and more nuanced, non-local definitions of robustness (HANS example in Section. 4.6.1 of Chapter 4), similar connections have yet to be established in a meaningful way.

5.3 Social Impacts of NLP

In this paper, we focus on conceptual unsoundness issues in low-stake settings. However, as NLP is applied for real-world applications such as hiring decisions [Raghavan et al., 2020], there is a stronger call for NLP models to become accountable, explainable and fair. Most importantly, many ethical concerns such as gender or racial bias [Lu et al., 2020c, Blodgett et al., 2020, Font and Costa-jussà, 2019, Zhao et al., 2019, Manzini et al., 2019, May et al., 2019] and privacy [Carlini et al., 2021] have been brought to attention, often uncorrelated with their benchmark performances. Going forward, how explainability methods can help uncover and address these issues in real-world high-stake decision-systems, and how explanations can be presented efficiently and effectively to end users should be valued equally to generating faithful and accurate explanations themselves.

Chapter 6

Appendices for Influence Patterns for Explaining Transformer Models

6.1 Proof of Proposition 1

Proposition 1 (Chain Rule) $\mathcal{I}(\mathbf{x}, \pi) = \mathbb{E}_{\mathbf{z}\sim\mathcal{D}(\mathbf{x})} \prod_{i=1}^{-1} \frac{\partial \pi_i(\mathbf{z})}{\partial \pi_{i-1}(\mathbf{z})}$ *for any distribution* $\mathcal{D}(\mathbf{x})$.

We first show that for a pattern $\pi = [\pi_1, \pi_1, ..., \pi_k, ..., \pi_n]$ the following equation holds:

$$\frac{\partial \pi_n}{\partial \pi_1} = \frac{\partial \pi_n}{\partial \pi_k} \frac{\partial \pi_k}{\partial \pi_1} \tag{6.1}$$

Proof: Let $\gamma(\pi_1 \to \pi_n)$ be a set of paths that the pattern π abstracts. Therefore, we have

$$\frac{\partial \pi_n}{\partial \pi_1} = \sum_{p \in \gamma(\pi_1 \to \pi_n)} \prod_{i=1}^{-1} \frac{\partial p_i}{\partial p_{i-1}} \tag{6.2}$$

Similarly, we have

$$\frac{\partial \pi_k}{\partial \pi_1} = \sum_{p \in \gamma(\pi_1 \to \pi_k)} \prod_{i=1}^{-1} \frac{\partial p_i}{\partial p_{i-1}}, \quad \frac{\partial \pi_n}{\partial \pi_k} = \sum_{p \in \gamma(\pi_k \to \pi_n)} \prod_{i=1}^{-1} \frac{\partial p_i}{\partial p_{i-1}} \tag{6.3}$$

Suppose $|\gamma(\pi_1 \to \pi_k)| = N_1$, $|\gamma(\pi_k \to \pi_n)| = N_2$ and we denote $\prod_{i=1}^{-1} \frac{\partial p_i^{(j)}}{\partial p_{i-1}}$ as path gradient flowing from the path j. Therefore,

$$\frac{\partial \pi_n}{\partial \pi_1} = \sum_{p \in \gamma(\pi_1 \to \pi_n)} \prod_{i=1}^{-1} \frac{\partial p_i}{\partial p_{i-1}} \tag{6.4}$$

$$= \prod_{i=k}^{-1} \frac{\partial p_i^{(1)}}{\partial p_{i-1}} \cdot \prod_{i=1}^{k} \frac{\partial p_i^{(1)}}{\partial p_{i-1}} + \prod_{i=k}^{-1} \frac{\partial p_i^{(1)}}{\partial p_{i-1}} \cdot \prod_{i=1}^{k} \frac{\partial p_i^{(2)}}{\partial p_{i-1}} + ... + \prod_{i=k}^{-1} \frac{\partial p_i^{(1)}}{\partial p_{i-1}} \cdot \prod_{i=1}^{k} \frac{\partial p_i^{(N_1)}}{\partial p_{i-1}} \tag{6.5}$$

$$+ \prod_{i=k}^{-1} \frac{\partial p_i^{(2)}}{\partial p_{i-1}} \cdot \prod_{i=1}^{k} \frac{\partial p_i^{(1)}}{\partial p_{i-1}} + \prod_{i=k}^{-1} \frac{\partial p_i^{(2)}}{\partial p_{i-1}} \cdot \prod_{i=1}^{k} \frac{\partial p_i^{(2)}}{\partial p_{i-1}} + ... + \prod_{i=k}^{-1} \frac{\partial p_i^{(2)}}{\partial p_{i-1}} \cdot \prod_{i=1}^{k} \frac{\partial p_i^{(N_1)}}{\partial p_{i-1}} \tag{6.6}$$

$$+ ... \tag{6.7}$$

$$+ \prod_{i=k}^{-1} \frac{\partial p_i^{(N_2)}}{\partial p_{i-1}} \cdot \prod_{i=1}^{k} \frac{\partial p_i^{(1)}}{\partial p_{i-1}} + \prod_{i=k}^{-1} \frac{\partial p_i^{(N_2)}}{\partial p_{i-1}} \cdot \prod_{i=1}^{k} \frac{\partial p_i^{(2)}}{\partial p_{i-1}} + ... + \prod_{i=k}^{-1} \frac{\partial p_i^{(N_2)}}{\partial p_{i-1}} \cdot \prod_{i=1}^{k} \frac{\partial p_i^{(N_1)}}{\partial p_{i-1}} \tag{6.8}$$

$$= \sum_{j}^{N_2} (\prod_{i=k}^{-1} \frac{\partial p_i^{(j)}}{\partial p_{i-1}} \cdot \sum_{m}^{N_1} \prod_{i=1}^{k} \frac{\partial p_i^{(m)}}{\partial p_{i-1}}) \tag{6.9}$$

$$= \sum_{j}^{N_2} (\prod_{i=k}^{-1} \frac{\partial p_i^{(j)}}{\partial p_{i-1}}) \cdot \sum_{m}^{N_1} (\prod_{i=1}^{k} \frac{\partial p_i^{(m)}}{\partial p_{i-1}}) \tag{6.10}$$

$$= \sum_{p \in \gamma(\pi_1 \to \pi_k)} \prod_{i=1}^{-1} \frac{\partial p_i}{\partial p_{i-1}} \cdot \sum_{p \in \gamma(\pi_k \to \pi_n)} \prod_{i=1}^{-1} \frac{\partial p_i}{\partial p_{i-1}} \tag{6.11}$$

$$= \frac{\partial \pi_n}{\partial \pi_k} \frac{\partial \pi_k}{\partial \pi_1} \tag{6.12}$$

Now we prove Proposition 1.

$$\mathcal{I}(\mathbf{x}, \pi) = \sum_{p \in \gamma(\pi)} \mathbb{E}_{\mathbf{z} \sim \mathcal{D}(\mathbf{x})} \prod_{i=1}^{-1} \frac{\partial p_i(\mathbf{z})}{\partial p_{i-1}(\mathbf{z})} \tag{6.13}$$

$$= \mathbb{E}_{\mathbf{z} \sim \mathcal{D}(\mathbf{x})} \sum_{p \in \gamma(\pi)} \prod_{i=1}^{-1} \frac{\partial p_i(\mathbf{z})}{\partial p_{i-1}(\mathbf{z})} \tag{6.14}$$

$$= \mathbb{E}_{\mathbf{z} \sim \mathcal{D}(\mathbf{x})} \prod_{i=1}^{-1} \frac{\partial \pi_i(\mathbf{z})}{\partial \pi_{i-1}(\mathbf{z})} \tag{6.15}$$

6.2 Guided Pattern Refinement

6.2.1 Optimality of GPR

The definition of pattern influence does not allow for polynomial-time searching algorithms such as dynamic programming. (Such an algorithm is possible for simple gradients/saliency maps but not for integrated gradients due to the expectation sum over the multiplication of Jacobians along all edges). As for the

optimality of the polynomial-time greedy algorithm, we hereby include a statistical analysis by randomly sampling 1000 alternative patterns for 100 word patterns in the *SVA-obj* task and sentiment analysis task (SST2). We also performed a t-test with the null hypothesis of a random pattern larger or equal than the pattern computed using GPR. The pattern influence of those random paths shows that the patterns extracted GPR are (1) more influential than 999.96 and 1000 (all) random patterns, averaged across all 100 word patterns evaluated for embedding-level attention-level patterns, respectively for *SVA-obj*; the same holds for sentiment analysis (1000 and 1000); (2) The extracted pattern influences are statistically significant assuming all randomly sampled patterns' influences follow a normal distribution, with $p = 2\text{e}{-10}$ for and $p = 0$ for embedding-level and attention-level patterns, respectively for *SVA-obj*; the same holds for sentiment analysis ($p = 0$ for and $p = 0$). In other words, the pattern influence values are far more significant than a random pattern, also confirmed by the high concentration values shown in Section. 3.5.4.

6.3 More Visualizations of Patterns

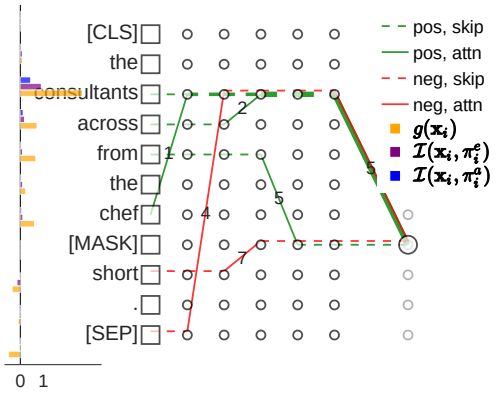
6.3.1 Example Visualizations

Figure 6.1, 6.2, 6.3 shows similar attractor examples from Figure 3.3(a) and 3.3(b), in three other evaluated subtask: SVA-Subj, SVA-APP, RA-NA. We observe similar discrepancies between SP and PS within each subtask, with `that`, and `across` and `that` functioning as attractors, respectively. Figure 6.4 through 6.8 show example patterns of actual sentences in the SST-2 dataset.

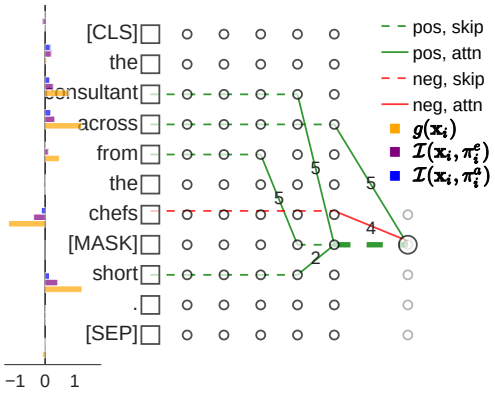
((a)) SVA-Subj, PS case

((b)) SVA-Subj, SP case

Figure 6.1: Examples of SVA-Subj

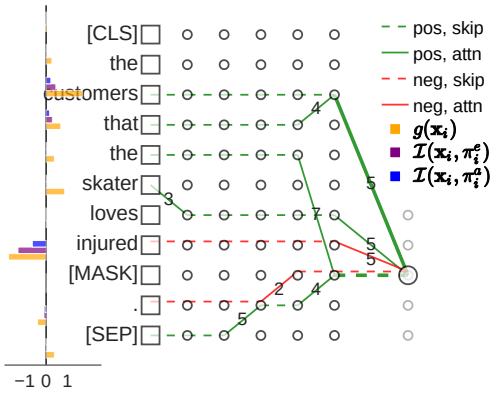


((a)) SVA-APP, PS case

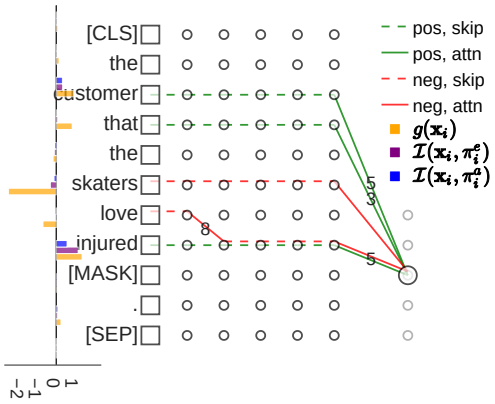


((b)) SVA-APP, SP case

Figure 6.2: Examples of SVA-APP



((a)) RA-NA, PS case



((b)) RA-NA, SP case

Figure 6.3: Examples of RA-NA

6.3.2 Aggregated Visualizations

In this section, we show the aggregated visualization (Fig. 6.9 and 6.10) across all examples of each case in two subtasks (SVA-Obj & NA-GA) by superimposing the patterns of individual instances (e.g. Figure 3.3(a) and 3.3(b)), while adjusting the line width to be proportional to the frequency of flow across all examples. The words within parenthesis represent one instance of the word in that position. The aggregated graphs verify (1) generality of patterns across examples in each case (including SS and PP). (2) a more intuitive visualization of the pattern entropy in these two tasks, with RA-NA showing "messier" aggregated patterns, or larger entropy.

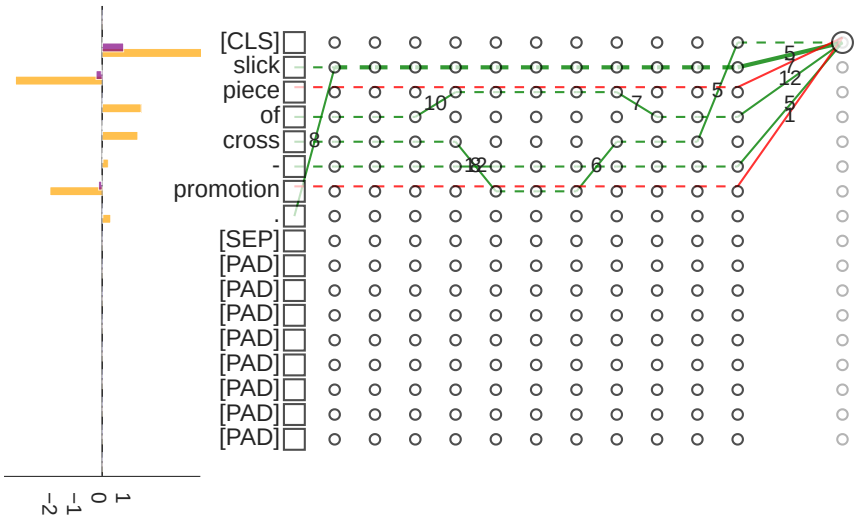


Figure 6.4: Example pattern of a positive sentence in SA

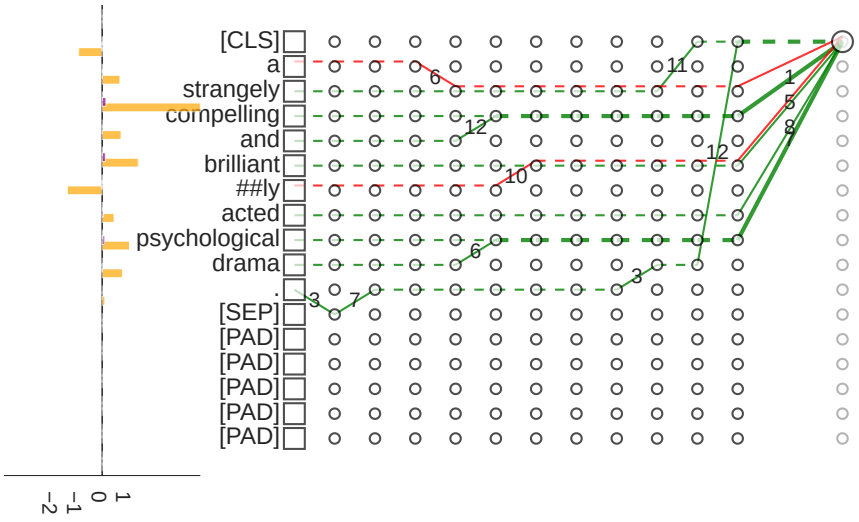


Figure 6.5: Example pattern of a positive sentence in SA

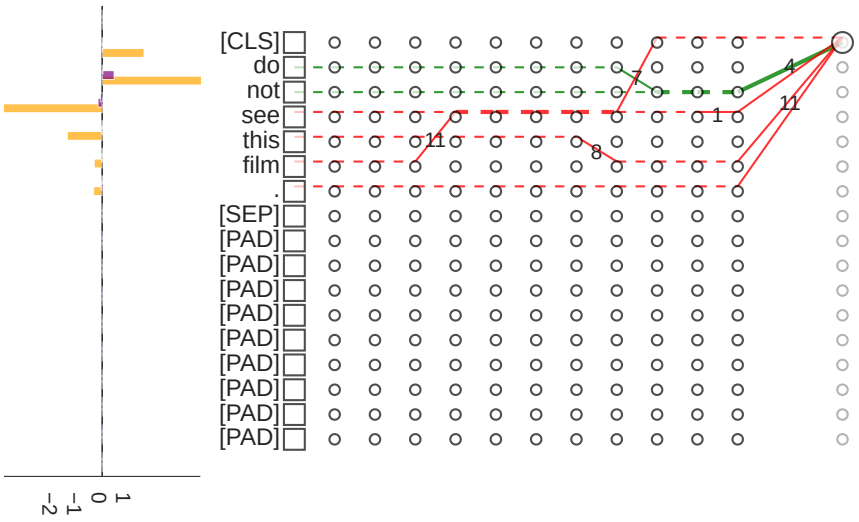


Figure 6.6: Example pattern of a negative sentence in SA

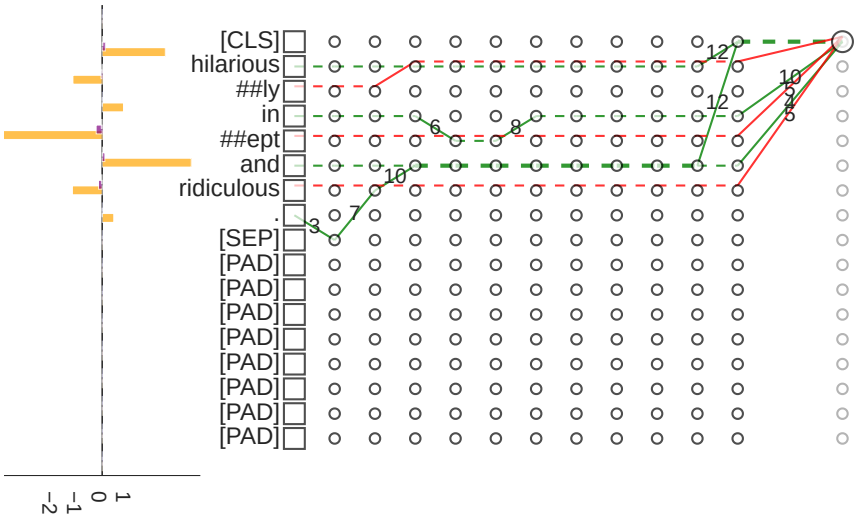


Figure 6.7: Example pattern of a positive sentence in SA

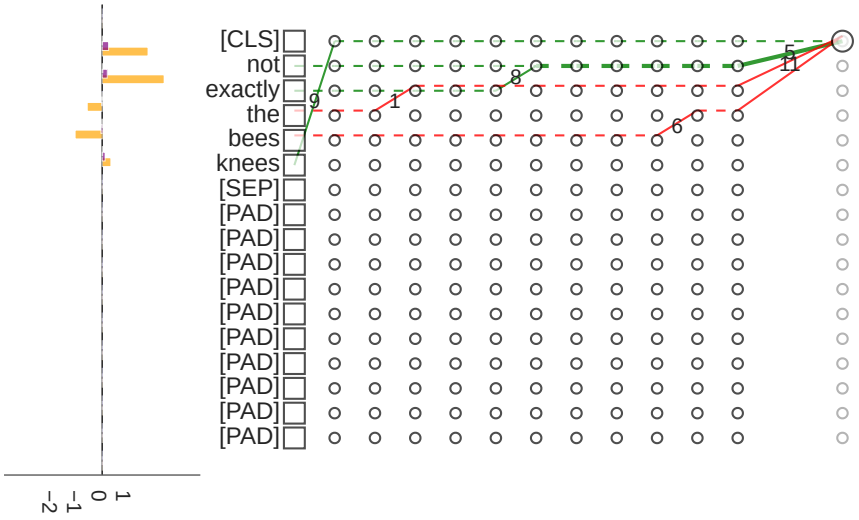
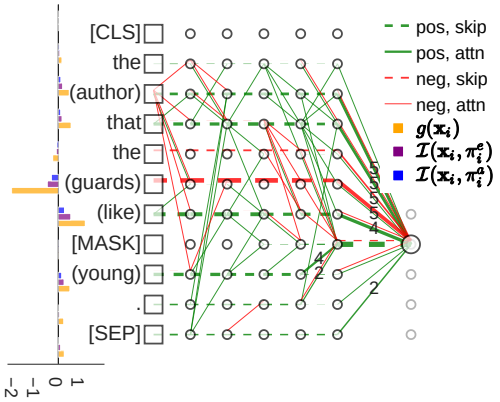
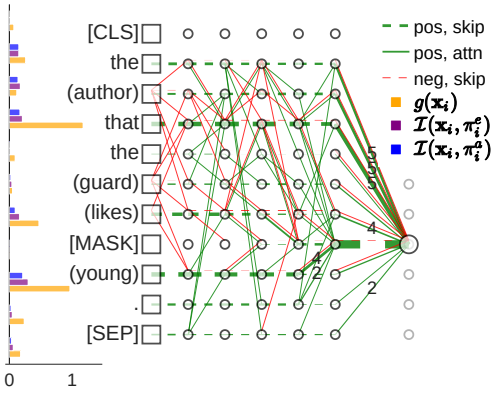


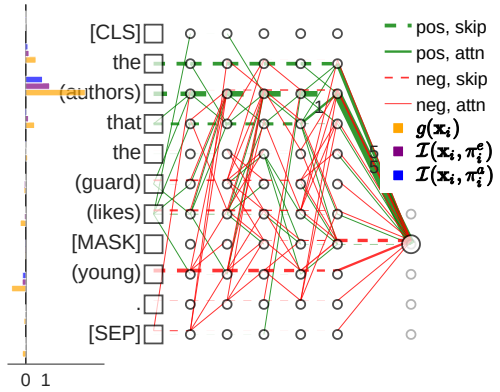
Figure 6.8: Example pattern of a negative sentence in SA



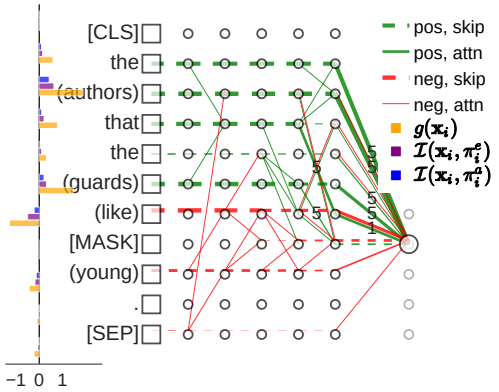
((a)) singular subject + plural intervening noun(SP)



((b)) singular subject + singular intervening noun(SS)

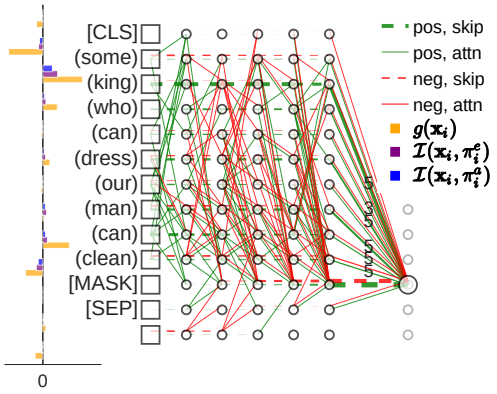


((c)) plural subject + singular intervening noun(PS)

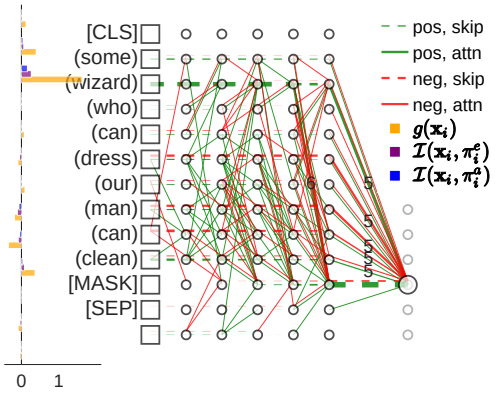


((d)) plural subject + plural intervening noun(PP)

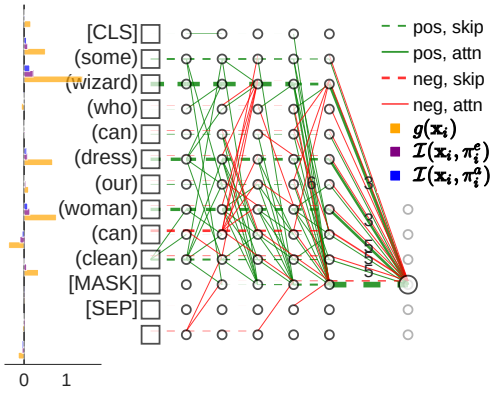
Figure 6.9: SVA-Obj. Aggregated



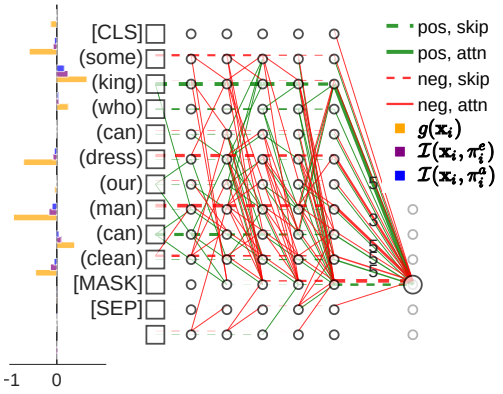
((a)) male subject + male intervening noun(MM)



((b)) female subject + male intervening noun(FM)



((c)) female subject + female intervening noun(FF)



((d)) male subject + female intervening noun(MF)

Figure 6.10: RA: GA, Aggregated

Chapter 7

Appendices for Order-sensitive Shapley Values for Evaluating Conceptual Soundness of NLP Models

7.1 Additional Results for HANS (Section. 4.6.1)

We compute the global explanations for all templates in [McCoy et al., 2019], where half of the templates is labeled as entailment. For those templates, word overlapping heuristics is actually the correct heuristic for predicting entailment. Table 7.1 shows the same stats as Table 4.5, but for entailment-labeled templates. As we can see, M_{for} is actually less order-sensitive than M_{aug}. Ideally, a conceptually sound model should learn both word overlaps and word orders; nevertheless this discrepancy can also be explained by overfitting and mere shifting of decision boundary: M_{aug} only learn that word overlap at the corresponding positions does not equate to entailment, making the model more order-sensitive for entailment-labeled sentences. While M_{for} also rely on word-overlapping features much more than order, but possibly in the same way as M_{ori}.

	$\sum_i \phi(x_i)$	$\sum_i \phi(z_i)$
M_{ori}	1.26	-0.37
M_{aug}	0.85(↓0.41)	-0.09(↑0.28)
M_{for}	0.89(↓0.37)	-0.22(↑0.15)

Table 7.1: Global Statistics across entailment-labeled templates

7.2 Additional Details for SA (Section. 4.6.2)

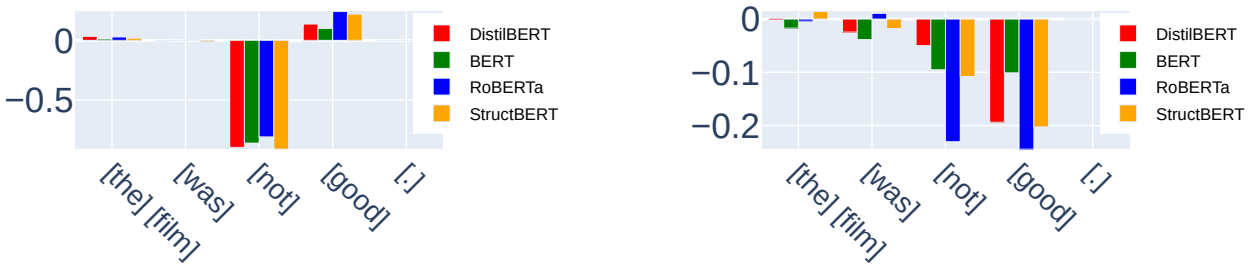
7.2.1 Templates for NEG/REG

[Det(The)] [Noun(movie)] [Verb(is)] [NEG/REG] [Adjective(good)][(Punc).]

- Det: 'the', 'that', 'this', 'a'
- Noun: 'film', 'movie', 'work', 'picture'
- Verb: 'is', 'was'
- NEG: 'not', 'never', 'not that', 'never that'
- REG: "(blank), 'very', 'pretty', 'quite'
- Adjective: (+): 'good', 'amazing', 'great'
- Adjective: (-): 'boring', 'bad', 'disappointing'
- Punctuations: '.', '!'

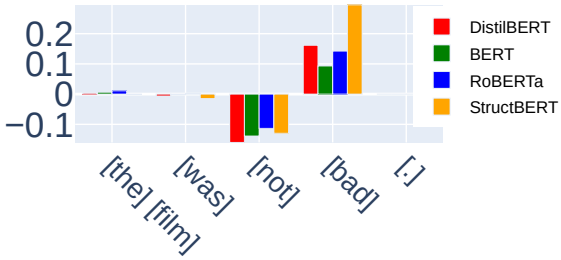
7.2.2 Aggregated Results for Figure 4.5

We compute the aggregated global explanations for all instances of NEG(+) and NEG(-), illustrated in Figure 7.1, to make sure the trend shown in Figure 4.5 represent the general trend. In Figure 7.2, we show local explanations for a sentence in REG with no special construct like negation, and as expected, word orders do not matter at all.

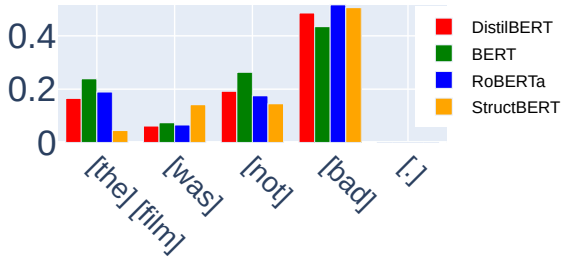


((a)) $\phi^a(x)$

((b)) $\phi^a(z)$

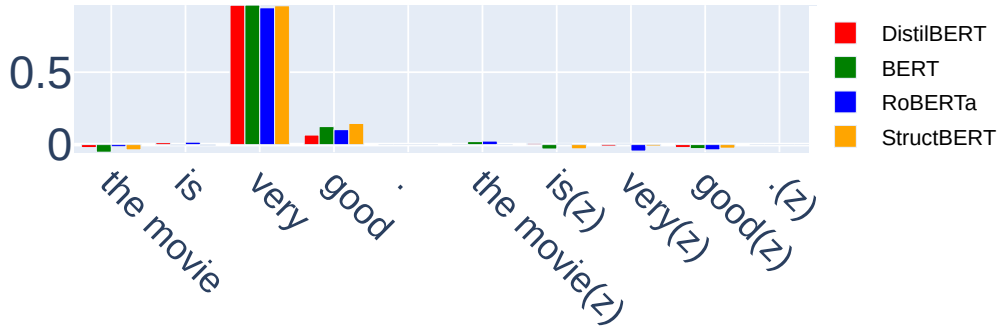


((c)) $\phi^a(x)$

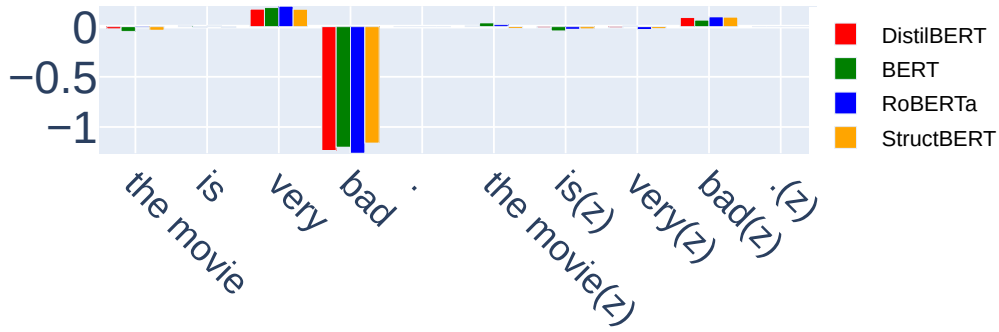


((d)) $\phi^a(z)$

Figure 7.1: Global explanations for NEG(+) and NEG(-), annotations follow Figure 4.5



((a)) ϕ^a for positive adjectives



((b)) ϕ^a for negative adjectives

Figure 7.2: Local explanations for two examples in REG

Bibliography

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org. 35

Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, 2020. 5, 25, 26, 28, 31, 32, 34, 43

Matteo Alleman, Jonathan Mamou, Miguel A Del Rio, Hanlin Tang, Yoon Kim, and SueYeon Chung. Syntactic perturbations reveal representational correlates of hierarchical phrase structure in pretrained language models. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 263–276, 2021. 5, 45, 52, 67

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Sy21R9JAW`. 40

Marco Ancona, Cengiz Oztireli, and Markus Gross. Explaining deep neural networks with a polynomial

time algorithm for shapley value approximation. In *International Conference on Machine Learning*, pages 272–281. PMLR, 2019. 54

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015. 11, 16

John F Banzhaf III. Weighted voting doesn't work: A mathematical analysis. *Rutgers L. Rev.*, 19:317, 1964. 51

Jeremy Barnes, Erik Velldal, and Lilja Øvrelid. Improving sentiment analysis with multi-task learning of negation. *Natural Language Engineering*, 27(2):249–269, 2021. 38

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 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, 2017. 3, 67

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, 2021. 1

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the ability and limitations of transformers to recognize formal languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7096–7116, 2020. 70

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. Language (technology) is power: A critical survey of "bias" in nlp. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, 2020. 72

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. On identifiability in transformers. In *International Conference on Learning Representations*, 2020. 43

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021. 72

Amnon Catav, Boyang Fu, Yazeed Zoabi, Ahuva Libi Weiss Meilik, Noam Shomron, Jason Ernst, Sriram Sankararaman, and Ran Gilad-Bachrach. Marginal contribution feature importance-an axiomatic approach for explaining data. In *International Conference on Machine Learning*, pages 1324–1335. PMLR, 2021. 48

Hugh Chen, Joseph D Janizek, Scott Lundberg, and Su-In Lee. True to the model or true to the data? *arXiv preprint arXiv:2006.16234*, 2020. 48

Jianbo Chen and Michael Jordan. Ls-tree: Model interpretation when the data are linguistic. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3454–3461, 2020. 3, 38, 67

Jianbo Chen, Le Song, Martin J Wainwright, and Michael I Jordan. L-shapley and c-shapley: Efficient model interpretation for structured data. In *International Conference on Learning Representations*, 2018. 3, 67

Minhao Cheng, Jinfeng Yi, Pin-Yu Chen, Huan Zhang, and Cho-Jui Hsieh. Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3601–3608, 2020. 67

David Chiang and Peter Cholak. Overcoming a theoretical limitation of self-attention. *arXiv preprint arXiv:2202.12172*, 2022. 70

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019. 71

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019. 3, 25, 43

Louis Clouatre, Prasanna Parthasarathi, Amal Zouaq, and Sarath Chandar. Demystifying neural language models' insensitivity to word-order. *arXiv preprint arXiv:2107.13955*, 2021. 5, 45, 67

Ian Covert, Scott M Lundberg, and Su-In Lee. Understanding global feature contributions with additive importance measures. *Advances in Neural Information Processing Systems*, 33, 2020. 3, 46, 48, 54, 55, 67

Ian Covert, Scott Lundberg, and Su-In Lee. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22(209):1–90, 2021. 47

P. Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. In *NIPS*, 2017. 40

Anupam Datta, Shayak Sen, and Yair Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, pages 598–617. IEEE, 2016. 3, 8, 46, 48, 50, 53, 54, 67

Anupam Datta, Matt Fredrikson, Klas Leino, Kaiji Lu, Shayak Sen, and Zifan Wang. Machine learning explainability and robustness: Connected at the hip. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 4035–4036, 2021. 40, 71

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, 2019. 1, 25, 26, 34, 45, 47, 60, 63, 66

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. Eraser: A benchmark to evaluate rationalized nlp models. *Transactions of the Association for Computational Linguistics*, 2020. 40

Kedar Dhamdhere, Mukund Sundararajan, and Qiqi Yan. How important is a neuron? In *International Conference on Learning Representations*, 2018. 5, 16, 26, 31, 32, 33, 41

Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: pure attention loses rank doubly exponentially with depth. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning*

Research, pages 2793–2803. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/dong21a.html. 44

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, 2018. 67

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. Amnesic Probing: Behavioral Explanation with Amnesic Counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175, 03 2021. ISSN 2307-387X. doi: 10.1162/tacl_a_00359. URL https://doi.org/10.1162/tacl_a_00359. 43

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html. 70

Kawin Ethayarajh and Dan Jurafsky. Attention flows are shapley value explanations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 49–54, 2021. 3, 43

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021. 71

James Fiacco, Samridhi Choudhary, and Carolyn Rose. Deep neural model inspection and comparison via functional neuron pathways. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5754–5764, 2019. 13

Joel Escudé Font and Marta R Costa-jussà. Equalizing gender bias in neural machine translation with word embeddings techniques. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 147–154, 2019. 72

Christopher Frye, Damien de Mijolla, Tom Begley, Laurence Cowton, Megan Stanley, and Ilya Feige. Shapley explainability on the data manifold. In *International Conference on Learning Representations*, 2020. 48

Felix A Gers and E Schmidhuber. Lstm recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, 12(6):1333–1340, 2001. 7

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. Under the Hood: Using Diagnostic Classifiers to Investigate and Improve how Language Models Track Agreement Information. aug 2018. URL http://arxiv.org/abs/1808.08079. 7, 11

Yoav Goldberg. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*, 2019. 25, 34, 43, 65

Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2017. 10

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 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, 2018. URL https://github.com/. 11, 18

Han Guo, Nazneen Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. Fastif: Scalable influence functions for efficient model interpretation and debugging. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10333–10350, 2021. 71

Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. Bert & family eat word salad: Experiments with text understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12946–12954, 2021. 67

Peter Hase, Harry Xie, and Mohit Bansal. The out-of-distribution problem in explainability and search methods for feature importance explanations. *Advances in Neural Information Processing Systems*, 34, 2021. 48

John Hewitt and Percy Liang. 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)*, 2019. 25, 43

John Hewitt and Christopher D Manning. 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)*, 2019. 25, 43, 65

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. In *NIPS 2014 Deep Learning Workshop*. 71

Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998. 10

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1, 10, 55

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. Visualisation and 'Diagnostic Classifiers' Reveal How Recurrent and Recursive Neural Networks Process Hierarchical Structure. *Journal of Artificial Intelligence Research*, 61:907–926, apr 2018. ISSN 1076-9757. doi: 10.1613/jair.1.11196. URL https://jair.org/index.php/jair/article/view/11196. 11, 43

Alon Jacovi and Yoav Goldberg. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, 2020. 54, 67

Alon Jacovi, Swabha Swayamdipta, Shauli Ravfogel, Yanai Elazar, Yejin Choi, and Yoav Goldberg. Contrastive explanations for model interpretability. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1597–1611, 2021. 71

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does bert learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. 43, 65

Rohan Jha, Charles Lovering, and Ellie Pavlick. Does data augmentation improve generalization in nlp? *arXiv preprint arXiv:2004.15012*, 2020. 61

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019. 43, 71

Yiming Ju, Yuanzhe Zhang, Zhao Yang, Zhongtao Jiang, Kang Liu, and Jun Zhao. The logic traps in evaluating post-hoc interpretations. *arXiv preprint arXiv:2109.05463*, 2021. 54

Jaap Jumelet, Willem Zuidema, and Dieuwke Hupkes. 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, 2019. 43

Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*, 2015. 11

Hyunjik Kim, George Papamakarios, and Andriy Mnih. The lipschitz constant of self-attention. In *International Conference on Machine Learning*, pages 5562–5571. PMLR, 2021. 70

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 61

Josef Klafka and Allyson Ettinger. Spying on your neighbors: Fine-grained probing of contextual embeddings for information about surrounding words. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020. 43

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017. 71

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. Revealing the dark secrets of bert. 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 4365–4374, 2019. 3, 37, 43

I Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. Problems with shapley-value-based explanations as feature importance measures. In *International Conference on Machine Learning*, pages 5491–5500. PMLR, 2020. 48

Yair Lakretz, German Kruszewski, Theo Desbordes, Dieuwke Hupkes, Stanislas Dehaene, and Marco Baroni. 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)*, 2019. ix, 4, 7, 11, 18, 19, 20, 21, 43

Klas Leino, Shayak Sen, Anupam Datta, Matt Fredrikson, and Linyi Li. Influence-directed explanations for deep convolutional networks. In *2018 IEEE International Test Conference (ITC)*, pages 1–8. IEEE, 2018. 4, 5, 8, 11, 12, 13, 15, 16, 26, 31, 32, 33, 40

Yongjie Lin, Yi Chern Tan, and Robert Frank. Open sesame: Getting inside bert's linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019. 3, 25, 34, 43

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535, 2016. 3, 4, 7, 11, 43, 65

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 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*, 2019a. 43

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019b. 47, 63, 66

Charles Lovering, Rohan Jha, Tal Linzen, and Ellie Pavlick. Predicting inductive biases of pre-trained models. In *International Conference on Learning Representations*, 2020. 55, 68, 71

Kaiji Lu, Piotr Mardziel, Klas Leino, Matt Fedrikson, and Anupam Datta. Influence paths for characterizing subject-verb number agreement in lstm language models. *arXiv preprint arXiv:2005.01190*, pages 4748–4757, 2020a. 6

Kaiji Lu, Piotr Mardziel, Klas Leino, Matt Fredrikson, and Anupam Datta. Influence paths for characterizing subject-verb number agreement in LSTM language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020b. 4, 40, 65

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. Gender bias in neural natural language processing. In *Logic, Language, and Security*, pages 189–202. Springer, 2020c. 61, 72

Kaiji Lu, Zifan Wang, Piotr Mardziel, and Anupam Datta. Influence patterns for explaining information flow in bert. *Advances in Neural Information Processing Systems*, 34, 2021. 6

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017. 3, 46, 48, 53, 54, 67

Nikolay Malkin, Sameera Lanka, Pranav Goel, and Nebojsa Jojic. Studying word order through iterative shuffling. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10351–10366, 2021. 45

Karttikeya Mangalam and Vinay Uday Prabhu. Do deep neural networks learn shallow learnable examples first? 2019. 68

Thomas Manzini, Lim Yao Chong, Alan W Black, and Yulia Tsvetkov. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. 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 615–621, 2019. 72

Rebecca Marvin and Tal Linzen. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018. 1, 34, 65

Chandler May, Alex Wang, Shikha Bordia, Samuel Bowman, and Rachel Rudinger. On measuring social biases in sentence encoders. 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 622–628, 2019. 72

R Thomas McCoy, Junghyun Min, and Tal Linzen. Berts of a feather do not generalize together: Large variability in generalization across models with similar test set performance. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 217–227, 2020. 40

Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, 2019. 1, 3, 5, 40, 45, 47, 59, 67, 81

Paul Michel. *Learning Neural Models for Natural Language Processing in the Face of Distributional Shift*. PhD thesis, Stanford University, 2021. 71

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, 2019. 3, 37, 43

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. 14

Junghyun Min, R Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2339–2352, 2020. 47, 59

W James Murdoch, Peter J Liu, and Bin Yu. Beyond word importance: Contextual decomposition to extract interactions from lstms. In *International Conference on Learning Representations*, 2018. 11

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, 2018. 67

Patrick M Parkinson. Sr 11-7: Guidance on model risk management. *Board of Governors of the Federal Reserve System, 20th Street NW, Washington, DC*, 20551, 2011. 1, 45

Thang Pham, Trung Bui, Long Mai, and Anh Nguyen. Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1145–1160, 2021. 5, 45, 63, 67

Sai Prasanna, Anna Rogers, and Anna Rumshisky. When bert plays the lottery, all tickets are winning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3208–3229, 2020. 43, 71

Manish Raghavan, Solon Barocas, Jon Kleinberg, and Karen Levy. Mitigating bias in algorithmic hiring: Evaluating claims and practices. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 469–481, 2020. 72

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. Visualizing and measuring the geometry of bert. In *Advances in Neural Information Processing Systems 32*. NeurIPS, 2019. 25, 43

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016. 51

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of nlp models with checklist. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, 2020. 1, 3, 40, 67

Alexis Ross, Ana Marasović, and Matthew E Peters. Explaining nlp models via minimal contrastive editing (mice). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3840–3852, 2021. 70, 71

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 43, 63, 66, 71

Sofia Serrano and Noah A Smith. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. 3, 43

Lloyd S Shapley, HW Kuhn, and AW Tucker. Contributions to the theory of games. *Annals of Mathematics studies*, 28(2):307–317, 1953. 2, 3, 46, 48, 53, 67

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. 2013. 11, 57

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, 2021a. 5, 45, 52, 67

Koustuv Sinha, Prasanna Parthasarathi, Joelle Pineau, and Adina Williams. Unnatural language inference. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7329–7346, 2021b. 5, 45, 67

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. 13

Erik Strumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, 11:1–18, 2010. 46, 54, 67

Mukund Sundararajan and Amir Najmi. The many shapley values for model explanation. In *International Conference on Machine Learning*, pages 9269–9278. PMLR, 2020. 12, 46, 67

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328. JMLR. org, 2017. 2, 3, 8, 11, 14, 15, 57

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *International Conference on Machine Learning*, pages 9438–9447. PMLR, 2020. 71

Ian Tenney, Dipanjan Das, and Ellie Pavlick. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019a. 25, 43

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel Bowman, Dipanjan Das, et al. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019*, 2019b. 43

Marcos Treviso and André FT Martins. The explanation game: Towards prediction explainability through sparse communication. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 107–118, 2020. 28

Henry Tsai, Jason Riesa, Melvin Johnson, Naveen Arivazhagan, Xin Li, and Amelia Archer. Small and practical bert models for sequence labeling. 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 3632–3636, 2019. 71

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*, 2019. 34

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017. 1, 26, 55, 57

Jesse Vig and Yonatan Belinkov. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019. 3, 43

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. 3, 37, 43

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing nlp. 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 2153–2162, 2019. 67

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, 2018. 1, 34, 63

Chenguang Wang, Zihao Ye, Aston Zhang, Zheng Zhang, and Alexander J Smola. Transformer on a diet. *arXiv preprint arXiv:2002.06170*, 2020a. 71

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Jiangnan Xia, Zuyi Bao, Liwei Peng, and Luo Si. Structbert: Incorporating language structures into pre-training for deep language understanding. In *International Conference on Learning Representations*, 2019. 47, 63, 71

Zifan Wang, Haofan Wang, Shakul Ramkumar, Matt Fredrikson, Piotr Mardziel, and Anupam Datta. Smoothed geometry for robust attribution. In *Advances in Neural Information Processing Systems*. NeurIPS, 2020b. 13, 71

Jason Wei, Dan Garrette, Tal Linzen, and Ellie Pavlick. Frequency effects on syntactic rule learning in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 932–948, 2021. 65

Gail Weiss, Yoav Goldberg, and Eran Yahav. Thinking like transformers. In *International Conference on Machine Learning*, pages 11080–11090. PMLR, 2021. 70

Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122, 2018. 59

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019. 60, 63, 66

Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S Weld. Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6707–6723, 2021. 3, 67, 71

Zhengxuan Wu, Thanh-Son Nguyen, and Desmond Ong. Structured self-attention weights encodes semantics in sentiment analysis. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 255–264, 2020. 25, 28, 31, 32, 43

Yadollah Yaghoobzadeh, Soroush Mehri, Remi Tachet des Combes, Timothy J Hazen, and Alessandro Sordoni. Increasing robustness to spurious correlations using forgettable examples. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3319–3332, 2021. 59

Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. Representer point selection for explaining deep neural networks. In *Advances in neural information processing systems*, pages 9291–9301, 2018. 71

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020. 71

Die Zhang, Huilin Zhou, Hao Zhang, Xiaoyi Bao, Da Huo, Ruizhao Chen, Xu Cheng, Mengyue Wu, and Quanshi Zhang. Building interpretable interaction trees for deep nlp models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14328–14337, 2021. 3, 67

Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41, 2020. 67

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. Gender bias in contextualized word embeddings. In *Proceedings of NAACL-HLT*, pages 629–634, 2019. 72

Yilun Zhou, Serena Booth, Marco Tulio Ribeiro, and Julie Shah. Do feature attribution methods correctly attribute features? In *eXplainable AI approaches for debugging and diagnosis.*, 2021. 54

Xiaodan Zhu, Hongyu Guo, Saif Mohammad, and Svetlana Kiritchenko. An empirical study on the effect of negation words on sentiment. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 304–313, 2014. 38