LS-Tree: Model Interpretation When the Data Are Linguistic

Jianbo Chen
University of California, Berkeley
jianbochen@berkeley.edu

Michael I. Jordan
University of California, Berkeley
jordan@cs.berkeley.edu

Abstract

We study the problem of interpreting trained classification models in the setting of linguistic data sets. Leveraging a parse tree, we propose to assign least-squares-based importance scores to each word of an instance by exploiting syntactic constituency structure. We establish an axiomatic characterization of these importance scores by relating them to the Banzhaf value in coalitional game theory. Based on these importance scores, we develop a principled method for detecting and quantifying interactions between words in a sentence. We demonstrate that the proposed method can aid in interpretability and diagnostics for several widely-used language models.

Introduction

Modern machine learning models can be difficult to probe and understand after they have been trained. This is a major problem for the field, with consequences for trustworthiness, diagnostics, debugging, robustness, and a range of other engineering and human interaction issues surrounding the deployment of a model.

There have been several lines of attack on this problem. One involves changing the model design or training process so as to enhance interpretability. This can involve retreating to simpler models and/or incorporating strong regularizers that effectively simplify a complex model. In both cases, however, there is a possible loss of prediction accuracy. Models can also be changed in more sophisticated ways to enhance interpretability; for example, attention-based methods have yielded deep models for vision and language tasks that improve interpretability at no loss to prediction accuracy (Ba, Mnih, and Kavukcuoglu 2014; Xu et al. 2015; Gregor et al. 2015; Chen et al. 2015; Yang et al. 2016; Xu and Saenko 2016; Vaswani et al. 2017).

Another approach treats interpretability as a separate problem from prediction. Given a predictive model, an interpretation method yields, for each instance to which the model is applied, a vector of importance scores associated with the underlying features. Within this general framework, methods can be classified as being model-agnostic or model-aware. Model-aware methods require additional assumptions, or are specific to a certain class of models (Simonyan, Vedaldi, and Zisserman 2014; Bach et al. 2015; Shrikumar, Greenside, and Kundaje 2017; Karpathy, Johnson, and Fei-Fei 2016; Sundararajan, Taly, and Yan 2017; Godin et al. 2018). Model-agnostic methods can be applied in a black-box manner to arbitrary models (Ribeiro, Singh, and Guestrin 2016; Baehrens et al. 2010; Lundberg and Lee 2017; Štrumbelj and Kononenko 2010; Datta, Sen, and Zick 2016; Li, Monroe, and Jurafsky 2016).

While the generality of the stand-alone approach to interpretation is appealing, current methods provide little opportunity to leverage prior knowledge about what constitutes a satisfying interpretation in a given domain. Such interpretive capabilities are available most notably in the setting of natural-language processing (NLP), where there is an ongoing effort to incorporate linguistic structure (syntactic, semantic and pragmatic) in machine learning models. Such structure can be brought to bear in the model construction, the interpretation of a model, or both. For example, Socher et al. (2013) introduced a recursive deep model to understand and leverage compositionality in tasks such as sentiment detection. Lei, Barzilay, and Jaakkola (2016) proposed to use a combination of two modular components, generator and encoder, to explicitly generate rationales and make prediction for NLP tasks.

Compositionality, expressed in the rules used to construct a sentence from its constituent expressions, is an important property of natural language. While current interpretation methods fall short of quantifying compositionality directly, there has been a growing interest in investigating the manner in which existing deep models capture the interactions between constituent expressions that are critical for successful prediction (Li et al. 2016; Lei, Barzilay, and Jaakkola 2016; Li, Monroe, and Jurafsky 2016; Godin et al. 2018). However, existing approaches generally fall short of providing a systematic, quantitative treatment of interactions, and the generality to be applied to arbitrary models.

In the current paper, we focus on the model-agnostic interpretation of NLP models. Our approach quantifies the importance of words by leveraging the syntactic structure of linguistic data, as represented by constituency-based parse trees. In particular, we develop the *LS-Tree value*, a procedure that provides instance-wise importance scores for a model by minimizing the sum-of-squared residuals at every node of a parse tree for the sentence in consideration. We provide theoretical support for this by relating it to the Banzhaf value

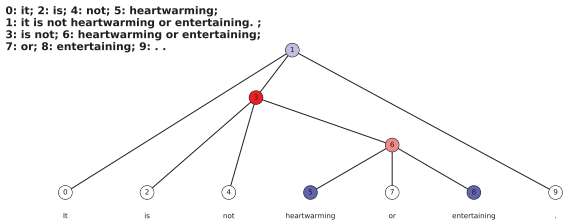


Figure 1: An example parse tree. Top left shows how each node corresponds to a word subset. Color indicates the direction and strength of interaction as assigned by Algorithm 1. Red is used for the direction of positive class, and blue otherwise.

in coalitional game theory (Banzhaf III 1964).

Our framework also provides a seedbed for studying compositionality in natural language. Based on the LS-Tree value, we develop a novel method for quantifying interactions between sibling nodes on a parse tree captured by the target model, by exploiting Cook's distance in linear regression (Cook 1977). We show that the proposed algorithm can be used to analyze several aspects of widely-used NLP models, including nonlinearity, the ability to capture adversative relations, and overfitting. In particular, we carry out a series of experiments studying four models—a linear model with Bag-Of-Word features, a convolutional neural network (Kim 2014), an LSTM (Hochreiter and Schmidhuber 1997), and the recently proposed BERT model (Devlin et al. 2018).

Least squares on parse trees

For simplicity, we restrict ourselves to classification. Assume a model maps a sentence to a vector of class probabilities. We use f to denote the function that maps an input sentence $x = (x_1, \dots, x_d)$ to the log-probability score of a selected class. Let $2^{[d]}$ denote the power set of $[d] := \{1, 2, \dots, d\}$. The parse tree maps the sentence to a collection of subsets, denoted as $\mathcal{F} \subset 2^{[d]}$, where each subset $S \in \mathcal{F}$ contains the indices of words corresponding to one node in the parse tree. See Figure 1 for an example. By abuse of notation, we use $f(S)$ to denote the output of the model evaluated on the words with indices S, with the rest of the words replaced by zero paddings or some reference placeholder. We call $v : \mathcal{F} \to \mathbb{R}$ defined by $v(S) := f(S) - f(\emptyset)$ a *characteristic function*, which captures the importance of each word subset to the prediction.

We seek the optimal linear function on the Boolean hypercube to approximate the characteristic function on $\mathcal{F}$, and use the coefficients as importance scores assigned to each word. Concretely, we solve the following least squares problem:

$$\min_{\psi \in \mathbb{R}^d} \sum_{S \in \mathcal{F}} [v(S) - \sum_{i \in S} \psi_i]^2, \tag{1}$$

where component ψ_i of the optimal ψ is the importance score of the word with index i. We refer to the map from $(\mathcal{F}, v)$ to the solution of Equation (1) as the *LS-Tree value*, because it results from least squares (LS) on parse trees, and can be considered as a *value* in coalitional game theory.

Connection to coalitional game theory

In this section, we give an interpretation of the LS-Tree value from the perspective of coalitional game theory.

Model interpretation has been studied using tools from coalitional game theory (Štrumbelj and Kononenko 2010; Datta, Sen, and Zick 2016; Lundberg and Lee 2017; Chen et al. 2019). We build on this line of research by considering a restriction on coalitions induced by the syntactic structure of the input.

Let $\mathcal{F} \subset 2^{[d]}$ be the collection of word subsets constructed from the parse tree. Taking each word as a player, we can define a coalitional game between d words in a sentence as a pair $(\mathcal{F}, v)$, where $\mathcal{F} \subset 2^{[d]}$ enforces restrictions on coalition among players and $v : \mathcal{F} \to \mathbb{R}$ with $v(\emptyset) = 0$ is the characteristic function defined by the model evaluated on each coalition. A *value* is a mapping that associates a d-dimensional payoff vector $\psi(\mathcal{F}, v)$ to each game $(\mathcal{F}, v)$, each entry corresponding to a word. The value provides rules which give allocations to each player for any game.

The problem of defining a fair value in the setting of full coalition (when $\mathcal{F} = 2^{[d]}$) has been studied extensively in coalitional game theory (Shapley 1953; Banzhaf III 1964). One popular value is the Banzhaf value introduced by Banzhaf III (1964). For each $i \in [d]$ it defines the value:

$$\phi_i(2^{[d]}, v) = \frac{1}{2^{d-1}} \sum_{S \subset N \setminus i} [v(S \cup i) - v(S)].$$

The Banzhaf value can be characterized as the unique value that satisfies the following four properties (Nowak 1997):

i) Symmetry: If $v(S \cup i) = v(S \cup j)$ for all $S \subset [d] \setminus \{i, j\}$, we have $\phi_i(2^{[d]}, v) = \phi_j(2^{[d]}, v)$.

ii) Dummy player property: If $v(S \cup i) = v(S) + v(i)$ for all $S \subset [d] \setminus i$, we have $\phi_i(2^{[d]}, v) = v(i)$.

iii) Marginal contributions: For any two characteristic functions v, w such that $v(S \cup i) - v(S) = w(S \cup i) - w(S)$ for any $S \subset [d]$, we have $\phi_i(2^{[d]}, v) = \phi_i(2^{[d]}, w)$.

iv) 2-Efficiency: If $i, j \in [d]$ merges into a new player p, then $\phi_p(2^{[d] \setminus \{i,j\} \cup p}, v^{ij}) = \phi_i(2^{[d]}, v) + \phi_j(2^{[d]}, v)$, where $v^{ij}(S) := v(S)$ if $p \notin S$ and $v^{ij}(S) := v(S \setminus p \cup i \cup j)$ otherwise, for any $S \subset [d] \setminus \{i, j\} \cup p$.

These properties are natural for allocation of importance to prediction in model interpretation. Symmetry states that two features have the same allocation if their marginal contributions to feature subsets are the same. The dummy property states that a feature is allocated the same amount as the contribution of itself alone if its marginal contribution always equals the model evaluation on its own. The linear model yields such an example. Marginal contributions states that a feature which has the same marginal contribution between two models for any word subset has the same amount of allocation. 2-Efficiency states that allocation of importance is immune to artificial merging of two features.

To employ game-theoretic concepts such as the Banzhaf value in the interpretation of NLP models, we need to recognize that arbitrary combinations of words are not likely to be accepted as valid interpretations by humans. We might wish to start with a set of combinations that are likely to be

interpretable by humans, and can be obtained via human-interpretable data, and then define the worth of other combinations of words via extrapolation. It turns out that the LS-Tree value as defined in the previous section can be interpreted as exactly such an extrapolation, where each node of the parse tree represents an interpretable word combination:

Theorem 1. *Suppose a value ψ coincides with the Banzhaf value ϕ for any game of full coalition, and for every game $(\mathcal{F}, v)$ with restricted coalition, it is consistent under the addition of an arbitrary subset $S \notin \mathcal{F}$:*

$$\psi(\mathcal{F}, v) = \psi(\mathcal{F} \cup \{S\}, v'), \tag{2}$$

where v' is defined as $v'(T) = v(T)$ for $T \neq S$ and $v'(S) = \sum_{i \in S} \psi_i(\mathcal{F}, v)$. Then ψ coincides with the LS-Tree value.

Proof. It was shown in Hammer and Holzman (1992) that the Banzhaf value assigns to each player i the corresponding coefficient in the best linear approximation of v. That is,

$$\phi(2^{[d]}, v) = \arg\min_{\psi \in \mathbb{R}^d} \sum_{S \subset [d]} [v(S) - \sum_{i \in S} \psi_i]^2.$$

Based on the proof of Theorem 3.3 in Katsev (2011),[1] it follows directly that ψ^*, as is defined by Equation (3), is the unique value that coincides with $v \to \psi^*(2^{[d]}, v)$ with full coalition and is consistent under the addition of an arbitrary subset:

$$\psi^*(\mathcal{F}, v) = \arg\min_{\psi \in \mathbb{R}^d} \sum_{S \in \mathcal{F}} w_S [v(S) - \sum_{i \in S} \psi_i]^2. \tag{3}$$

Taking $w_S \equiv 1$, the theorem is established. □

The Shapley value is another well-known concept of value in cooperative game theory. The Banzhaf value differs from the Shapley value by replacing the axiom of Efficiency in the definition of the Shapley value with 2-Efficiency (Shapley 1953; Dubey and Shapley 1979). Both values have been employed to capture notions of model interpretation in previous work (Lundberg and Lee 2017; Štrumbelj and Kononenko 2010; Datta, Sen, and Zick 2016; Dubey and Shapley 1979). We prefer to build our framework on the Banzhaf value instead of the Shapley value, because the structure on the features imposed by the parse tree can be more naturally incorporated into the former, as demonstrated in Theorem 1.

Detecting interactions

We aim to detect and quantify interactions between words in a sentence that have been captured by the target model. While there are exponentially many possible interactions between arbitrary words, we restrict ourselves to the ones permitted by the structure of language. Concretely, we focus on interactions between siblings, or nodes with a common parent, in the parse tree. As an example, node 3 in Figure 1 represents interaction between "is," "not" and "heartwarming or entertaining."

[1]The original theorem is established for the solution to Problem (3) with the efficiency constraint that $\sum_{i \in [d]} x_i = v([d])$. But the same proof follows for the unconstrained version.

We define interaction as *deviation of composition from linearity* in a given sentence. As a result, all non-leaf nodes in the tree are expected to admit zero interaction for a linear model. The above definition suggests that interaction can be quantified by studying how the inclusion of a common parent representing the interaction affects the coefficients of the linear approximation of the model.

Cook's distance is a classic metric in linear regression that captures the influence of a data point (Cook 1977). It is defined as a constant multiple of the squared distance between coefficients after a data point is moved, where the distance metric is defined by the data matrix $X \in \mathbb{R}^{n \times d}$:

$$D_i = \text{Const.} \cdot (\hat{\beta}_{(i)} - \hat{\beta})^T X^T X (\hat{\beta}_{(i)} - \hat{\beta}),$$

where $\hat{\beta}_{(i)}$ and $\hat{\beta}$ are the least-squares estimate with the ith data point deleted and the original least-squares estimate respectively. A larger Cook's distance indicates a larger influence of the corresponding data point.

In our setting, the data matrix X is a Boolean matrix where each row corresponds to a node in the tree, and an entry is one if and only if the word of the corresponding index lies in the subtree of the node. To capture the interaction of a non-leaf node i (corresponding to some $S \in \mathcal{F}$), it does not suffice to only delete the corresponding row, because all of its ancestor nodes contain the segment represented by the node as well. To deal with this issue, we compute the distance between the least-squares estimate with the rows corresponding to the node and all of its ancestors deleted, and the least-squares estimate with only the rows corresponding to the ancestors deleted:

$$D_i = d(\hat{\beta}_{(\geq i)}, \hat{\beta}_{(>i)}), \tag{4}$$

where $\hat{\beta}_{(\geq i)}, \hat{\beta}_{(>i)}$ denote the estimates with all ancestors, including and excluding node i, deleted. Cook's distance $d(a, b) = a^T X^T X b$ no longer has its statistical meaning here, as the normality assumption of the linear model no longer holds. A natural choice is the Euclidean distance $d_1(a, b) := \sqrt{a^T b}$, which was also introduced by Cook (1977). One drawback of the Euclidean distance is that it is unable to capture the direction of interaction. When this is an issue, we may use a signed distance: $d_2(a, b) := \sum_i (b_i - a_i)$, which sums up the influence of introducing the extra row on every coefficient of the linear model. We call the score defined by d_1 and d_2 absolute and signed *LS-Tree interaction scores* respectively, as they are constructed from the LS-Tree value.

We propose an iterative algorithm to efficiently compute the interaction of each node on a tree with $n := |\mathcal{F}|$ nodes. As a first step, n model evaluations are performed, one evaluation for each node. For a node i, we denote as $\text{Ch}(i)$ the set of its children, $X_{(\geq i)}$ and $X_{(>i)}$ the data matrices excluding the ancestors of i, further excluding and including i itself respectively, and x_j^T the row corresponding to node j. The interaction score of each $j \in \text{Ch}(i)$ is a function of $\hat{\beta}_{(>j)} - \hat{\beta}_{(\geq j)}$. Denote $A_j = X_{(\geq j)}^T X_{(\geq j)}$. For each non-leaf node j, A_j is of full rank and thus invertible. We show how A_j^{-1} and $\hat{\beta}_{\geq j}$ can be computed from A_i^{-1} and $\hat{\beta}_{\geq i}$. In fact, with an application of the Sherman-Morrison formula (Sherman

Algorithm 1 LS-Tree Interaction Detection

Require: Model f.
Require: Sentence x.
Ensure: LS-Tree value; interaction score.
  Find the parse tree $\mathcal{T}$ of x.
  Find the collection of subsets $\mathcal{F}$ corr. to the parse tree.
  for each node i in $\mathcal{T}$ **do**
    Query the model at the corr. subset S to get $v(S)$.
  end for
  Compute LS-Tree value $\hat{\beta}$ for words via least squares.
  Find the root r of $\mathcal{T}$.
  Recursion(v, $\mathcal{F}$, r, $(X^TX)^{-1}$, $\hat{\beta}$)

Algorithm 2 Recursion

Require: $v, \mathcal{F}$, node j, A_i^{-1}, $\hat{\beta}_{(\geq i)}$
  if j is not a leaf **then**
    Compute $A_j^{-1}, \hat{\beta}_{(\geq j)}, D_j$ via Equation (7) and (6).
    for each child c in of j **do**
      Recursion(v, $\mathcal{F}$, c, A_j^{-1}, $\hat{\beta}_{(\geq j)}$)
    end for
  else
    Assign D_j with $v(j)$ or $|v(j)|$.
  end if

and Morrison 1950), we have

$$\begin{aligned}\hat{\beta}_{(>j)} &= (X_{(\geq j)}^T X_{(\geq j)} + x_j^T x_j)^{-1}(X_{(\geq j)}^T Y_{(\geq j)} + x_j^T Y_j) \\ &= (I - \frac{A_j^{-1}x_j x_j^T}{1 + x_j^T A_j^{-1} x_j})\hat{\beta}_{(\geq j)} + \frac{A_j^{-1}x_j Y_j}{1 + x_j^T A_j^{-1} x_j}.\end{aligned} \tag{5}$$

Rearranging the terms in Equation (5), we have

$$\hat{\beta}_{(\geq j)} = \hat{\beta}_{(>j)} - A_j^{-1}x_j[Y_j - x_j^T \hat{\beta}_{(>j)}]. \tag{6}$$

With another application of the Sherman-Morrison formula, we have

$$\begin{aligned}A_j^{-1} &= (X_{(\geq i)}^T X_{(\geq i)} - x_j x_j^T)^{-1} \\ &= A_i^{-1} + \frac{A_i^{-1}x_j x_j^T A_i^{-1}}{1 - x_j^T A_i^{-1} x_j}.\end{aligned} \tag{7}$$

For leaf nodes, the entry of $\hat{\beta}_{(\geq j)}$ corresponding to j is set to zero, with the remaining entries equal to those of $\hat{\beta}_{(>j)}$. This is a result of the minimal Euclidean norm solution of Problem 1, obtained from the pseudoinverse of A_j. Consequently, the (signed) interaction score of a leaf equals the model evaluation on the leaf alone.

We summarize the derivation in Algorithm 1, which traverses the parse tree from root to leaves in a top-down fashion to compute the interaction scores of each node. As the number of nodes in a parse tree is linear in the number of words, Algorithm 1 is of complexity $\mathcal{O}(d^3)$, plus the complexity of parsing the sentence, which is $\mathcal{O}(d)$ in our experiments, and $\mathcal{O}(d)$ model evaluations. Figure 1 shows how Algorithm 1 assigns signed interaction scores to a given example.

Data Set	Classes	Train Size	Test Size	Avg. Len.	BoW	CNN	LSTM	BERT
SST	2	6,920	872	19.7	82%	85%	85%	93%
IMDB	2	25,000	25,000	325.6	94%	90%	88%	93%
Yelp	2	560,000	38,000	136.2	94%	95%	96%	97%

Table 1: Statistics of the three data sets, together with the test accuracy of the four models.

Experiments

We carry out experiments to analyze the performance of four different models: Bag of Words (BoW), Word-based Convolutional Neural Network (CNN) (Kim 2014), bidirectional Long Short-Term Memory network (LSTM) (Hochreiter and Schmidhuber 1997), and Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al. 2018), across three sentiment data sets of different sizes: Stanford Sentiment Treebank (SST) (Socher et al. 2013), IMDB Movie reviews (Maas et al. 2011) and Yelp reviews Polarity (Zhang, Zhao, and LeCun 2015). For an instance with multiple sentences, we parse each sentence separately, and introduce an extra node as the common parent of all roots. Interactions between sentences are not considered in our experiments. The code for replicating the experiments is available online at https://github.com/Jianbo-Lab/LS-Tree.

BoW fits a linear model on the Bag-of-Words features. Both CNN and LSTM use a 300-dimensional GloVe word embedding (Pennington, Socher, and Manning 2014). The CNN is composed of three 100-dimensional convolutional 1D layers with 3, 4 and 5 kernels respectively, concatenated and fed into a max-pooling layer followed by a hidden dense layer. The LSTM uses a bidirectional LSTM layer with 128 units for each direction. BERT pre-trains a deep bidirectional Transformer (Vaswani et al. 2017) on a large corpus of text by jointly conditioning on both left and right context in all layers. It has achieved state-of-the-art performance on a large suite of sentence-level and token-level tasks. See Table 1 for a summary of data sets and the accuracies of the four models.

We use the Stanford constituency parser (Goldberg and Nivre 2012; Sagae and Lavie 2005; Zhang and Clark 2009; Zhu et al. 2013) for all the experiments. It is a transition-based parser that is faster than chart-based parsers yet achieves comparable accuracy, by employing a set of shift-reduce operations and making use of non-local features.

Deviation from linearity

We quantify the deviation of three nonlinear models from a linear model via the proposed LS-Tree value and interaction scores, both for specific instances and on a data set.

The LS-Tree value can be interpreted as supplying the coefficients of the best linear model used to approximate the target model locally for each instance. The correlation between the LS-Tree value and the global linear model with Bag of Words (BoW) features can be used as a measure of nonlinearity of the target model at the instance. Table 2a shows three examples in SST, correctly classified by both BERT and BoW. BERT has low and high correlations with linear models at the first and second examples in Table 2a respectively. In particular, the top keywords, as ranked by the LS-Tree value, are different between two models.

BERT	BoW	Category	Correlation	Depth
Even if you do n't think kissinger's any more guilty of criminal activity than most contemporary statesmen, he'd sure make a courtroom trial great fun to watch.	Even if you don't think kissinger's any more guilty of criminal activity than most contemporary statesmen, he'd sure make a courtroom trial great fun to watch.	Positive	0.173	11
The problem with this film is that it lacks focus.	The problem with this film is that it lacks focus.	Negative	0.939	1
Funny but perilously slight.	Funny but perilously slight.	Positive	0.938	4

(a) Correlation with linear coefficients and depth of the top node are listed. The top two words ranked by the LS-Tree value, and by the linear coefficients, are colorized.

	BoW	CNN	LSTM	BERT
SST	1.000	0.591	0.580	**0.465**
IMDB	1.000	0.442	0.552	**0.321**
Yelp	1.000	0.683	0.684	**0.476**

(b) The average correlation is comparable across different models on the same data set.

Table 2: (a) Examples from SST with BERT and BoW. (b) Average correlation of the LS-Tree values with linear coefficients.

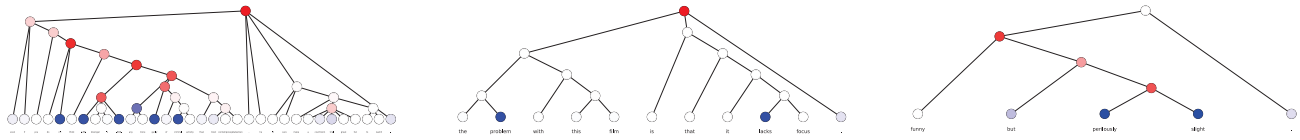
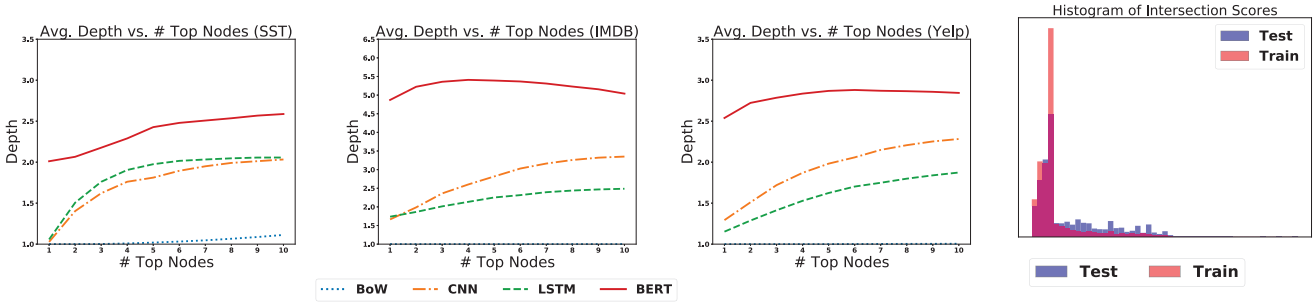


Figure 2: Visualization of parse trees of examples in Table 2a. Nodes are colorized based on the signed interaction scores, red for the direction of positive class, and blue otherwise.



(a) (b)

Figure 3: (a) Average depth of top nodes as the number of the selected top nodes varies. (b) The histogram of intersection scores with train and test data for BERT on SST.

The average of correlation with BoW across instances can be used as a measure of nonlinearity on a certain data set. The average correlation of BoW, CNN, LSTM and BERT with a linear model is shown in Table 2b, which indicates that BERT is the most nonlinear model among the four. CNN is more nonlinear than LSTM on IMDB but comparably nonlinear on SST and Yelp.

Correlation alone may not suffice to capture the nonlinearity of a model. For example, the third sentence in Table 2a has a relatively high correlation, but the bottom left parse tree in Figure 2 indicates that the top interaction ranked by the signed interaction score is the node combining "funny" with "but perilously slight." This indicates the BERT model has captured the adversative conjunction, which BoW is not capable of. The ability to capture closer-to-the-top nodes in a parse tree is an indication of nonlinearity of the model. To quantify this ability, we define the depth of a node in the parse tree as the maximum distance of the node from the bottom:

$$\text{Depth}(i) = \begin{cases} 1 + \max_{c \in \text{Ch}(i)} \text{Depth}(c) & \text{if Ch}(i) \neq \emptyset, \\ 1 & \text{otherwise.} \end{cases}$$

For a linear model, all non-leaf nodes have zero interaction, and thus the top-ranked nodes are of depth 1, until all leaves with positive weights are enumerated. The higher the depth of top-ranked nodes, the more nonlinear a model is at a specific instance.

The average depths of top nodes ranked by interaction scores across instances can be used as a measure of the nonlinearity of the model on that data set. Figure 3a compares the average depths across BoW, CNN, LSTM and BERT on the three data sets, with top $k = 1, 2, \ldots, 10$ words selected. BoW is used as a baseline whose non-leaf nodes have zero interaction scores. We use the absolute interaction scores here to capture all interactions, no matter whether they are in the same or opposite direction of prediction. BERT is still the most capable of capturing deeper interactions, followed by CNN and LSTM. CNN turns out to be a more nonlinear model than LSTM on Yelp, which was not captured by correlation.

Adversative relations

Adversative words are those which express opposition or contrast. They often play an important role in determining the sentiment of an instance, by reversing the sentiment of a preceding or succeeding word, phrase or statement. We focus on four types of adversative words: negation that reverses the sentiment of a phrase or word (e.g., "not"), adversative coordinating conjunctions that express opposition or contrast between two statements (e.g., "but" and "yet"), subordinating conjunctions indicating adversative relationship (e.g., "though," "although," "even though," and "whereas"), prepo-

Dataset	Model	Avg. Score	not	but	yet	though	although	even though	whereas	except	despite	in spite of
SST	BoW	0.153	0.000(6.318)	0.000(0.079)	0.000(2.005)	0.000(0.865)	0.000(2.222)	0.000(0.000)	-(-)	0.000(4.280)	0.000(3.519)	0.000(0.000)
	CNN	0.634	1.673(4.592)	1.694(1.444)	0.568(0.959)	0.213(0.735)	0.915(0.462)	0.626(0.407)	-(-)	0.948(1.175)	**1.452**(4.270)	**2.119**(1.943)
	LSTM	0.79	**1.746**(2.580)	1.502(0.453)	1.449(2.368)	1.153(1.094)	0.338(0.197)	1.794(0.998)	-(-)	**2.353**(3.835)	1.256(1.818)	0.590(0.624)
	BERT	1.238	1.714(4.383)	**2.148**(1.760)	**1.669**(3.120)	**1.525**(3.268)	**1.741**(3.256)	**1.885**(2.092)	-(-)	1.156(3.331)	1.160(2.998)	0.864(2.352)
IMDB	BoW	0.038	0.000(2.683)	0.000(0.263)	0.000(2.210)	0.000(1.473)	0.000(1.710)	0.000(0.000)	0.000(3.604)	0.000(1.342)	0.000(0.132)	-(-)
	CNN	0.424	1.050(0.819)	**3.442**(0.021)	**1.689**(0.295)	0.922(0.085)	1.036(0.071)	1.175(0.467)	0.469(1.064)	**1.590**(4.067)	0.363(0.434)	-(-)
	LSTM	0.126	0.960(3.087)	2.222(0.524)	1.500(0.238)	0.611(0.087)	0.492(1.270)	0.944(0.683)	**1.222**(3.865)	1.294(4.008)	0.286(0.508)	-(-)
	BERT	1.159	**1.616**(2.057)	3.390(1.800)	1.644(1.152)	**1.371**(2.061)	**1.735**(2.123)	**1.457**(1.557)	0.285(0.430)	1.421(2.060)	**1.518**(2.241)	-(-)
Yelp	BoW	0.035	0.000(8.488)	0.000(1.015)	0.000(3.553)	0.000(1.664)	0.000(1.128)	0.000(0.000)	0.000(0.536)	0.000(0.367)	0.000(1.213)	-(-)
	CNN	0.161	**2.287**(3.467)	**2.454**(0.932)	0.516(0.043)	0.988(0.435)	**0.889**(0.075)	0.789(0.621)	0.286(0.671)	**0.522**(2.529)	0.423(0.889)	-(-)
	LSTM	0.224	2.173(5.950)	1.712(1.676)	**0.988**(2.065)	0.984(1.310)	0.706(1.194)	0.559(0.483)	**1.395**(1.793)	0.344(1.408)	0.514(1.153)	-(-)
	BERT	0.746	1.384(2.106)	2.448(0.658)	0.781(0.184)	**1.336**(0.953)	0.596(0.615)	**1.019**(0.880)	0.095(0.162)	0.331(0.074)	**1.041**(0.414)	-(-)

Table 3: Scores with and without parentheses are for nodes containing adversative words alone and their parents where the adversative relation takes place respectively.

Sentence	Meaning	BoW	CNN	LSTM	BERT
... He said he couldn't help. We had to walk **while** the snow blew in our faces. When we were almost there, we saw the shuttle pull out with the smoking shuttle driver in it, driving in the opposite direction, away from us. I can not believe how rude they were.	during the time that	0.000(0.338)	0.781(0.300)	1.761(0.839)	0.062(0.092)
... I ordered a cappuccino. It tasted like milk and no coffee. I was exceptionally disappointed. So **while** the place has a great reputation, even they can screw it up if they don t pay attention to detail, and at this level they should never screw it up. I had a better cup at Martys Market for crying out loud!	whereas (indicating a contrast)	0.000(0.338)	1.142(0.300)	2.155(0.839)	2.167(0.092)
Usually asking the server what is her favorite dish gets you a pretty good recommendation, but in this case, it was crap! The smoked brisket had that discoloration that happens to meat when it's been sitting out for a **while**. And it wasn't even tender!! Am I asking for too much?	a period of time	0.000(0.338)	0.206(0.300)	0.465(0.839)	0.082 (0.092)

Table 4: The word "while" in different contexts. Scores with and without parentheses are for nodes containing "while" alone and their parents respectively.

sitions that precede and govern nouns adversatively (e.g., "except," "despite" and "in spite of").

In most cases, adversative words only function if they interact with their preceding or succeeding companion. In order to verify whether models are able to capture the adversative relationship, we examine the LS-Tree interaction scores of the parent nodes of these words.

We extract all instances that contain any of the above adversative words. Then for each word in an instance, we compute the interaction score of the corresponding node with the word alone, and that of its parent node. A high interaction score on the node with the adversative word alone indicates the model inappropriately attributes to the word itself a negative or positive sentiment. A high interaction score on the parent node indicates the model captures the interaction of the adversative word with its preceding or succeeding component. To compare across different models, we further compute the average interaction score of a generic node across all instances, and report the ratio of average interaction scores of specific nodes to the average score of a generic node for respective models.

Table 3 reports the results on three data sets. We observe the ability of capturing adversative relation for different models varies across data sets. BERT takes the lead in capturing adversative relations on SST and IMDB, perhaps with the help of BERT's pre-training process on a large corpus, but CNN and LSTM catch up with BERT on Yelp, which has a larger amount of training data. On the other hand, all models assign a high score on nodes with adversative words alone. This may result from the uneven distribution of adversative words like "not" among the positive and negative classes. An additional observation is that BERT has the highest score for a generic node on average across three data sets, indicating that BERT is the most sensitive to words and interactions on average.

Some words have different meanings in different contexts. It is interesting to investigate whether a model can distinguish the same word under different contexts. The word "while" is such an example. Table 4 shows three Yelp reviews that include "while." It can be observed that the scores of the parent nodes of "while" is higher than average when "while" contains an adversative meaning, but lower otherwise. This observation holds across CNN, LSTM and BERT, with the sharpest distinction on BERT.

Detecting overfitting

Overfitting happens when a model captures sampling noise in training data, while failing to capture underlying relationships between the inputs and outputs. Overfitting can be a problem in modern machine learning models like deep neural networks, due to their expressive nature. To mitigate overfitting, one often splits the initial training set into a training and a validation set, and uses the latter to obtain an estimate of the generalization performance (Larson 1931). This leads to a waste of training data, depriving the model of potential opportunities to learn from the labelled validation data. We observe that the LS-Tree interaction scores can be used to construct a diagnostic for overfitting, one which is solely computed with unlabelled data.

Figure 3b shows the histograms of absolute interaction scores on small subsets of training and test data of SST, for an overfitted BERT model. The scores are more spread out on test data than those on training data. In fact, we have observed that this phenomenon holds true on average across instances for a overfitted model. In particular, interaction scores of test instances have a larger variance on average than those of training instances when the model is overfitted, but comparable otherwise. The observation can also be generalized to other types of neural networks, including CNN and LSTM. We show in Figure 4 the average variance on training and

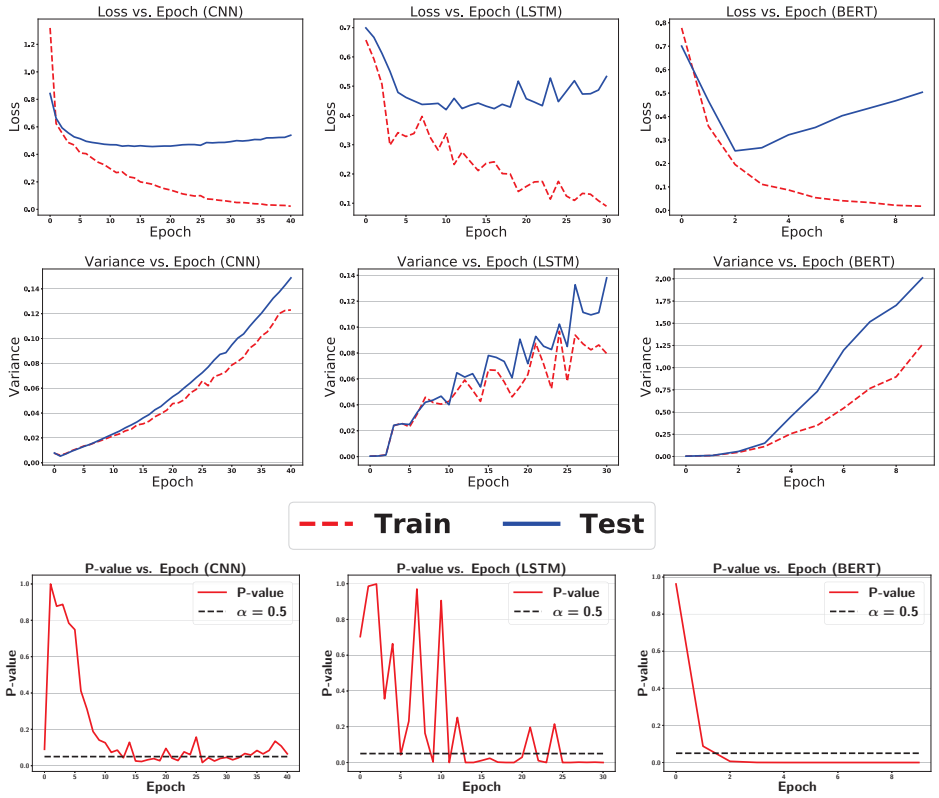


Figure 4: The three figures in Line 1 plot training and test loss of CNN, LSTM, BERT respectively. The figures in Line 2 plot the corresponding average variance of interaction scores across instances over training and test sets. The figures in Line 3 show p-values of permutation tests of $50,000$ iterations with 300 randomly selected instances in training and test sets respectively.

test sets for CNN, LSTM and BERT models against training epochs, together with the loss curves. We observe that overfitting occurs when the variances between training and test sets differ.

The observation suggests we may use the difference of average variances of interaction scores between training and test sets as a diagnostic for overfitting. In particular, a permutation test can be carried out under the null hypothesis of equal average variance. The resulting p-values are plotted against the number of training epochs in the third line of Figure 4. It can be observed that p-values fall below the significance level of 0.05 when overfitting occurs, which suggests the rejection of the null hypothesis as an early stopping criterion in training.

Discussion

We have proposed the LS-Tree value as a fundamental quantity for interpreting NLP models. This value leverages a constituency-based parser so that syntactic structure can play a role in determining interpretations. We have also presented an algorithm based on the LS-Tree value for detecting interactions between siblings of a parse tree. To the best of our knowledge, this is the first model-interpretation algorithm to quantify the interaction between words for arbitrary NLP models. We have applied the proposed algorithm to the problem of assessing the nonlinearity of common neural network models and the effect of adversative relations on the models. We have presented a permutation test based on the LS-Tree interaction scores as a diagnostic for overfitting.

One limitation of LS-Tree is that it is only applicable to interpreting interactions permitted by the syntax of natural language. Further interactions, including semantic interactions, can potentially be incorporated via decoration of the syntactic trees.

References

Ba, J.; Mnih, V.; and Kavukcuoglu, K. 2014. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*.

Bach, S.; Binder, A.; Montavon, G.; Klauschen, F.; Müller, K.-R.; and Samek, W. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS One* 10(7):e0130140.

Baehrens, D.; Schroeter, T.; Harmeling, S.; Kawanabe, M.; Hansen, K.; and Müller, K.-R. 2010. How to explain individual classification decisions. *Journal of Machine Learning Research* 11:1803–1831.

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

Chen, K.; Wang, J.; Chen, L.-C.; Gao, H.; Xu, W.; and Nevatia, R. 2015. ABC-CNN: An attention based convolutional neural network for visual question answering. *arXiv preprint arXiv:1511.05960*.

Chen, J.; Song, L.; Wainwright, M. J.; and Jordan, M. I. 2019. L-shapley and C-shapley: Efficient model interpretation for structured data. In *International Conference on Learning Representations*.

Cook, R. D. 1977. Detection of influential observation in linear regression. *Technometrics* 19(1):15–18.

Datta, A.; Sen, S.; and Zick, Y. 2016. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *Security and Privacy (SP), 2016 IEEE Symposium on*, 598–617. IEEE.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dubey, P., and Shapley, L. S. 1979. Mathematical properties of the banzhaf power index. *Mathematics of Operations Research* 4(2):99–131.

Godin, F.; Demuynck, K.; Dambre, J.; De Neve, W.; and Demeester, T. 2018. Explaining character-aware neural networks for word-level prediction: Do they discover linguistic rules? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 3275–3284. ACL.

Goldberg, Y., and Nivre, J. 2012. A dynamic oracle for arc-eager dependency parsing. *Proceedings of COLING 2012* 959–976.

Gregor, K.; Danihelka, I.; Graves, A.; Rezende, D. J.; and Wierstra, D. 2015. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*.

Hammer, P. L., and Holzman, R. 1992. Approximations of pseudo-boolean functions; applications to game theory. *Zeitschrift für Operations Research* 36(1):3–21.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Karpathy, A.; Johnson, J.; and Fei-Fei, L. 2016. Visualizing and understanding recurrent networks. In *International Conference on Learning Representations*.

Katsev, I. 2011. The least square values for games with restricted cooperation. In *Game Theory and Management*, 117.

Kim, Y. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1746–1751. ACL.

Larson, S. C. 1931. The shrinkage of the coefficient of multiple correlation. *Journal of Educational Psychology* 22(1):45.

Lei, T.; Barzilay, R.; and Jaakkola, T. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 107–117. ACL.

Li, J.; Chen, X.; Hovy, E.; and Jurafsky, D. 2016. Visualizing and understanding neural models in NLP. In *Proceedings of NAACL-HLT*, 681–691.

Li, J.; Monroe, W.; and Jurafsky, D. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Lundberg, S. M., and Lee, S.-I. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc. 4765–4774.

Maas, A. L.; Daly, R. E.; Pham, P. T.; Huang, D.; Ng, A. Y.; and Potts, C. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, 142–150. ACL.

Nowak, A. S. 1997. On an axiomatization of the Banzhaf value without the additivity axiom. *International Journal of Game Theory* 26(1):137–141.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. ACL.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. 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*, 1135–1144. ACM.

Sagae, K., and Lavie, A. 2005. A classifier-based parser with linear run-time complexity. In *Proceedings of the Ninth International Workshop on Parsing Technology*, 125–132. ACL.

Shapley, L. S. 1953. A value for n-person games. *Contributions to the Theory of Games* 2(28):307–317.

Sherman, J., and Morrison, W. J. 1950. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics* 21(1):124–127.

Shrikumar, A.; Greenside, P.; and Kundaje, A. 2017. Learning important features through propagating activation differences. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, 3145–3153. PMLR.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1631–1642. ACL.

Štrumbelj, E., and Kononenko, I. 2010. An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research* 11:1–18.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, 3319–3328.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, 5998–6008.

Xu, H., and Saenko, K. 2016. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *European Conference on Computer Vision*, 451–466. Springer.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, 2048–2057.

Yang, Z.; He, X.; Gao, J.; Deng, L.; and Smola, A. 2016. Stacked attention networks for image question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 21–29.

Zhang, Y., and Clark, S. 2009. Transition-based parsing of the Chinese treebank using a global discriminative model. In *Proceedings of the 11th International Conference on Parsing Technologies*, 162–171. ACL.

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, 649–657.

Zhu, M.; Zhang, Y.; Chen, W.; Zhang, M.; and Zhu, J. 2013. Fast and accurate shift-reduce constituent parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 434–443.