ABSTRACT

This paper aims to explain deep neural networks (DNNs) from the perspective of multivariate interactions. In this paper, we define and quantify the significance of interactions among multiple input variables of the DNN. Input variables with strong interactions usually form a coalition and reflect prototype features, which are memorized and used by the DNN for inference. We define the significance of interactions based on the Shapley value, which is designed to assign the attribution value of each input variable to the inference. We have conducted experiments with various DNNs. Experimental results have demonstrated the effectiveness of the proposed method.

1 INTRODUCTION

Deep neural networks (DNNs) have exhibited significant success in many tasks, and the interpretability of DNNs has received increasing attention in recent years. Most previous studies of post-hoc explanation of DNNs either explain DNN semantically/visually [Lundberg & Lee (2017); Ribeiro et al. (2016)], or analyze the representation capacity of DNNs [Higgins et al. (2017); Achille & Soatto (2018); Fort et al. (2019); Liang et al. (2019)].

In this paper, we propose a new perspective to explain a trained DNN, i.e., quantifying interactions among input variables that are used by the DNN during the inference process. Each input variable of a DNN usually does not work individually. Instead, input variables may cooperate with other variables to make inferences. We can consider the strongly interacted input variables to form a prototype feature (or a coalition), which is memorized by the DNN. For example, the face is a prototype feature for person detection, which is comprised of the eyes, nose, and mouth. Each compositional part inside a face does not contribute to the inference individually, but they collaborate with each other to make the combination of all parts a meaningful feature for person detection.

Previous studies mainly focused on the interaction between two variables [Lundberg et al. (2018); Singh et al. (2018); Murdoch et al. (2018); Janizek et al. (2020)]. Grabisch & Roubens (1999) measured $2^n$ different interaction values for all the $2^n$ combinations of $n$ variables, and the cost of computing each interaction value is NP-hard. In comparison to the unaffordable computational cost, our research summarizes all $2^n$ interactions into a single metric, which provides a global view to explain potential prototype features encoded by DNNs.

* Contribute equally to this paper
† Correspondence.
Figure 1: Overview of the proposed method. In this paper, we aim to quantify the significance of interactions among a set of input variables. We divide the set of input variables into various coalitions, which mainly encode either positive or negative interaction effects. In this figure, each red region represents a coalition that mainly encodes positive interaction effects, and each blue region represents a coalition that mainly encodes negative interaction effects. Thickness of the edge indicates the strength of the interaction.

In this study, we define the significance of interactions among multiple input variables based on game theory. In the game theory, each input variable can be viewed as a player. All players (variables) are supposed to obtain a high reward in a game. Let us consider interactions w.r.t. the scalar output $y = f(I)$ of the DNN (or interactions w.r.t. one dimension of the vectorized network output), given the input $I$ with overall $n$ players. The absolute change of $y$ w.r.t. an empty input $0$, i.e. $|f(I) - f(0)|$, can be used as the total reward of all players (input variables). The reward can be allocated to each player, which indicates the contribution made by the player (input variable), denoted by $\phi_1, \phi_2, \ldots, \phi_m$, satisfies $|f(I) - f(0)| = \sum_{i=1}^{n} \phi_i$.

A simple definition of the interaction is given as follows. If a set of $m$ players $S$ always participates in the game together, these players can be regarded to form a coalition. The coalition can obtain a reward, denoted by $\phi_S$. $\phi_S$ is usually different from the sum of rewards when players in the set $S$ participate in the game individually. The additional reward $\phi_S = \sum_{i \in S} \phi_i$ obtained by the coalition can be quantified as the interaction. If $\phi_S = \sum_{i \in S} \phi_i > 0$, we consider players in the coalition have a positive effect. Whereas, $\phi_S = \sum_{i \in S} \phi_i < 0$ indicates an negative/adversarial effect among the set of variables in $S$.

However, $\phi_S = \sum_{i \in S} \phi_i$ mainly measures the interaction in a single coalition, whose interaction is either purely positive or purely negative. In real applications, for the set $S$ with $m$ players, players may form at most $2^m - m - 1$ different coalitions. Some coalitions have positive interaction effects, while others have negative interaction effects. Thus, interactions of different coalitions can counteract each other.

In order to objectively measure the significance of the overall interactions within a specific set $S$ with $m$ players (variables), we propose a new metric, which reflects both positive and negative interaction effects. We develop a method to divide the $m$ players into a few coalitions, and ensures that each coalition mainly contains positive interaction effects. Similarly, we can also divide the $m$ players into another few coalitions, each of which mainly encodes negative interaction effects. In this way, we can quantify the significance of both positive and negative interaction effects, as shown in Figure [1]

In experiments, we have applied our method to five DNNs with different architectures for various tasks. Experimental results have provided new insights to understand these DNNs. Besides, our method can be used to mine prototype features towards correct and incorrect predictions of DNNs.

Contributions of this study can be summarized as follows. (1) In this paper, we define and quantify the significance of interactions among multiple input variables in the DNN, which can reflect both positive and negative interaction effects. (2) Our method can extract prototype features, which provides new perspectives to understand the DNN, and mines prototype features towards correct and incorrect predictions. (3) We develop a method to efficiently approximate the significance of interactions.

2 RELATED WORK

The interpretability of DNNs is an emerging direction in machine learning, and many methods have been proposed to explain the DNN in natural language processing (NLP). Some methods computed importance/attribution/saliency values for input variables w.r.t. the output of the DNN [Lundberg & Lee (2017); Ribeiro et al. (2016); Binder et al. (2016); Simonyan et al. (2013); Shrikumar et al. (2016); Springenberg et al. (2014); Li et al. (2015) 2016]. Another kind of methods to understand
DNNs are to measure the representation capacity of DNNs [Higgins et al. (2017); Achille & Soatto (2018); Fort et al. (2019); Liang et al. (2019); Guan et al. (2019)] proposed a metric to measure how much information of an input variable was encoded in an intermediate-layer of the DNN. Some studies designed a network to learn interpretable feature representations. Chung et al. (2017) modified the architecture of recurrent neural network (RNN) to capture the latent hierarchical structure in the sentence. Shen et al. (2019) revised the LSTM to learn interpretable representations. Other studies explained the feature processing encoded in a DNN by embedding hierarchical structures into the DNN [Tai et al. (2015); Wang et al. (2019b); Wang et al. (2019a)] used structural position representations to model the latent structure of the input sentence by embedding a dependency tree into the DNN. Dyer et al. (2016) proposed a generative model to model hierarchical relationships among words and phrases.

In comparison, this paper focuses on the significance of interactions among multiple input variables, which provides a new perspective to understand the behavior of DNNs.

### 2.1 Interactions

Several studies explored the interaction between input variables. Bien et al. (2013) developed an algorithm to learn hierarchical pairwise interactions inside an additive model. Sorokina et al. (2008) proposed a method to detect the statistical interaction via an additive model-based ensemble of regression trees. Some studies defined different types of interactions between variables to explain DNNs from different perspectives. Murdoch et al. (2018) proposed to use the contextual decomposition to extract the interaction between gates in the LSTM. Singh et al. (2018); Jin et al. (2019) extended the contextual decomposition to produce a hierarchical clustering of input features and contextual independence of input words, respectively. Janizek et al. (2020) quantified the pairwise feature interaction in DNNs by extending the explanation method of Integrated Gradients Sundararajan et al. (2017). Tsang et al. (2018) measured the pairwise interaction by applying the learned weights of the DNN.

Unlike above studies, Lundberg et al. (2018) defined the interaction between two variables based on the Shapley value for tree ensembles. Because Shapley value was considered as the unique standard method to estimate contributions of input words to the prediction score with solid theoretical foundations Weber (1988), this definition of the interaction can be regarded to objectively reflect the collaborative/adversarial effects between variables w.r.t. the prediction score. Furthermore, Grabisch & Roubens (1999); Dhamdhere et al. (2019) extended this definition to the elementary interaction component and the Shapley-Taylor index among various variables, respectively. These studies quantified the significance of interaction for each possible specific combination of variables, instead of providing a single metric to quantify interactions among all combinations of variables. In comparison, this paper aims to use a single metric to quantify the significance of interactions among all combinations of input variables, and reveal prototype features modeled by the DNN.

### 2.2 Shapley Values

The Shapley value was initially proposed in the game theory Shapley (1953). Let us consider a game with multiple players. Each player can participate in the game and receive a reward individually. Besides, some players can form a coalition and play together to pursue a higher reward. Different players in a coalition usually contribute differently to the game, thereby being assigned with different proportions of the coalition’s reward. The Shapley value is considered as a unique method that fairly allocates the reward to players with certain desirable properties Weber (1988); Ancona et al. (2019). Let \( N = \{1, 2, \ldots, n\} \) denote the set of all players, and \( 2^N \) represents all potential subsets of \( N \). A game \( v : 2^N \to \mathbb{R} \) is implemented as a function that maps from a subset to a real number. When a subset of players \( S \subseteq N \) plays the game, the subset can obtain a reward \( v(S) \). Specifically, \( v(\emptyset) = 0 \). The Shapley value of the \( i \)-th player \( \phi_v(i|N) \) can be considered as an unbiased contribution of the \( i \)-th player.

\[
\phi_v(i|N) = \sum_{S \subseteq N \setminus \{i\}} \left( \frac{n - |S| - 1)!|S|!}{n!} \right) \left[ v(S \cup \{i\}) - v(S) \right] \tag{1}
\]

Weber (1988) have proved that the Shapley value is the only reward with the following axioms.

**Linearity axiom:** If the reward of a game \( u \) satisfies \( u(S) = v(S) + w(S) \), where \( v \) and \( w \) are another two games. Then the Shapley value of each player \( i \in N \) in the game \( v \) is the sum of Shapley values of the player \( i \) in the game \( v \) and \( w \), i.e. \( \phi_v(i|N) = \phi_u(i|N) + \phi_w(i|N) \).
**Dummy axiom:** The dummy player is defined as the player that satisfies $\forall S \subseteq N \setminus \{i\}$, $v(S \cup \{i\}) = v(S) + v(\emptyset)$. In this way, the dummy player $i$ satisfies $v(\{i\}) = \phi_v(i|N)$, i.e. the dummy player has no interaction with other players in $N$.

**Symmetry axiom:** If $\forall S \subseteq N \setminus \{j, i\}$, $v(S \cup \{i\}) = v(S \cup \{j\})$, then $\phi_v(i|N) = \phi_v(j|N)$.

**Efficiency axiom:** $\sum_{i \in N} \phi_v(i|N) = v(N)$. The efficiency axiom can ensure the overall reward can be distributed to each player in the game.

### 3 Algorithm

#### 3.1 Multivariate interactions in game theory

Given a game with $n$ players $N = \{1, 2, \ldots, n\}$, in this paper, we define the interaction among multiple players based on game theory. Specifically, we focus on the significance of interactions of a set of $m$ players selected from all players. Let $2^N$ denote all potential subsets of $N$, e.g. if $N = \{a, b\}$, then $2^N = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$. The game $v : 2^N \rightarrow \mathbb{R}$ is a set function that maps a set of players to a scalar value. We consider this scalar value as the reward for the set of players. For example, we consider the DNN with a scalar output as a game, and consider the input variables as a set of players. In this way, $v(S), S \subseteq N$, represents the output of the DNN when all input variables in $N \setminus S$ are replaced by the baseline value. In this paper, the baseline value is set as zero, which is similar to [Ancona et al. 2019]. Please see supplementary materials for discussions about the selection of the baseline value. If the DNN is trained for multi-category classification, $v(S)$ is implemented as the score of the true category before the softmax layer. The overall reward can be represented as $v(N) - v(\emptyset)$, i.e. the score obtained by all input variables subtract the score obtained by the zero input. The overall reward can be allocated to each player as $\sum_{i=1}^{n} \phi_v(i|N) = v(N) - v(\emptyset)$, where $\phi_v(i|N)$ denotes the allocated reward of the player $i$ over the set $N$. Each player is supposed to obtain a high reward in the game. In this paper, we use $\phi(i|N)$ to represent $\phi_v(i|N)$ in the following manuscript for simplicity without causing ambiguity. Specifically, $\phi(i|N)$ can be computed as the Shapley value of $i$ based on Equation equation 1. The Shapley value is an unbiased method to allocate the reward, so we use the Shapley value to define the interaction.

**Interaction between two players:** Let us first consider the interaction between two players in the game $v$. Given two players $i$ and $j$, $\phi(i|N)$ and $\phi(j|N)$ represent their Shapley values, respectively. If players $i$ and $j$ always participate in the game together and always be absent together, then they can be regarded to form a coalition. The reward obtained by the coalition is usually different from the sum of rewards when players $i$ and $j$ participate in the game individually. This coalition can be considered as one singleton player, denoted by $S_{ij}$. Note that the Shapley value of the coalition can also be computed using Equation equation 1 by replacing the player $i$ with $S_{ij}$. In this way, we can consider there are only $n - 1$ players in the game, including $S_{ij}$, and excluding players $i$ and $j$, $N' = N \setminus \{i, j\} \cup S_{ij}$. The interaction between players $i$ and $j$ is defined as the additional reward brought by the coalition $S_{ij}$ w.r.t. the sum of rewards of each player playing individually, i.e.

$$B(S_{ij}) = \phi(S_{ij}|N') - [\phi(i|N_i) + \phi(j|N_j)],$$

where $\phi(i|N_i), \phi(j|N_j)$, and $\phi(S_{ij}|N')$ are computed over the set of players $N_i = N \setminus \{j\}, N_j = N \setminus \{i\}$, and $N'$, respectively. (1) If $B(S_{ij}) > 0$, then players $i$ and $j$ cooperate with each other for a higher reward, i.e. the interaction is positive. (2) If $B(S_{ij}) < 0$, then the interaction between players $i$ and $j$ leads to a lower reward, i.e. the interaction is negative.

**Interaction among multiple players:** We can extend the definition of interactions to multiple players. Let us focus on a subset of players $A \subseteq N$. If players in $A$ always participate in the game together, then players in $A$ form a coalition, which can be considered as a singleton player, denoted by $|A|$. The interaction in $A$ can be measured as

$$B(|A|) = \phi(|A|\cup N_{\setminus |A|}) - \sum_{i \in A} \phi(i|N_i),$$

where $\phi(|A|\cup N_{\setminus |A|})$ is the Shapley value of the coalition $|A|$. We compute the Shapley value over the player set $N_{\setminus |A|} = N \setminus A \cup \{|A|\}$. Similarly, $\phi(i|N_i)$ represents the Shapley value computed over the set $N_i = N \setminus A \cup \{i\}$, where players in $A \setminus \{i\}$ are supposed not to attend the game. In this way, $B(|A|)$ reflects all interactions inside $A$, including both positive and negative interactions, which counteract each other. For example, let the set $A = \{i, j, k, l\}$, we assume that only the interaction
between players $i$ and $j$ and the interaction between players $k$ and $l$ are positive. Interactions of other coalitions are negative. Then, the positive interaction effects are counteracted by negative interaction effects in $B([A])$. We will introduce more details in Equation 6.

However, the significance of interactions is supposed to reflect both positive and negative interaction effects. Therefore, we hope to propose a new metric to measure the significance of interactions, which contains both positive and negative effects among a set of players $A \subseteq N$. We aim to divide all players in $A$ into a number of coalitions $\Omega = \{C_1, C_2, \ldots, C_k\}$, which are ensured to mainly encode positive interaction effects. $\Omega$ is a partition of the set $A$. In the above example, the four players $\{i, j, k, l\}$ is supposed to be divided into $\Omega = \{C_1 = \{i, j\}, C_2 = \{k, l\}\}$. In this case, the following equation can better reflect positive interaction effects than Equation 5 when we measure the reward by taking each coalition $C_i$ as a singleton player.

$$B_{\text{max}}([A]) = \max_{\Omega} \sum_{C \in \Omega} \phi(C|N_C) - \sum_{i \in A} \phi(i|N_i)$$

where $\phi(C|N_C)$ is computed over $N_C = N \setminus A \cup \{\{C\}\}$, and $\phi(i|N_i)$ is computed over $N_i = N \setminus A \cup \{i\}$. Similarly, we can use $B_{\text{max}}([A]) = \min_{\Omega} \sum_{C \in \Omega} \phi(C|N_C) - \sum_{i \in A} \phi(i|N_i)$ to roughly quantify the negative interaction effects inside the set $A$. Thus, we define the metric $T([A])$ to measure the significance of both positive and negative interaction effects, as follows.

$$T([A]) = B_{\text{max}}([A]) - B_{\text{min}}([A]) = \max_{\Omega} \sum_{C \in \Omega} \phi(C|N_C) - \min_{\Omega} \sum_{C \in \Omega} \phi(C|N_C)$$

**Relationship between $T([A])$ and $B([A])$:** Theoretically, the interaction $B([A])$ in Equation 3 can be decomposed into elementary interaction components $I(A)$, which was firstly proposed by Grabisch & Roubens (1999). $I(A)$ quantifies the marginal reward of $A$, which removes all marginal rewards from all combinations of $A$. We derive the following equation to encode the relationship between $B([A])$ and $I(A)$. Supplementary materials provide more information about the elementary interaction component, as well as some extended properties of the interaction metric $B([A])$.

$$B([A]) = \sum_{A' \subseteq A, |A'| > 1} I(A').$$

According to Equation 6, there are a total of $2^m - m - 1$ different elementary interaction components inside the interaction $B([A])$, where $m$ represents the number of players inside $A$. Some of these interaction components are positive, and others are negative, which leads to a counteraction among all possible interaction components. In order to quantify the significance of all interaction components, a simple idea is to sum up the absolute values of all elementary interaction components, as follows. $B'([A]) = \sum_{A' \subseteq A, |A'| > 1} I(A') = B^+ - B^-$, where $B^+ = \sum_{A' \subseteq A, I(A') > 0} I(A')$ and $B^- = \sum_{A' \subseteq A, I(A') < 0} I(A')$, subject to $|A'| > 1$. In Equation 4, $B_{\text{max}}([A])$ is supposed to mainly encode most positive interaction effects within $A$, which is highly related to $B^+$. Similarly, $B_{\text{min}}([A])$ is highly related to $B^-$ for mainly encoding negative interaction effects within $A$. Therefore, $T([A]) = B_{\text{max}}([A]) - B_{\text{min}}([A])$ is highly related to $B'([A]) = B^+ - B^-$. Please see supplementary materials for more discussions about the relationship between $T([A])$ and $B'([A])$.

**Why we need a single metric?** Grabisch & Roubens (1999) designed $I(A)$, which computed $2^m$ different values of $I(A')$ for all possible $A' \subseteq A$. In comparison, the proposed $B([A])$ is a single metric to represent the overall interaction significance among all $m$ input variables, which provides a global view to explain DNNs.

### 3.2 Explanation of DNNs using multivariate interactions in NLP

In this paper, we use the interaction based on game theory to explain DNNs for NLP tasks. Given an input sentence with $n$ words $N = \{1, 2, \ldots, n\}$, each word in the sentence can be considered as a player, and the output score of DNN can be taken as the game score $v(S)$. $v(S)$ represents the output of DNN when we mask all words in the set $N \setminus S$, i.e. setting word vectors of masked words to zero vectors. For the DNN with a scalar output, $v(S)$ denotes the output of the DNN. If the DNN was trained for the multi-category classification task, we take the output of the true category before the softmax layer as $v(S)$. Strongly interacted words usually cooperate with each other and form a prototype feature, which is memorized by the DNN for inference. Thus, the multivariate interaction can be used to analyze prototype features in DNNs.
Among all input words in \( N \), we aim to quantify the significance of interactions among a subset of \( m \) sequential words \( A \subseteq N \). \[1\] Chen et al. (2019) showed that non-successive words usually contain much fewer interactions than successive words. Thus, we require each coalition consists of several sequential words to simplify the implementation. Although \( T(|A|) \) in Equation \[5\] can be computed by enumerating all possible partitions in \( A, \Omega \), the computational cost of such a method is too unaffordable.

Therefore, we develop a sampling-based method to efficiently approximate \( T(|A|) \) in Equation \[5\]. The computation of \( \max_{\Omega} \sum_{C \in \Omega} \phi(C|N_C) \) requires us to enumerate all potential partitions to determine the maximal value. In order to avoid such computational expansive enumeration, we propose to use \( p = \{p_1, p_2, \cdots, p_{m-1}\} \) to represent all possible partitions. \( p_i (0 \leq p_i \leq 1) \) denotes the probability of the \( i \)-th word and the \((i+1)\)-th word belonging to the same coalition. We can sample \( g = \{g_1, g_2, \cdots, g_{m-1}\} \) based on the \( p \), i.e. \( g_i \in \{0, 1\}, g_i \sim \text{Bernoulli}(p_i) \), to represent a specific partition \( \Omega \). \( g_i \) indicates the \( i \)-th word and the \((i+1)\)-th word belong to the same coalition. \( g_i = 0 \) represents these words belong to different coalitions. In this way, we can use \( \max_p \mathbb{E}_{g \sim \text{Bernoulli}(p)} \sum_{C \in \Omega_g} \phi(C|N_C) \) to approximate \( \max_{\Omega} \sum_{C \in \Omega} \phi(C|N_C) \), where \( \Omega_g \) denotes the partition determined by \( g \).

In addition, \( \phi(C|N_C) \) can be approximated using a sampling-based method \[2\] Castro et al. (2009).

\[
\phi(C|N_C) = \mathbb{E}_{r \sim \text{Bernoulli}(p)} \left( \frac{\mathbb{E}_{S \subseteq N_C} [v(S \cup C) - v(S)]}{|S| = r} \right) 
\]

\[(7)\]

where \( r \) represents the number of players fed into the DNN, \( N_C = N \setminus A \cup \{C\} \). In this way, we can approximate \( \max_{\Omega} \sum_{C \in \Omega} \phi(C|N_C) \) as \( \max_p \mathbb{E}_{g \sim \text{Bernoulli}(p)} \sum_{C \in \Omega_g} \phi(C|N_C) \), as follows.

\[
\mathcal{L} = \max_{g \sim \text{Bernoulli}(p)} \sum_{C \in \Omega_g} \phi(C|N_C), \quad \max \mathcal{L} = \max_p \sum_{i \in A} \mathbb{E}_{g \sim \text{Bernoulli}(p)} \left( \frac{\mathbb{E}_{g \sim \text{Bernoulli}(p)}}{g_i=1} \left( \mathbb{E}_{S \subseteq \text{Sam}(\Omega_g)} [v(S)] - \mathbb{E}_{S \subseteq \text{Sam}(\Omega_g)} [v(S)] \right) \right)
\]

\[
\frac{\partial \mathcal{L}}{\partial p_i} = \sum_{j \in A} \mathbb{E}_{g \sim \text{Bernoulli}(p)} \left( \frac{\mathbb{E}_{g \sim \text{Bernoulli}(p)}}{g_i=0} \mathbb{E}_{S \subseteq \text{Sam}(\Omega_g)} [v(S)] - \mathbb{E}_{S \subseteq \text{Sam}(\Omega_g)} [v(S)] \right)
\]

\[(8)\]

\[(9)\]

where \( \lambda_i(g) = \frac{1}{n_i g_i} \), and \( S \in \text{Sam}(\Omega_g) \) represents the sampled set of words, which contains all words in \( N \setminus A \) and the coalition \( C_i, C_i \in \Omega_g \) denotes the coalition determined by \( g \) that contains the word \( i \). We learn \( p \) to maximize the above equation. For example, let \( A = \{x_1, x_2, x_3, x_4, x_5, x_6\} \), and the sampled \( g = \{g_1 = 1, g_2 = 1, g_3 = 0, g_4 = 0, g_5 = 1\} \). We consider the first three words in \( A \) as a coalition, and the last two words in \( A \) as another coalition, i.e. \( \Omega_g = \{C_1 = \{x_1, x_2, x_3\}, C_2 = \{x_4\}, C_3 = \{x_5, x_6\}\}. \) The set \( S \) is supposed to be sampled over these coalitions in \( \Omega_g \), instead of over individual words. In this case, we have \( \lambda_1(g) = \lambda_2(g) = \lambda_3(g) = 1/3, \lambda_4(g) = 1, \lambda_5(g) = \lambda_6(g) = 1/2 \). Please see supplementary materials for the physical meaning of Equation \[8\] and more discussions about the computation. In order to learn \( p \), we compute \( \partial \mathcal{L}/\partial p_i \), which is given in Equation \[10\]. In this way, we can use \( \max_p \mathcal{L} \) in Equation \[10\] to approximate the solution to \( \max_{\Omega} \sum_{C \in \Omega} \phi(C|N_C) \). Similarly, we can use \( \min_p \mathcal{L} \) to approximate the solution to \( \min_{\Omega} \sum_{C \in \Omega} \phi(C|N_C) \), thereby obtaining \( T(|A|) = \max_{\Omega} \sum_{C \in \Omega} \phi(C|N_C) \). 

**Comparison of computational cost:** Given a set of \( m \) words in a sentence with \( n \) words, the computation of \( T(|A|) \) based on its definition in Equation \[5\] and Equation \[10\] is NP-hard, i.e. \( O(2^m n) \). Instead, we propose a polynomial method to approximate \( T(|A|) \) with computational cost \( O(K_1 K_2 K_3 n m) \), where \( K_1 \) denotes the number of updating the probability \( p_{\Omega_g} \).

---

\[1\] Please see supplementary materials for the proof.
$K_2$ and $K_3$ represent the sampling number of $p \sim \text{Bernoulli}(p)$ and $S \in \text{Sam}(\Omega_g)$, respectively. In addition, we conducted experiments to show the accuracy of the estimation of $T([A])$ increases along with the increase of the sampling number.

4 Experiments

Evaluation of the correctness of the estimated partition of coalitions: The core challenge of evaluating the correctness of the partition of coalitions was that we had no ground-truth annotations of inter-word interactions, which were encoded in DNNs. To this end, we constructed three new datasets with ground-truth partitions between input variables, i.e. the Addition-Multiplication Dataset, the AND-OR Dataset, and the Exponential Dataset.

- The Addition-Multiplication Dataset contained 10000 addition-multiplication models, each of which only consisted of addition operations and multiplication operations. For example, $y = f(x) = x_1 + x_2 \times x_3 + x_4 \times x_5 + x_6 + x_7$. Each variable $x_i$ was a binary variable, i.e. $x_i \in \{0, 1\}$. For each model, we selected a number of sequential input variables to construct $A$, e.g. $A = \{x_2, x_3, x_4, x_5\}$. We applied our method to extract the partition of coalitions w.r.t. $y$, which maximized $\sum_{C \in \Omega} \phi(C|N_C)$.

- The AND-OR Dataset contained 10000 AND-OR models, each of which only contained AND operations and OR operations. For simplicity, we used & to denote the AND operation, and used | to denote the OR operation. For example, $y = f(x) = x_1 \mid (x_2 \& x_3) \mid (x_4 \& x_5) \mid x_6 \mid x_7$. Each variable $x_i$ was a binary variables, i.e. $x_i \in \{0, 1\}$. For each model, we selected a number of sequential input variables to construct $A$, e.g. $A = \{x_2, x_3, x_4, x_5\}$. We used our method to extract the partition of coalitions, which maximized $\sum_{C \in \Omega} \phi(C|N_C)$.

- The Exponential Dataset contained 10000 models, each of which contained exponential operations and addition operations. For example, $y = f(x) = x_1^2 + x_3^4 + x_5 + x_6$. Each variable $x_i$ was a binary variables, i.e. $x_i \in \{0, 1\}$. For each model, we selected a number of sequential input variables to construct $A$, which always contained the exponential operations, e.g. $A = \{x_1, x_2, x_3, x_4\}$. We used our method to extract the partition of coalitions, which maximized $\sum_{C \in \Omega} \phi(C|N_C)$.

We compared the extracted partition of coalitions with the ground-truth partition of coalitions. Variables in multiplication operations and AND operations usually had positive interactions, which were supposed to be allocated into one coalition during $\max_{\Omega} \sum_{C \in \Omega} \phi(C|N_C)$. Variables in OR operations had negative interactions, which were supposed to be allocated to different coalitions during $\max_{\Omega} \sum_{C \in \Omega} \phi(C|N_C)$. Note that the addition operation did not cause any interactions. Therefore, we did not consider interactions of addition operations when we evaluated the correctness of the partition. In the above example of addition-multiplication model, if we considered the set $A = \{x_2, x_3, x_4, x_5\}$, then the ground-truth partition was supposed to be either $\{\{x_2, x_3\}, \{x_4, x_5\}\}$ or $\{\{x_2, x_3, x_4, x_5\}\}$. For the above example of AND-OR model and $A = \{x_2, x_3, x_4, x_5\}$, the ground-truth partition should be $\{\{x_2, x_3\}, \{x_4, x_5\}\}$. For each operation, if the proposed method allocated variables of an operation in the same way as the ground-truth partition, then we considered the operation was correctly allocated by the proposed method. The average rate of correctly allocated operations over all operations was reported in Table[1].

We also designed three baselines. Baseline 1 randomly combined input variables to generate coalitions. Lundberg et al. [2018] defined the interaction between two input variables, which was used as the Baseline 2. Li et al. [2016] proposed a method to estimate the importance of input variables, which was taken as Baseline 3. Baseline 2 and Baseline 3 merged variables whose interactions were greater than zero to generate coalitions.

Table[1] compares the accuracy of coalitions generated by our method and that of other baselines. Our method achieved a better accuracy than other baselines. Figure[2] shows the change of the probability $p_i$ during the training process. Our method successfully merged variables into a single coalition in multiplication operations and AND operations. Besides, our method did not merge variables in OR operations. Note that the probability $p_3$ of the addition operation in Figure[2] did not converge. It was because the addition operation did not cause any interactions, i.e. arbitrary value of $p_3$ satisfied the ground-truth.

Evaluation of the accuracy of $T([A])$: We applied our method to DNNs in NLP to quantify the interaction among a set of input words. We trained DNNs for two tasks, i.e. binary sentiment
Table 1: Accuracy of the estimated partition.

| Dataset | Add-Multiple | AND-OR | Exponential |
|---------|--------------|-------|-------------|
| Baseline 1 | 0.500 | 0.503 | 0.506 |
| Baseline 2 | 1.000 | 0.996 | 1.000 |
| Baseline 3 | 1.000 | 0.523 | 0.846 |
| Our method | **1.000** | **0.999** | **1.000** |

We compared the extracted significance of interactions \( T([A]) \) with the accurate significance of interactions. The accuracy of interactions was quantified based on the definition in Equation 5 and Equation 1, which was computed by enumerating all partitions \( \Omega \) and all subsets \( S \) with an extremely high computational cost. Considering the unaffordable computational cost, such evaluation could only be applied to sentences with less than 12 words. Figure 5 (a) reports the error of \( T([A]) \), i.e., \( |T_{\text{true}}([A]) - T([A])| \), where \( T_{\text{true}}([A]) \) was the true interaction significance accurately computed via massive enumerations. We found that the estimated significance of interactions was accurate enough after the training of 100 epochs.

**Stability of \( T([A]) \):** We also measured the stability of \( T([A]) \), when we computed \( T([A]) \) multiple times with different sampled sets of \( g \) and \( S \). The instability of \( T([A]) \) was computed as instability = \( \mathbb{E}_f \left[ \frac{1}{|\Omega|} \sum_{u \in \Omega} \left( T_{(u)}([A]) - T_{(\omega)}([A]) \right) \right] \), where \( T_{(u)}([A]) \) denotes the \( u \)-th computation result of \( T([A]) \). Figure 5 (b) shows the instability of interactions in different DNNs and datasets. Experimental results showed that interactions in CNN and ELMo converged more quickly. Moreover, instability decreased quickly along with the increase of the sampling numbers. Our metric was stable enough when the number of sampling was larger than 2000.

**Prototype features modeled by DNNs:** We analyzed results obtained by the proposed method to explore prototype features modeled by DNNs. Figure 4 shows several results on the BERT. We found that: (1) For an entire constituent in the sentence, such as a short clause or a noun phrase, the BERT usually took the whole constituent as a single coalition, i.e., a prototype feature. (2) For the set of words that contained conjunctions or punctuations, such as “and”, “or”, “,”, etc., the BERT divided the set at conjunctions or punctuations to generate different coalitions (prototype features). (3) For the constituent modified by multiple adjectives and adverbs, the BERT usually merged all adjectives and adverbs into one coalition, and took the modified constituent as another coalition. Such phenomena fit the syntactic structure parsing according to human knowledge.

**Interactions w.r.t. the intermediate-layer feature:** We could also compute the significance of interactions among a set of input words w.r.t. the computation of an intermediate-layer feature \( f \). The reward was computed using the intermediate-layer feature. Let \( f_N \) and \( f_S \) represent...
...most intriguing explorations of alienation. ...is a compliment to kuras and miller. 

a solid examination of the male midlife crisis. ...overall very good for what it’s trying to do. 

the best film about 2 baseball to hit theaters since field of dreams. ...a good piece of work more often than not. 

the film will ...play equally well ...on both the standard and giant screens. ...a movie fans. get ready to take off. ...the other direction. 

it doesn’t ...believe in itself, it has no sense of humor... ...it’s just plain bored. 

it’s a remarkably solid and subtly satirical/tout de force. ...it’s a charming and often affecting journey. 

this is a good script, good dialogue... ...it’s actually pretty good in the first few minutes. 

dull, lifeless, and amateurness assembles. ...it actually pretty good in the first few minutes. 

a warm but realistic meditation on friendship, family and affection. ...it’s actually pretty good in the first few minutes. 

no telegraphing is too obvious or simplistic for this movie... just as moving... ...uplifting and funny as ever. 

a tender, witty, captivating film about friendship, love, ...memory, trust and loyalty. 

a dumb movie with dumb characters doing dumb things and you have to be really dumb not to see where this is going... makes for a pretty unpleasant viewing experience... 

a strangely compelling and brilliantly acted psychological drama... a tender, heartwelt family drama... 

preaches to two completely different choirs at the same time, which is a pretty amazing accomplishment... 

Figure 4: Prototype features modeled by the BERT using the SST-2 dataset. Red brackets indicated words in A. We used different colors to indicate the extracted coalitions. Words in the same coalition formed a prototype feature modeled by the DNN. Words in the same coalition formed a prototype feature modeled by the DNN. The set of words A was randomly selected. 

maximum (prototypes towards incorrect predictions): if steven soderbergh’s ‘solaris’ is a failure ...label: positive predict: negative 

minimum (prototypes towards correct predictions): the longer the movie goes, the worse it gets... ...label: negative predict: positive 

minimum (prototypes towards correct predictions): the longer the movie goes, the worse it gets... ...label: negative predict: positive 

minimum (prototypes towards incorrect predictions): on the heels of the ring comes a similarly morose and humorless horror movie... that, although flawed, is to be recommended for its straight-ahead approach to creepiness... 

minimum (prototypes towards correct predictions): on the heels of the ring comes a similarly morose and humorless horror movie... that, although flawed, is to be recommended for its straight-ahead approach to creepiness... 

minimum (prototypes towards correct predictions): on the heels of the ring comes a similarly morose and humorless horror movie... that, although flawed, is to be recommended for its straight-ahead approach to creepiness... 

Figure 5: Prototype features modeled by the BERT trained using the SST-2 dataset. Grey box indicated words in A. We used different colors to indicate the extracted coalitions. 

intermediate-layer features obtained by the set of input variables N and S, respectively. Since the intermediate-layer feature usually can be represented as high dimensional vectors, instead of a scalar value. We computed the output \( v(S) \) as \( \psi_f(S) = \langle f_N, f_S \rangle / \| f_N \|_2 \), where \( \| f_N \|_2 \) was the L2-norm and was used for normalization. Figure 5(c) shows the significance of interactions computed with output features of different layers. The significance of interactions among a set of input variables increased along with the layer number. This phenomenon showed that the prototype feature was gradually modeled in deep layers of the DNN. Supplementary materials provided more examples of prototype features extracted by the proposed method and baselines. 

Mining prototype features towards incorrect predictions of DNNs: The proposed method could be used to analyze sentences, which were mistakenly classified by the DNN. We used the BERT learned using the SST-2 dataset. Figure 5 shows two partitions that maximized and minimized the Shapley value of coalitions for each sentence. For the partition that maximized the Shapley value, the generated coalitions were usually used to explain prototype features toward incorrect predictions of the DNN. In comparison, coalitions generated by minimizing the Shapley value represented prototype features towards correct predictions. For example, we tested two different sets of input variables \( A \) for the sentence “on the heels of ... to creepiness.”. This sentence was incorrectly predicted to be negative by the DNN. As Figure 5 shows, the DNN used “humorless horror movie” as a prototype features, which led to the incorrect prediction. Besides, when we minimized the Shapley value, we got the coalition “is to be recommended for its straight-ahead approach to creep” as the evidence towards the positive label. 

5 Conclusion 

In this paper, we have defined the multivariate interaction in the DNN based on game theory. We quantify the interaction among multiple input variables, which reflects both positive and negative
interaction effects inside these input variables. Furthermore, we have proposed a method to approximate the interaction efficiently. Our method can be applied to various DNNs for different tasks in NLP. Note that the quantified interaction is just an approximation of the accurate interaction in Equation equation [5]. Nevertheless, experimental results have verified high accuracy of the approximation. The proposed method can extract prototype features modeled by the DNN, which provides a new perspective to analyze the DNN.

**ETHICAL IMPACT**

This study has broad impacts on the understanding of signal processing in DNNs. Our work provides researchers in the field of explainable AI with new mathematical tools to analyze DNNs. Currently, existing methods mainly focus on interactions between two input variables. Our research proposes a new metric to quantify interactions among multiple input variables, which sheds new light on the understanding of prototype features in a DNN. We also develop a method efficiently approximate such interactions. As a generic tool to analyze DNNs, we have applied our method to classic DNNs and have obtained several new insights on signal processing encoded in DNNs for NLP tasks.

**REFERENCES**

Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *The Journal of Machine Learning Research*, 19(1):1947–1980, 2018.

Marco Ancona, Cengiz Oztireli, and Markus Gross. Explaining deep neural networks with a polynomial time algorithm for shapley values approximation. *arXiv:1903.10992*, 2019.

Jacob Bien, Jonathan Taylor, and Robert Tibshirani. A lasso for hierarchical interactions. *Annals of Statistics*, 41(3):1111–1141, 2013.

Alexander Binder, Gregoire Montavon, Sebastian Lapuschkin, Klaus Robert Muller, and Wojciech Samek. Layerwise relevance propagation for neural networks with local renormalization layers. In *International Conference on Artificial Neural Networks (ICANN)*, pp. 63–71, 2016.

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

Jianbo Chen, Le Song, Martin J. Wainwright, and Michael I. Jordan. L-Shapley and C-Shapley: Efficient model interpretation for structured data. In *ICLR*, 2019.

Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. Hierarchical multiscale recurrent neural networks. In *ICLR*, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Kedar Dhamdhere, Ashish Agarwal, and Mukund Sundararajan. The shapley taylor interaction index. *arXiv:1902.05622*, 2019.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A Smith. Recurrent neural network grammars. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 199–209, 2016.

Stanislav Fort, Paweł Krzysztof Nowak, Stanislaw Jastrzebski, and Srin Narayanan. Stiffness: A new perspective on generalization in neural network. *arXiv preprint arXiv:1901.09491*, 2019.

Michel Grabisch and Marc Roubens. An axiomatic approach to the concept of interaction among players in cooperative games. *International Journal of Game Theory*, 28:547–565, 1999.

Chaoyu Guan, Xiting Wang, Quanshi Zhang, Runjin Chen, Di He, and Xing Xie. Towards a deep and unified understanding of deep neural models in nlp. In *ICML*, pp. 2454–2463, 2019.

Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
Joseph D Janizek, Pascal Sturmefels, and Suin Lee. Explaining explanations: Axiomatic feature interactions for deep networks. arXiv:2002.04138, 2020.

Xisen Jin, Junyi Du, Zhongyu Wei, Xiangyang Xue, and Xiang Ren. Towards hierarchical importance attribution: Explaining compositional semantics for neural sequence models. arXiv:1911.06194, 2019.

Yoon Kim. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882, 2014.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and understanding neural models in nlp. In Annual Conference of the North American Chapter of the Association for Computational Linguistics, 2015.

Jiwei Li, Will S Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. arXiv: 1612:08220, 2016.

Ruohan Liang, Tianlin Li, Longfei Li, Jing Wang, and Quanshi Zhang. Knowledge consistency between neural networks and beyond. In ICLR, 2019.

Scott Lundberg, Gabriel G Erion, and Suin Lee. Consistent individualized feature attribution for tree ensembles. arXiv: Learning, 2018.

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

W James Murdoch, Peter J Liu, and Bin Yu. Beyond word importance: Contextual decomposition to extract interactions from lstms. arXiv:1801.05453, 2018.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. arXiv preprint arXiv:1802.05365, 2018.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “why should i trust you?” explaining the predictions of any classifier. In KDD, 2016.

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

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. Ordered neurons: Integrating tree structures into recurrent neural networks. In ICLR, 2019.

Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. In arXiv:1605.01713, 2016.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In arXiv:1312.6034, 2013.

Chandan Singh, W James Murdoch, and Bin Yu. Hierarchical interpretations for neural network predictions. arXiv:1806.05337, 2018.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 conference on empirical methods in natural language processing, pp. 1631–1642, 2013.

Daria Sorokina, Rich Caruana, Mirek Riedewald, and Daniel Fink. Detecting statistical interactions with additive groves of trees. In ICML, pp. 1000–1007, 2008.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. In arXiv:1412.6806, 2014.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In ICML, pp. 3319–3328, 2017.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. In International Joint Conference on Natural Language Processing, volume 1, pp. 1556–1566, 2015.

Michael Tsang, Dehua Cheng, and Yan Liu. Detecting statistical interactions from neural network weights. In ICLR, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in neural information processing systems, pp. 5998–6008, 2017.
Xing Wang, Zhaopeng Tu, Longyue Wang, and Shuming Shi. Self-attention with structural position representations. In *International Joint Conference on Natural Language Processing*, pp. 1403–1409, 2019a.

Yau-Shian Wang, Hung-Yi Lee, and Yun-Nung Chen. Tree transformer: Integrating tree structures into self-attention. *arXiv preprint arXiv:1909.06639*, 2019b.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*, 2018.

Robert J Weber. Probabilistic values for games. *The Shapley Value. Essays in Honor of Lloyd S. Shapley*, pp. 101–119, 1988.
A  RELATIONSHIP BETWEEN $B([A])$ AND ELEMENTARY INTERACTION COMPONENTS $I(A)$

In Lines 300-328 of the section “multivariate interactions in game theory” in the paper, we briefly introduce the elementary interaction component $I(A)$ (Grabisch & Roubens 1999) and the relationship between $B([A])$ and $I(A)$. In this section, we will introduce more details about elementary interaction components and prove the relationship between $B([A])$ and $I(A)$. The elementary interaction component was firstly proposed by Grabisch & Roubens (1999). Given a game $v$, and the set of all players $N$. For a set of players $A \subseteq N$, the elementary interaction component $I(A)$ is computed as follows.

$$I(A) = \sum_{T \subseteq N \setminus A} \frac{(|N| - |T| - |A|)!|T|!}{(|N| - |A| + 1)!} \sum_{L \subseteq A} (-1)^{|A| - |L|} v(L \cup T), \quad \forall A \subseteq N.$$  

Note that for a singleton player, the elementary interaction component is equal to its Shapley value. According to Grabisch & Roubens (1999), the elementary interaction component satisfies the recursive axiom. I.e. if we consider the set of players $A$ as a singleton player $[A]$, then for every $|A| > 1$, $A \subseteq N$, we have

$$I(A) = I_{[A]}([A]) - \sum_{K \subseteq A, A \neq \emptyset} I_{N \setminus K} (A \setminus K)$$

$$= I_{[A]}([A]) - \sum_{A' \subseteq A, A \neq \emptyset} I_{N \setminus A'} (A')$$

where $N_{[A]} = N \setminus A \cup \{[A]\}$, and $N_{A'} = N \setminus A \cup A'$. In the section “multivariate interactions in game theory”, we use $B([A])$ to measure the interaction among all players in the set $A$. In this way, we can establish the relationship between $B([A])$ and $I(A)$ based on the above equation.

$$B([A]) = \phi_{[A]}(A) - \sum_{i \in A} \phi_i N_{[A]}(i)$$

$$= I_{[A]}([A]) - \sum_{i \in A} I_{N_{[A]}}(i)$$

$$= I(A) + \sum_{A' \subseteq A, A \neq \emptyset} I_{N_{[A']}}(A') - \sum_{i \in A} I_{N_{[A]}}(i)$$

$$= \sum_{A' \subseteq A, |A'| > 1} I_{N_{[A']}}(A')$$

Therefore, $B([A])$ can be decomposed into $2^m - m - 1$ elementary interaction components.

B  IMPORTANT PROPERTIES OF SHAPLEY VALUE AND INTERACTION

B.1  IMPORTANT PROPERTIES OF SHAPLEY VALUE

Weber (1988) proved that Shapley value is the only reward that satisfies the following properties.

**Linearity property:** If the reward of a game $u$ satisfies $u(S) = u(S) + w(S)$, where $v$ and $w$ are another two games. Then the Shapley value of each player $i \in N$ in the game $u$ is the sum of Shapley values of the player $i$ in the game $v$ and $w$, i.e. $\phi_u (i | N) = \phi_v (i | N) + \phi_w (i | N)$.

**Dummy property:** The dummy player is defined as the player that satisfies $\forall S \subseteq N \setminus \{i\}, v(S \cup \{i\}) = v(S) + v(\{i\})$. In this way, the dummy player $i$ satisfies $v(\{i\}) = \phi_v (i | N)$, i.e. the dummy player has no interaction with other players in $N$.

**Symmetry property:** If $\forall S \subseteq N \setminus \{i, j\}, v(S \cup \{i\}) = v(S \cup \{j\})$, then $\phi_v (i | N) = \phi_v (j | N)$.

**Efficiency property:** The sum of Shapley values of all players in the game equals to the reward when all players participate in the game $u$, i.e. $\sum_{i \in N} \phi_u (i | N) = v(N) - v(\emptyset)$. The efficiency axiom can ensure the overall reward can be distributed to each player in the game.

B.2  IMPORTANT PROPERTIES OF THE ELEMENTARY INTERACTION COMPONENT $I(A)$

The elementary interaction component $I(A)$ is firstly proposed by Grabisch & Roubens (1999). $I(A)$ quantifies the marginal reward of $A$, which removes all marginal rewards from all combinations of $A$. According to Grabisch & Roubens (1999), the elementary interaction component $I(A)$ satisfies following properties.
We further define multi-order Shapley value and multi-order interaction based on the Shapley value and the

\[ r \]

where the coalition \( B \) between the player \( i \) and \( j \), respectively. According to Castro et al. (2009) and Equation (7) in the

Note that the interaction between two players \( B \)

In this paper, we propose define the interaction among multiple input variables \( I \)

Recursive property 1: Let \( I(A|N) \) denote the elementary interaction component that computed with the set of players \( N \). We use \( |A| \) to represent the coalition that contains all players inside \( A \). Then we have \( I(A|N) = I([A]|N \setminus A \cup \{[A]\}) - \sum_{A' \subseteq A, A' \neq \emptyset} I(A'|N \setminus A \cup A') \).

Recursive property 2: The elementary interaction component in \( A \) is equal to the interaction among players \( A \setminus i \) with the presence of the player \( i \) minus the interaction among players \( A \setminus i \) with the absence of the player \( i \), i.e., \( \forall i \in A : I(A|N) = I_{w/o}(A \setminus \{i\}|N \setminus \{i\}) - I_{w/o}(A|N \setminus \{i\}) \). We use \( \Delta \) to denote the interaction among \( A \setminus \{i\} \) with the presence of \( i \), and \( \Delta_{w/o}(A \setminus \{i\}|N \setminus \{i\}) \) denotes the interaction among \( A \setminus \{i\} \) with the absence of \( i \).

B.3 IMPORTANT PROPERTIES OF THE MULTIVARIATE INTERACTION \( I(A) \)

In this paper, we propose define the interaction among multiple input variables \( B(I(A)) \), which is measure as Equation (3) in the paper. We can derive the relationship between \( B(I(A)) \) and the elementary interaction component \( I(A) \) as follows.

\[ B([A]) = \phi([A]|N_{[A]}) = \sum_{i \in A} \phi(i|N_i) = I([A]|N_{[A]}) - \sum_{i \in A} I(i|N_i) \% N_{[A]} = N \setminus A \cup \{[A]\}, N_i = N \setminus A \cup \{i\} \]

Based on the Recursive property 1 of \( I(A) \).

According to the relationship between \( B([A]) \) and \( I(A) \), we can get following properties of \( B([A]) \).

Linearity property: Given three different game \( v, w, \) and \( u \). We use \( B^w([A]), B^w([A]), \) and \( B^w([A]) \) to denote the interaction in set \( A \). If any subset of player \( S \subseteq N \), rewards of games \( u, v, \) and \( w \) satisfy \( u(S) = v(S) + w(S) \), then we have \( B_u(A) = B_v(A) + B_w(A) \).

Dummy property: If the player \( i \) is a dummy player, then \( \forall A \subseteq N \setminus \{i\} \), we have \( B([A \cup \{i\}] = B([A]) \).

Symmetry property: If \( \forall S \subseteq N \) and \( i \neq j \), there is \( v(S \cup \{i\}) = v(S \cup \{j\}) \), then \( \forall A \subseteq N \setminus \{i\} \), we have \( B([A \cup \{i\}] = B([A \cup \{j\}] \).

Note that the interaction between two players \( B(S_{ij}) \) is a special case of the multivariate interaction \( B([A]) \), where the coalition \( A \) contains two players \( i \) and \( j \). Therefore, \( B(S_{ij}) \) also satisfies above three properties.

B.4 IMPORTANT PROPERTIES OF MULTI-ORDER INTERACTION

We further define multi-order Shapley value and multi-order interaction based on the Shapley value and the interaction between two input variables, respectively. According to Castro et al. (2009) and Equation (7) in the paper, the Shapley value of the player \( i \) can be computed as follows.

\[ \phi(i|N) = E_r \left[ E_S \subseteq N \setminus \{i\}, |S| = r [v(S \cup \{i\}) - v(S)] \right] \]

where \( r \) is the number of players that participate in the game except \( i \). We define the \( r \)-order Shapley value of the player \( i \) as the reward obtained by \( i \) when there are \( r \) players participate in the game, as follows.

\[ \phi^r(i|N) = E_S \subseteq N \setminus \{i\}, |S| = r[v(S \cup \{i\}) - v(S)] \]

In this way, the Shapley value of \( i \) can be disentangled as \( \phi(i|N) = \sum_{r=1}^{n-1} \phi^r(i|N) = E_r[\phi^r(i|N)] \).

Similarly, the interaction \( B(S_{ij}) \) between players \( i \) and \( j \) can be computed as follows.

\[ B(S_{ij}) = E_r \left[ E_S \subseteq N \setminus \{i,j\}, |S| = r [\Delta v(S, i, j)] \right] \]

where \( \Delta v(S, i, j) = v(S \cup \{i,j\}) - v(S \cup \{i\}) - v(S \cup \{j\}) + v(S) \). We define the \( r \)-order interaction between \( i \) and \( j \) as follows.

\[ B^r(S_{ij}) = E_S \subseteq N \setminus \{i,j\}, |S| = r [\Delta v(S, i, j)] \]
In this way, the interaction between i and j can be disentangled as $B(S_{i,j}) = \sum_{r=0}^{n-2} [B^r(S_{i,j})/(n-1)] = \mathbb{E}_v[B^r(S_{i,j})]$. Similar to the interaction between i and j, the r-order interaction between i and j also satisfies the linearity property, dummy property and the symmetry property, as follows.

**Linearity property:** If the game $u$ satisfies $u(S) = u(S) + v(S)$, where $v$ and $w$ are another two games. Then, for all $i \in N$, $B^r(S_{i,j})$ can be decomposed into $B^r(S_{i,j}) = B^r_w(S_{i,j}) + B^r(S_{i,j})$.

**Dummy property:** The dummy pixel $i \in N$ satisfies $\forall S \subseteq N, v(S \cup \{i\}) = v(S) + v(\{i\})$. It means that the pixel i has no interactions with other pixels, i.e. $\forall j \in N, B^r(S_{i,j}) = 0$.

**Symmetry property:** If pixels $i, j \in N$ satisfy $\forall S \subseteq N \setminus \{i, j\}$, $v(S \cup \{i\}) = v(S \cup \{j\})$, then their contributions have same dependence on contexts, $\forall k \in N \setminus \{i, j\}$, $B^r(S_{i,k}) = B^r(S_{j,k})$.

Furthermore, we can prove that the r-order interaction between players i and j satisfies several more desirable properties, as follows.

**Marginal contribution property:** The difference of $(r+1)$-order Shapley value and r-order Shapley value can be represented as the average r-order interaction between i and all other players. I.e. if $\forall i, j \in N$, $i \neq j$, $\phi^{r+1}(i|N) - \phi^r(i|N) = \mathbb{E}_{j \in N \setminus \{i\}} [B^r(S_{i,j})]$. The proof is given as follows.

$$
\phi^{r+1}(i|N) - \phi^r(i|N) = E_{|S|={r+1}} \left[ E_{S \subseteq N \setminus \{i\}} [v(S \cup \{i\}) - v(S)] - E_{|S|={r}} \left[ E_{S \subseteq N \setminus \{i\}} [v(S \cup \{i\}) - v(S)] \right] \right]
$$

$$
= E_{|S|={r}} \left[ E_{S \subseteq N \setminus \{i\}} [v(S \cup \{i\}) - v(S)] \right] = \sum \frac{m!(n-1-m)!}{(n-1)!} \sum \frac{1}{n-1-m} [\Delta v(S, i, j)]
$$

$$
= \sum \frac{m!(n-m-2)!}{(n-1)!} \sum \frac{1}{n-m-1} [\Delta v(S, i, j)]
$$

$$
= \sum \frac{m!(n-m-2)!}{(n-1)!} \sum \frac{1}{n-m-1} [\Delta v(S, i, j)]
$$

$$
= E_{j \in N \setminus \{i\}} [B^r(S_{i,j})]
$$

**Accumulation property:** The contribution of $i \in N$ with contexts on m pixels can be decomposed into interactions dependent on less than m pixels. I.e. $\phi^r(i|N) = E_{j \in N \setminus \{i\}} \sum_{k=0}^{r-1} B^k(S_{i,j}) + \phi^0(i|N)$. The proof is given as follows.

$$
\phi^r(i|N) = \phi^r(i|N) - \phi^{r-1}(i|N) + \phi^{r-1}(i|N) - \phi^{r-2}(i|N) + \phi^{r-2}(i|N) + \cdots + \phi^0(i|N) + \phi^0(i|N)
$$

% Based on the marginal contribution property

$$
= \sum_{k=0}^{r-1} E_{j \in N \setminus \{i\}} [B^k(S_{i,j})] + \phi^0(i|N)
$$

$$
= E_{j \in N \setminus \{i\}} \left( \sum_{k=0}^{r-1} B^k(S_{i,j}) + \phi^0(i|N) \right)
$$
According to the accumulation property, when we set $\lambda$ to a single player that combines all players in $N \setminus \{i\}$. The proof is given as follows.

\[
\phi^{(n-1)}(i|N) = E_{j \in N\backslash \{i\}} [\sum_{k=0}^{n-2} B^k(S_{ij})] + \phi^0(i|N) \implies \phi^{(n-1)}(i|N) - \phi^0(i|N) = E_{j \in N\backslash \{i\}} [\sum_{k=0}^{n-2} B^k(S_{ij})]
\]

According to the definition of $r$-order Shapley value, we have

\[
\phi^{(n-1)}(i|N) - \phi^0(i|N) = v(N) - v(N \setminus \{i\}) - v(\emptyset)
\]

\[
= [v(N) - v(\emptyset)] - [v(N \setminus \{i\}) - v(\emptyset)] - [v(\emptyset) - v(\emptyset)]
\]

\[
= \phi([N]|N) - \phi([N \setminus \{i\}|N \setminus \{i\}) - \phi(\emptyset|\emptyset)
\]

\[
= B(S_{N\backslash \{i\},i})
\]

In this way, we have proved the third term in the efficiency property. Now we aim to prove that the last term is equal to the second term in the efficiency property.

\[
E_{j \in N\backslash \{i\}} [\sum_{k=0}^{n-2} B^k(S_{ij})] = \sum_{j \in N\backslash \{i\}} \frac{1}{n-1} \sum_{k=0}^{n-2} B^k(S_{ij})
\]

\[
= \sum_{j \in N\backslash \{i\}} E_r [I^r(S_{ij})]
\]

\[
= \sum_{j \in N\backslash \{i\}} B(S_{ij})
\]

Integration property: The contribution of the pixel $i$ can be decomposed into interaction components dependent on different contexts, i.e. $\phi(i|N) = \phi^0(i|N) + \frac{1}{n-1} \sum_{j \in N\backslash \{i\}} [\sum_{k=0}^{n-2} \frac{n-1-k}{n} B^k(S_{ij})]$. The proof is given as follows.

\[
\phi(i|N) = \sum_{r=0}^{n-1} \frac{1}{n} [\phi^r(i|N)]
\]

\[
= \sum_{r=0}^{n-1} \frac{1}{n} E_{j \in N\backslash \{i\}} [\sum_{k=0}^{r-1} B^k(S_{ij})] + \phi^0(i|N)
\]

\[
= E_{j \in N\backslash \{i\}} \left( \sum_{r=0}^{n-1} \sum_{k=0}^{r-1} \frac{1}{n} B^k(S_{ij}) \right) + \phi^0(i|N)
\]

\[
= E_{j \in N\backslash \{i\}} \left( \sum_{r=0}^{n-1} \frac{1}{n} B^k(S_{ij}) \right) + \phi^0(i|N)
\]

### C DISCUSSIONS ABOUT EQUATION (8) AND EQUATION (9) IN THE PAPER

In Lines 377-403 of the section “explanation of DNNs using multivariate interactions in NLP” in the paper, we propose a method to approximate $\max_{i} \sum_{C \in \Omega_S} \phi^C_{i|C}$ by optimizing $\mathcal{L} = \mathbb{E}_{g \sim \text{Bernoulli}(p)} \sum_{C \in \Omega_S} \phi^C_{i|C}$, which is given as follows.

\[
\mathcal{L} = \mathbb{E}_{g \sim \text{Bernoulli}(p)} \sum_{C \in \Omega_S} \phi^C_{i|C}, \quad \max_P = \max_P \mathbb{E}_r \left\{ \mathbb{E}_{g \sim \text{Bernoulli}(p)} \left[ \sum_{S \in \text{Sam}\left(\Omega_S, k\right), |S|=r+1} \mathbb{E}_{S \in \text{Sam}(\Omega_S)} \left[ v(S) \right] - \mathbb{E}_{S \in \text{Sam}(\Omega_S)} \left[ v(S) \right] \right] \right\}
\]

where $\Omega_S$ is the partition that determined by $g$, and $S \in \text{Sam}(\Omega_S)$ represents the set of the sampled players. $\lambda_i(g)$ is a scalar determined by $g$, and $\lambda_i(g) = 1/|C|$, where $|C|$ is the number of players in the coalition that contains $i$.

The reason for the computation of $\lambda_i(g)$ is given as follows. For the set $S \in \text{Sam}(\Omega_S)$, if the player $i \in S$ belongs to a coalition $C$, then all players in the same coalition $C$ are also supposed to be simultaneously sampled.
by $S$, since players in the same coalition always participate in the game together. Similarly, if $i \notin S$, then any player in the same coalition $C$ with $i$ is not supposed to be present in the set $S$. In this way, the term $\frac{E}{S \in \text{Sim}(\Omega_q)} \left\{ r(S) \right\}$ actually reflects the additional reward obtained by the entire coalition $C$ that contains the player $i$, instead of just the reward of the player $i$. According to above equations, if there are multiple players in the coalition $C$, then the additional reward obtained by $C$ will be computed for $|C|$ times in the above equation, where $|C|$ is the number of players in the coalition $C$. Thus, in order to approximate $\sum_{C \in \Omega} \phi_C^N$, accurately, we set $\lambda_i(g) = 1/|C|$ for normalization. For example, let $C = \{x_1, x_2, x_3\}$, if $x_1 \in S$, then $x_2 \in S$ and $x_3 \in S$, since they belong to a same coalition. Similarly, if $x_1 \notin S$, then $x_2 \notin S$ and $x_3 \notin S$. The additional reward obtained by $C$ is computed for three times. Hence, we set $\lambda_i(g) = \lambda_2(g) = \lambda_3(g) = 1/3$.

In order to optimize $L$ to approximate $\max_\Omega \sum_{C \in \Omega} \phi_C^N$, we need to train the probability $p$. For each $p_i \in p$, the gradient can be computed as follows.

$$\frac{\partial L}{\partial p_i} = \frac{\partial}{\partial p_i} \left\{ \sum_{j \in A} \left[ \frac{\mathbb{E}_{g \sim \text{Bernoulli}(p)} \left[ \frac{\mathbb{E}_{S \in \text{Sim}(\Omega_q)} \left[ r(S) \right]}{S \ni |S| = r + 1} \right] - \frac{\mathbb{E}_{S \in \text{Sim}(\Omega_q)} \left[ v(S) \right]}{S \ni |S| = r} \right] - \frac{\mathbb{E}_{g \sim \text{Bernoulli}(p)} \left[ \frac{\mathbb{E}_{S \in \text{Sim}(\Omega_q)} \left[ r(S) \right]}{S \ni |S| = r + 1} \right] - \frac{\mathbb{E}_{S \in \text{Sim}(\Omega_q)} \left[ v(S) \right]}{S \ni |S| = r} \right] \right\}$$

### D ANALYSIS OF THE COMPUTATIONAL COMPLEXITY

In Lines 404-414 of the section “explanation of DNNs using multivariate interactions in NLP” in the paper, we compare the computational complexity of our proposed metric, and the computational complexity of the computation based on Equation (5) and Equation (1) in the paper. In this section, we will show the estimation of the computational complexity.

**Original computation of $T([A])$ based on the definition is NP-hard**: Let us first consider the complexity of the computation based on Equation (5) and Equation (1) in the paper. For the convenience of readers, we rewrite these two equations as follows.

$$T([A]) = B_{\text{max}}([A]) - B_{\text{min}}([A]) = \max_{\Omega} \sum_{C \in \Omega} \phi_C^N - \min_{\Omega} \sum_{C \in \Omega} \phi_C^N$$

$$\phi_i^N = \sum_{S \subseteq N \setminus \{i\}} \frac{(n - |S| - 1)! |S|!}{n!} \left[ r(S \cup \{i\}) - v(S) \right]$$

For a sentence with $n$ words, we aim to quantify the significance of interactions among a set of $m$ words $A$. If each player only forms a coalition with its adjacent players, then there are $2^{m-1}$ possible partition plans for the set $A$. Thus, according to Equation (5) in the paper, we need to enumerate $2^{m-1}$ possible $\Omega$ to find $\max_{\Omega} \sum_{C \in \Omega} \phi_C^N$ and $\min_{\Omega} \sum_{C \in \Omega} \phi_C^N$. Given a specific $\Omega$ and a coalition $C \in \Omega$, we consider the coalition $C$ as a singleton player $[C]$. The Shapley value of the singleton player $[C]$, denoted by $\phi_C$, is computed over the set of players $N \setminus A \cup \{[C]\}$, i.e. we remove all $m$ players in $A$ from $N$, and add one singleton player $[C]$. Hence, there are total $n - m + 1$ players in the set $N_C$. According to Equation (1) in the paper, when we compute the accurate Shapley value of the player $[C]$, we need to consider $2^{n-m-1}$ possible subsets of $N_C$. Therefore, the computational complexity of the computation based on Equation (5) and Equation (1) is $O(2^{n-m-1}2^{m-1}) = O(2^nm)$, which increases exponentially along with the number of words $n$ in the sentence.

**In comparison, the computational cost of the proposed method is $O(K_1K_2K_3n^m)$**: As for the proposed method in the section “explanation of DNNs using multivariate interactions in NLP” of the paper, we approximate the significance of interactions $T([A])$ by sampling. We learn the probability $p$, as shown in Equation (8) and Equation (9). For the convenience of readers, we rewrite these two equations as follows.

$$\mathcal{L} = \frac{E}{g \sim \text{Bernoulli}(p)} \sum_{C \in \Omega_q} \phi_C^N, \max \mathcal{L} = \max_{p \in P} \sum_{i \in A} \left[ \frac{E}{g \sim \text{Bernoulli}(p)} \left[ \frac{E_{S \in \text{Sim}(\Omega_q)} \left[ r(S) \right]}{S \ni |S| = r + 1} \right] - \frac{E_{S \in \text{Sim}(\Omega_q)} \left[ v(S) \right]}{S \ni |S| = r} \right] \right\}$$

$$\frac{\partial \mathcal{L}}{\partial p_i} = \sum_{j \in A} \left[ \frac{E_{g \sim \text{Bernoulli}(p)}}{g_i = 0} \left[ \frac{E_{S \in \text{Sim}(\Omega_q)} \left[ r(S) \right]}{S \ni |S| = r + 1} \right] - \frac{E_{S \in \text{Sim}(\Omega_q)} \left[ v(S) \right]}{S \ni |S| = r} \right] \right\}$$
we show more examples of prototype features. We found that: (1) For an entire constituent in the sentence, we recursively select the value of variables.

In Figure 4 of the paper, we provide several examples of prototype features modeled by the DNN. In this section, less than or equal to was firstly proposed in Lines 318-328. Since the computation of have been widely used in Ancona et al. (2019). When we estimate the attribution of input variables and the

Figure 6: Relationship between $B'(\{A\})$ and $T(\{A\})$. We found that these two metrics were positively correlated.

The original equation in the paper, i.e. Equation (9) in the paper, has a small typo. We have corrected the typo in the above equation. Furthermore, we apply the sampling-based method to approximate the gradient as follows.

$$\frac{\partial L}{\partial p_i} = \sum_{j \in A} \sum_{r=0}^{n-1} \left\{ \frac{1}{K_2} \sum_{g^{(k)} = \text{Bernoulli}(p)} \lambda_i(g^{(k)}) \left[ \sum_{u=1}^{K_3} \sum_{g^{(n)} \in \text{Sam}(\Omega_g^{(k)})} [v(S)] - \frac{1}{K_3} \sum_{g^{(n)} \in \text{Sam}(\Omega_g^{(k)})} [v(S)] \right] \right\}$$

The computational cost of the above equation is $O(K_1 K_2 K_3 n m)$, which is shown as follows. Firstly, in order to compute $L$, we need to consider each player $i$ inside the set $A$, $1 \leq i \leq m$. Then, for a specific player $i \in A$, we recursively select the value of $r$ from 0 to $n-1$. Thirdly, given the player $i \in A$ and a specific value of $r$, we need to sample $g \sim \text{Bernoulli}(p)$ for $K_2$ times. For each $g$, we sample $S \in \text{Sam}(\Omega_g)$ for $K_3$ times to compute the gradient $\partial L / \partial p_i$. Last, in order to learn the probability $p$ that $\max_i \mathbb{E}_{g \sim \text{Bernoulli}(p)} \phi_C \sum_{C \in \Omega_g} \phi_C$, we update the probability $p$ for $K_1$ times. Thus, the computational cost of our proposed method is $O(K_1 K_2 K_3 n m)$.

E  SELECTION OF THE BASELINE VALUE

In this section, we introduce details about the selection of the baseline value. Several previous studies have discussed this problem. In this paper, we simply follow the settings that have been widely used in Ancona et al. (2019). When we estimate the attribution of input variables and the interaction among input variables, we need to remove several input variables. The removed variables correspond to players that do not participate in the game. However, it is usually difficult to actually remove an input variable from the input of the DNN. Thus, we usually need a baseline value to simulate the absence of input variables.

Setting an input variable to the baseline value indicates this input variable is removed. A canonical choice is to set the baseline value as zero. Given an input with all zero input variables, if all bias terms in a DNN are zero, then the output of the DNN is always zero. Besides, according to Ancona et al. (2019), the output of the DNN is close to zero even if bias terms are different from zero. Thus, it is an enough reasonable choice to select the baseline value as zero.

F  VERIFICATION OF THE RELATIONSHIP BETWEEN $T(\{A\})$ AND $B'(\{A\})$ IN LINES 318-328

In this section, we conducted an experiment to show the relationship between $T(\{A\})$ and $B'(\{A\})$, which was firstly proposed in Lines 318-328. Since the computation of $B'(\{A\})$ was NP-hard, we only selected the set with less than or equal to 4 words as $A$. We enumerated all potential interaction effects inside $A$ to compute $B'(\{A\})$. Figure 6 shows that $B'(\{A\})$ and $T(\{A\})$ were approximately positively correlated to each other. Therefore, we could use $T(\{A\})$ as an effective metric to measure the significance of interactions among multiple input variables.

G  MORE EXAMPLES OF PROTOTYPE FEATURES MODELED BY THE BERT

In Figure 4 of the paper, we provide several examples of prototype features modeled by the DNN. In this section, we show more examples of prototype features. We found that: (1) For an entire constituent in the sentence, such as a short clause or a noun phrase, the BERT usually took the whole constituent as a single coalition, i.e.
a tender, witty, captivating film about friendship, love, memory, trust and loyalty.

allows us to hope that Nolan is poised to embark a major career as a commercial yet inventive filmmaker, although laced with humor and a few fanciful touches, the film is a refreshingly serious look at young women.

. . . nothing scary here except for some awful acting and lame special effects.

an entertaining, colorful, action-filled crime story with an intimate heart.

far more imaginative and ambitious than the trivial, cash-in features nickelodeon has made from its other animated tv series.

a very well-made, funny and entertaining picture.

the film is powerful, accessible and funny.

a coarse and stupid gross-out.

a smart, witty follow-up.

has a lot of the virtues of eastwood at his best.

dense with characters and contains some thrilling moments.

there is no pleasure in watching a child suffer.

the piquant story needs more dramatic meat on its bones.

as unseemly as its title suggests.

a coarse and stupid gross-out.

and that’s a big part of why we go to the movies.

in a way, the film feels like a breath of fresh air, but only to those that allow it in.

far more imaginative and ambitious than the trivial, cash-in features nickelodeon has made from its other animated tv series.

an entertaining, colorful, action-filled crime story with an intimate heart.

it proves quite compelling as an intense, brooding character study.

. . . nothing scary here except for some awful acting and lame special effects.

although laced with humor and a few fanciful touches, the film is a refreshingly serious look at young women.

allows us to hope that Nolan is poised to embark a major career as a commercial yet inventive filmmaker, a tender, witty, captivating film about friendship, love, memory, trust and loyalty.

Figure 7: More examples of prototype features modeled by the BERT. Red brackets indicated words in A. We used different colors to indicate the extracted coalitions. Words in the same coalition formed a prototype feature modeled by the DNN.

Baseline 2: (Lundberg, Erion, and Lee 2018)

this is a good script, good dialogue, funny even for adults. entertains by providing good, lively company.

Baseline 3: (Li, Monroe, and Jurafsky 2016)

this is a good script, good dialogue, funny even for adults. entertains by providing good, lively company.

Our method:

this is a good script, good dialogue, funny even for adults. entertains by providing good, lively company.

Figure 8: Coalitions generated by two baseline methods and the proposed method.

a prototype feature. (2) For the set of words that contained conjunctions or punctuations, such as “and”, “or”, “,”, “etc.”, the BERT divided the set at conjunctions or punctuations to generate different coalitions (prototype features). (3) For the constituent that was modified by multiple adjectives and adverbs, the BERT usually merged all adjectives and adverbs into one coalition, and took the modified constituent as another coalition. Such phenomena fit the syntactic structure parsing according to human knowledge.

H Comparisons with other baselines on real data

In order to further compare the effectiveness of the proposed method and previous studies, we also tested several other baselines on the BERT learned using the SST-2 dataset. We applied the Baseline 2 (Lundberg et al. 2018) and Baseline 3 (Li et al. 2016), which were introduced in the section “experiments”, to sentences in the SST-2 dataset. Baseline 2 (Lundberg et al. 2018) measured the interaction between two input variables. Baseline 3 (Li et al. 2016) estimated the importance of input variables by masking one input variable in the input. Just like the experiment using toy datasets, Baseline 2 and Baseline 3 merged input variables whose interactions were larger than zero to generate coalitions. Note that both Baseline 2 and Baseline 3 were designed to measure the interaction between two input variables, instead of measuring the interaction among multiple input variables. Thus, coalitions generated by these two baselines could not reflect the prototype features encoded by DNNs. As Figure 8 shows, the coalition generated by the proposed method usually fitted the syntactic structure parsing according to human knowledge better than baselines. In comparison, the Baseline 2 sometimes combined all variables into one single coalition, and the Baseline 3 usually could only generate small coalitions with no more than two words.