TOWARDS UNDERSTANDING AND IMPROVING DROPOUT IN GAME THEORY

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper aims to understand and improve the utility of the dropout operation from the perspective of game-theoretic interactions. We prove that dropout can suppress the strength of interactions between input variables of deep neural networks (DNNs). The theoretic proof is also verified by various experiments. Furthermore, we find that such interactions were strongly related to the over-fitting problem in deep learning. Thus, the utility of dropout can be regarded as decreasing interactions to alleviate the significance of over-fitting. Based on this understanding, we propose an interaction loss to further improve the utility of dropout. Experimental results have shown that the interaction loss can effectively improve the utility of dropout and boost the performance of DNNs. *We will release the code when the paper is accepted.*

1 INTRODUCTION

Deep neural networks (DNNs) have exhibited significant success in various tasks, but the over-fitting problem is still a considerable challenge for deep learning. Dropout is usually considered as an effective operation to alleviate the over-fitting problem of DNNs. Hinton et al. (2012); Srivastava et al. (2014) thought that dropout could encourage each unit in an intermediate-layer feature to model useful information without much dependence on other units. Konda et al. (2016) considered dropout as a specific method of data augmentation. Gal & Ghahramani (2016) proved that dropout was equivalent to the Bayesian approximation in a Gaussian process.

In this paper, we aim to explain, model, and improve the utility of dropout from the following perspectives. First, we prove that the dropout operation suppresses interactions between input units encoded by DNNs. This is also verified by various experiments. To this end, the interaction is defined in game theory, as follows. Let x denote the input, and let $f(x)$ denote the output of the DNN. For the i-th input variable, we can compute its importance value $\phi(i)$, which measures the numerical contribution of the i-th variable to the output $f(x)$. We notice that the importance value of the i-th variable would be different when we mask the j-th variable *w.r.t.* the case when we do not mask the j-th variable. Thus, the interaction between input variables i and j is measured as the difference $\phi_{\text{w/ }j}(i) - \phi_{\text{w/o }j}(i)$.

Second, we also discover a strong correlation between interactions of input variables and the over-fitting problem of the DNN. Specifically, the over-fitted samples usually exhibit much stronger interactions than ordinary samples.

Therefore, we consider that the utility of dropout is to alleviate the significance of over-fitting by decreasing the strength of interactions encoded by the DNN. Based on this understanding, we propose an interaction loss to further improve the utility of dropout. The interaction loss directly penalizes the interaction strength, in order to improve the performance of DNNs. The interaction loss exhibits the following two distinct advantages over the dropout operation. (1) The interaction loss explicitly controls the penalty of the interaction strength, which enables people to trade off between over-fitting and under-fitting. (2) Unlike dropout which is incompatible with the batch normalization operation (Li et al., 2019), the interaction loss can work in harmony with batch normalization. Various experimental results show that the interaction loss can boost the performance of DNNs.

Furthermore, we analyze interactions encoded by DNNs from the following three perspectives. (1) First, we discover the consistency between the sampling process in dropout (when the dropout rate

$p = 0.5$) and the sampling in the computation of the Banzhaf value. It shows that the frequent inference patterns in Banzhaf interactions (Grabisch & Roubens, 1999) are also prone to be frequently sampled by dropout, thereby being stably learned. This ensures the DNN to encode smooth Banzhaf interactions. We also prove that the Banzhaf interaction is close to the aforementioned interaction, which also relates to the dropout operation with the interaction used in this paper. (2) Besides, we find that the interaction loss is better to be applied to low layers than being applied to high layers. (3) Furthermore, we decompose the overall interaction into interaction components of different orders. We visualize the strongly interacted regions within each input sample. We find out that interaction components of high orders take the main part of interactions, and are more prone to be suppressed by the dropout operation and the interaction loss than other interaction components.

Contributions of this paper can be summarized as follows. (1) We mathematically prove and experimentally show that dropout can suppress the strength of interactions encoded by a DNN. (2) We find that the over-fitted samples usually contain stronger interactions than other samples. (3) Based on this, we consider the utility of dropout is to alleviate over-fitting by decreasing the interaction. We design a novel loss function to penalize the strength of interactions, which improves the performance of DNNs. (4) We analyze the properties of interactions encoded by DNNs, and conduct comparative studies to obtain new insights into interactions encoded by DNNs.

2 Related Work

The dropout operation. Dropout is an effective operation to alleviate the over-fitting problem and improve the performance of DNNs (Hinton et al., 2012). Several studies have been proposed to explain the inherent mechanism of dropout. According to (Hinton et al., 2012; Krizhevsky et al., 2012; Srivastava et al., 2014), dropout could prevent complex co-adaptation between units in intermediate layers, and could encourage each unit to encode useful representations itself. However, these studies only qualitatively analyzed the utility of dropout, instead of providing quantitative results. Wager et al. (2013) showed that dropout performed as an adaptive regularization, and established a connection to the algorithm AdaGrad. Konda et al. (2016) interpreted dropout as a kind of data augmentation in the input space, and Gal & Ghahramani (2016) proved that dropout was equivalent to a Bayesian approximation in the Gaussian process. Gao et al. (2019) disentangled the dropout operation into the forward dropout and the backward dropout, and improved the performance by setting different dropout rates for the forward dropout and the backward dropout, respectively. Unlike previous studies, we aim to explain the utility of dropout from the view of game theory. Furthermore, we propose a method to improve the utility of dropout.

Interaction. Previous studies have explored interactions between input variables. Bien et al. (2013) developed an algorithm to learn hierarchical pairwise interactions inside an additive model. Sorokina et al. (2008) detected the statistical interaction using an additive model-based ensemble of regression trees. Murdoch et al. (2018); Singh et al. (2018); Jin et al. (2019) proposed and extended the contextual decomposition to measure the interaction encoded by DNNs in NLP tasks. Tsang et al. (2018) measured the pairwise interaction based on the learned weights of the DNN. Janizek et al. (2020) extended the explanation method of Integrated Gradients (Sundararajan et al., 2017) to quantify the pairwise feature interaction in DNNs. Tsang et al. (2020) proposed a method, namely GLIDER, to detect the feature interaction modeled by a recommender system.

Besides interactions measured from the above views, game theory is also a typical perspective to analyze the interaction. Several studies explored the interaction based on game theory. 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 theoretic foundations (Weber, 1988), this definition of interaction can be regarded to objectively reflect the collaborative/adversarial effects between variables *w.r.t* the prediction score. Furthermore, Grabisch & Roubens (1999) extended this definition to interactions among different numbers of input variables. Grabisch & Roubens (1999) also proposed the interaction based on the Banzhaf value. In comparison, the target interaction used in this paper is based on the Shapley value (Shapley, 1953). Since the Shapley is the unique metric that satisfies *the linearity property, the dummy property, the symmetry property, and the efficiency property* (Ancona et al., 2019), the interaction based on the Shapley value is usually considered as a more standard metric than the interaction based on the Banzhaf value. In this paper, we aim to explain the utility of

dropout using the interaction defined in game theory. We reveal the close relationship between the strength of interactions and the over-fitting of DNNs. We also design an interaction loss to improve the performance of DNNs.

3 GAME-THEORETIC EXPLANATIONS OF DROPOUT

Preliminaries: Shapley values. The Shapley value was initially proposed by Shapley (1953) in game theory. It is considered as a unique unbiased metric that fairly allocates the numerical contribution of each player to the total award. Given a set of players $N = \{1, 2, \cdots, n\}$, $2^N \overset{\text{def}}{=} \{S | S \subseteq N\}$ denotes all possible subsets of N. A game $f : 2^N \to \mathbb{R}$ is a function that maps from a subset to a real number. $f(S)$ is the *score* obtained by the subset $S \subseteq N$. Thus, $f(N) - f(\emptyset)$ denotes the *award* obtained by all players in the game. The Shapley value allocates the overall reward to each player, as its numerical contribution $\phi(i|N)$, as shown in Equation (1). The Shapley value of player i in the game f, $\phi(i|N)$, is computed as follows.

$$\sum\nolimits_{i=1}^{n} \phi(i|N) = f(N) - f(\emptyset)\ ,\ \phi(i|N) = \sum\nolimits_{S \subseteq N \setminus \{i\}} P_{\text{Shapley}}\big(S \big| N \setminus \{i\}\big) \big[f(S \cup \{i\}) - f(S) \big] \quad (1)$$

where $P_{\text{Shapley}}(S|M) = \frac{(|M|-|S|)!|S|!}{(|M|+1)!}$ is the likelihood of S being sampled, $S \subseteq M$. The Shapley value is the unique metric that satisfies *the linearity property, the dummy property, the symmetry property, and the efficiency property* (Ancona et al., 2019). We summarize these properties in Appendix A.

Understanding DNNs via game theory. In game theory, some players may form a coalition to compete with other players, and win an award (Grabisch & Roubens, 1999). Accordingly, a DNN f can be considered as a game, and the output of the DNN corresponds to the score $f(\cdot)$ in Equation (1). For example, if the DNN has a scalar output, we can take this output as the score. If the DNN outputs a vector for multi-category classification, we select the classification score corresponding to the true class as the score $f(\cdot)$.

The set of players N corresponds to the set of input variables. We can analyze the interaction and the element-wise contribution at two different levels. (1) We can consider input variables (players) as the input of the entire DNN, *e.g.* pixels in images and words in sentences. In this case, the game f is considered as the entire DNN. (2) Alternatively, we can also consider input variables as a set of activation units before the dropout operation. In this case, the game f is considered as consequent modules of the DNN.

$S \subseteq N$ in Equation (1) denotes the context of the input variable i, which consists of a subset of input variables. In order to compute the network output $f(S)$, we replace variables in $N \setminus S$ with the baseline value (*e.g.* mask them), and we do not change the variables in S. In particular, when we consider neural activations before dropout as input variables, such activations are usually non-negative after ReLU. Thus, their baseline values are set to 0. Both (Ancona et al., 2019) and Appendix I introduce details about the baseline value. In this way, $f(\emptyset)$ measures the output score when all input variables are masked, and $f(S) - f(\emptyset)$ measures the entire award obtained by all input variables in S.

Interactions encoded by DNNs. In this section, we introduce how to use the interaction defined in game theory to explain DNNs. Two input variables may interact with each other to contribute to the output of a DNN. Let us suppose input variables i and j have an interaction. In other words, the contribution of i and j when they work jointly is different with the case when they work individually. For example, in the sentence *he is a green hand*, the word *green* and the word *hand* have a strong interaction, because the words *green* and *hand* contribute to the person's identity jointly, rather than independently. In this case, we can consider these two input variables to form a certain inference pattern as a singleton player $S_{ij} = \{i, j\}$. Thus, this DNN can be considered to have only $(n-1)$ input variables, $N' = N \setminus \{i, j\} \cup S_{ij}$, *i.e.* S_{ij} is supposed to be always absent or present simultaneously as a constituent. In this way, the interaction $I(i, j)$ between input variables i and j is defined by Grabisch & Roubens (1999), as the contribution increase of S_{ij} when input variables i and j cooperate with each other *w.r.t.* the case when i and j work individually, as follows.

$$I(i, j) \overset{\text{def}}{=} \phi(S_{ij}|N') - \big[\phi(i|N \setminus \{j\}) + \phi(j|N \setminus \{i\})\big] = \sum\nolimits_{S \subseteq N \setminus \{i,j\}} P_{\text{Shapley}}\big(S \big| N \setminus \{i, j\}\big) \Delta f(S, i, j) \quad (2)$$

where $\Delta f(S, i, j) \overset{\text{def}}{=} f(S \cup \{i, j\}) - f(S \cup \{j\}) - f(S \cup \{i\}) + f(S)$. $\phi(i|N \setminus \{j\})$ and $\phi(j|N \setminus \{i\})$ correspond to the contribution to the DNN output when i and j work individually. Theoretically,

Figure 1: Visualization of the interaction strength encoded by DNNs.

$I(i,j)$ is also equal to the change of the variable i's Shapley value when we mask another input variable j *w.r.t.* the case when we do not mask j. Please see Appendix B for the proof.

If $I(i,j) > 0$, input variables i and j cooperate with each other for a higher output value. Whereas, if $I(i,j) < 0$, i and j have a negative/adversarial effect. The strength of the interaction can be computed as the absolute value of the interaction, *i.e.* $|I(i,j)|$. We find that the overall interaction $I(i,j)$ can be decomposed into interaction components with different orders s, as follows.

$$I(i,j) = \sum_{s=0}^{n-2} \Big[\frac{I^{(s)}(i,j)}{n-1} \Big], \qquad I^{(s)}(i,j) \stackrel{\text{def}}{=} \mathbb{E}_{S \subseteq N \setminus \{i,j\}, |S|=s} \Big[\Delta f(S,i,j) \Big] \tag{3}$$

where s denote the size of the context S for the interaction. We use $I^{(s)}(i,j)$ to represent the s-order interaction between input variables i and j. $I^{(s)}(i,j)$ reflects the average interaction between input variables i and j among all contexts S with s input variables. For example, when s is small, $I^{(s)}(i,j)$ measures the interaction relying on inference patterns consisting of very few input variables, *i.e.* the interaction depends on a small context. When s is large, $I^{(s)}(i,j)$ corresponds to the interaction relying on inference patterns consisting of a large number of input variables, *i.e.* the interaction depends on the context of a large scale.

Visualization of interactions in each sample, which are encoded by the DNN: There exists a specific interaction between each pair of pixels, which boosts the difficulty of visualization. In order to simplify the visualization, we divide the original image into 16×16 grids, and we only visualize the strength of interactions between each grid g and its neighboring grids g' as $\text{Color}(g) = \mathbb{E}_{g' \in \text{neighbor}(g)}[|I(g,g')|]$. Figure 1 visualizes the interaction strength within the image in the CelebA dataset (Liu et al., 2015), which are normalized to the range $[0,1]$. Grids on the face usually contain more significant interactions with neighboring grids than grids in the background.

Proof of the relationship between dropout and the interaction. In this section, we aim to mathematically prove that dropout is an effective method to suppress the interaction strength encoded by DNNs. Given the context S, let us consider its subset $T \subseteq S$, which forms a coalition to represent a specific inference pattern $T \cup \{i,j\}$. Note that for dropout, the context refers to activation units in the intermediate-layer feature without semantic meanings. Nevertheless, we just consider i,j as pixels as a toy example to illustrate the basic idea, in order to simplify the introduction. For example, let S represent the face, and let $T \cup \{i,j\}$ represent pixels of an eye in the face. Let $R^T(i,j)$ quantify the marginal award obtained from the inference pattern of an eye. All interaction effects from smaller coalitions $T' \subsetneqq T$ are removed from $R^T(i,j)$.

According to the above example, Let $T' \subsetneqq T$ correspond to the pupil inside the eye. Then, $R^{T'}(i,j)$ measures the marginal award benefited from the existence of the pupil $T' \cup \{i,j\}$, while $R^T(i,j)$ represents the marginal benefit from the existence of the entire eye, in which the award from the pupil has been removed. *I.e.* the inference pattern $T \cup \{i,j\}$ can be exclusively triggered by the co-occurrence of all pixels in the eye, but cannot be triggered by a subset of pixels in the pupil $T' \cup \{i,j\}$. The benefit from the pupil pattern has been removed from $R^T(i,j)$. Thus, the s-order interaction can be decomposed into components *w.r.t.* all inference patterns $T \cup \{i,j\}, T \subseteq S$.

$$I^{(s)}(i,j) = \mathbb{E}_{S \subseteq N \setminus \{i,j\}, |S|=s} \Big[\sum\nolimits_{T \subseteq S} R^T(i,j) \Big] = \sum\nolimits_{0 \le q \le s} \tbinom{s}{q} J^{(q)}(i,j) = \sum\nolimits_{0 \le q \le s} \Gamma^{(q)}(i,j|s) \tag{4}$$

where $J^q(i,j) \stackrel{\text{def}}{=} \mathbb{E}_{T \subseteq N \setminus \{i,j\}, |T|=q}[R^T(i,j)]$ denotes the average interaction between i and j given all potential inference patterns $T \cup \{i,j\}$ with a fixed inference pattern size $|T| = q$; $\Gamma^{(q)}(i,j|s) \stackrel{\text{def}}{=} \binom{s}{q} J^{(q)}(i,j)$. The computation of $R^T(i,j)$ and the proof of Equation (4) are provided in Appendices E and C, respectively.

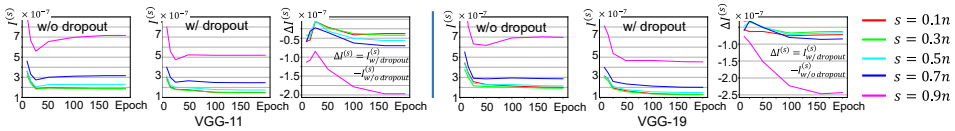

Figure 2: Curves of strength of interaction components with different orders during the learning process. Interaction components with high orders took the main part of the interaction. Dropout mainly penalized interaction components with high orders. All interactions were computed between activation units before dropout. $\Delta I^{(s)} = I^{(s)}_{\text{w/ dropout}} - I^{(s)}_{\text{w/o dropout}}$ denotes the change of interaction strength.

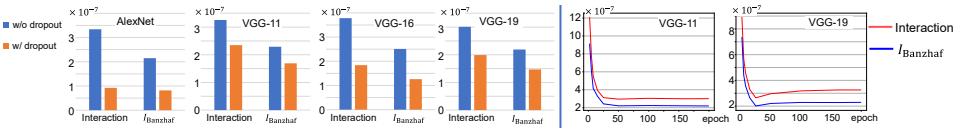

Figure 3: (left) The decrease of interactions caused by the dropout operation. (right) Close relationship between the target interaction $\mathbb{E}_{\text{image}}[\mathbb{E}_{(i,j)\in\text{image}}|I(i,j)|]$ and the interaction based on the Banzhaf value $\mathbb{E}_{\text{image}}[\mathbb{E}_{(i,j)\in\text{image}}|I_{\text{Banzhaf}}(i,j)|]$. We show the change of the two types of interactions during the learning process. The target interaction is highly related to the interaction based on Banzhaf value. Here, all interactions were computed between activation units before dropout.

However, when input variables in N are randomly removed by the dropout operation, the computation of $I^{(s)}_{\text{dropout}}(i,j)$ only involves a subset of inference patterns consisting of variables that are not dropped. Let the dropout rate be $(1-p), p \in [0,1]$, and let $S' \subseteq S$ denote the input variables that remain in the context S after the dropout operation. Let us consider cases when r activation units remain after the dropout operation, *i.e.* $|S'| = r$. Then, the average interaction $I^{(s)}_{\text{dropout}}(i,j)$ in these cases can be computed as follows.

$$I^{(s)}_{\text{dropout}}(i,j) = \mathop{\mathbb{E}}_{S\subseteq N\setminus\{i,j\},|S|=s}\Big[\mathop{\mathbb{E}}_{S'\subseteq S,|S'|=r}\Big(\sum\nolimits_{T\subseteq S'} R^T(i,j)\Big)\Big] = \sum\nolimits_{0\le q\le r}\Gamma^{(q)}(i,j|r) \tag{5}$$

The interaction only comprises the marginal awards from the inference patterns consisting of at most $r \sim \text{B}(s,p)$ variables, where $\text{B}(s,p)$ is the binomial distribution with the sample number s and the sample rate p. Since $r = |S'| \le s$, we have

$$1 \ge \frac{\Gamma^{(1)}(i,j|r)}{\Gamma^{(1)}(i,j|s)} \ge \cdots \ge \frac{\Gamma^{(r)}(i,j|r)}{\Gamma^{(r)}(i,j|s)} \ge 0, \qquad \frac{I^{(s)}_{\text{dropout}}(i,j)}{I^{(s)}(i,j)} = \frac{\sum_{0\le q\le r}\Gamma^{(q)}(i,j|r)}{\sum_{0\le q\le s}\Gamma^{(q)}(i,j|s)} \le 1. \tag{6}$$

Equation (6, right) shows that the number of inference patterns usually significantly decreases when we use dropout to remove $s-r, r\sim\text{B}(s,p)$, activation units. Please see Appendix D for the proof.

Experimental verification 1: Inference patterns with more activation units are less likely to be sampled when we use dropout, thereby being more vulnerable to the dropout operation, according to Equation (6, left). Inspired by this, we conducted experiments to explore the following two terms, (1) which component $\{I^{(s)}(i,j)\}(s=0,...,n-2)$ took the main part of the overall interaction strength among all interaction components with different orders; (2) which interaction component was mainly penalized by dropout.

To this end, the strength of the s-order interaction component was averaged over images, *i.e.* $I^{(s)} = \mathbb{E}_{\text{image}}[\mathbb{E}_{(i,j)\in\text{image}}(|I^{(s)}(i,j)|)]$. Specifically, when we analyzed the strength of interaction components, we selected the following orders, $s = 0.1n, 0.3n, ..., 0.9n$. Figure 2 shows curves of different interaction components[1] within VGG-11/19 (Simonyan & Zisserman, 2015) learned on the CIFAR-10 (Krizhevsky & Hinton, 2009) dataset with and without the dropout operation. We found that the interaction components with high orders took the main part of the interaction. According to Equation (6, left), dropout mainly penalized the interaction component with high orders. This phenomenon was also verified in Figure 2. In this way, we proved that high-order interaction components were usually more suppressed than low-order interaction components. In other words, inference patterns with more activation units (with higher orders) were more vulnerable to the dropout operation.

[1]For fair comparison, we normalize the value of interaction using the range of output scores of DNNs. Please see Appendix G for more details.

Dataset	Model	Ordinary	Over-fitted
MNIST	RN-44	2.17×10^{-3}	$\mathbf{3.64\times10^{-3}}$
Tiny-ImageNet	RN-34	2.57×10^{-3}	$\mathbf{2.89\times10^{-3}}$
CelebA	RN-34	6.46×10^{-3}	$\mathbf{1.17\times10^{-2}}$

Table 1: Comparison of the interaction strength between the over-fitted samples and ordinary samples. DNNs usually encode more interactions for the over-fitted samples than ordinary samples.

DNN	Dataset	$\text{Loss}_{\text{interaction}}$		Dropout	
		low layers	high layers	low layers	high layers
AlexNet	CIFAR-10	**69.6**	67.2	67.5	68.4
VGG-13	CIFAR-10	**66.2**	61.4	60.9	59.7
VGG-16	CIFAR-10	**64.9**	62.6	63.0	59.5
VGG-19	Tiny-ImageNet	**45.2**	37.4	32.6	37.0
VGG-16	CelebA	**94.6**	93.9	92.4	94.0

Table 2: Classification accuracy of DNNs when we applied the dropout operation and the interaction loss at different positions.

Experimental verification 2: We also conducted experiments to illustrate how dropout suppressed the interaction modeled by DNNs, which was a verification of Equation (6, right). In experiments, we trained AlexNet (Krizhevsky et al., 2012), and VGG-11/16/19 on CIFAR-10 with and without the dropout. Figure 3(left) compares the strength of interactions[1] encoded by DNNs, which were learned with or without the dropout operation. We averaged the strength of interactions over images, *i.e.* $I = \mathbb{E}_{\text{image}}[\mathbb{E}_{(i,j)\in\text{image}}(|I(i,j)|)]$, where $I(i,j)$ was obtained according to Equation (2). Note that accurately computing the interaction of two input variables was an NP-hard problem. Thus, we applied a sampling-based method (Castro et al., 2009) to approximate the strength of interactions. Appendix G shows technical details of (Castro et al., 2009). We found that dropout could effectively suppress the strength of the interaction, which verified the above proof.

Property: The sampling process in dropout is the same as the computation in the Banzhaf value. In this section, we aim to show that the sampling process in dropout (when the dropout rate is 0.5) is similar to the sampling in the computation of the Banzhaf value. Just like the Shapley value, the Banzhaf value (Banzhaf III, 1964) is another typical metric to measure importance of each input variable in game theory. Unlike the Shapley value, the Banzhaf value is computed under the assumption that each input variable independently participates in the game with the probability 0.5. The Banzhaf value is computed as $\psi(i|N) = \sum_{S\subseteq N\setminus\{i\}} P_{\text{Banzhaf}}(S|N\setminus\{i\})\big[f(S\cup\{i\}) - f(S)\big]$, where $P_{\text{Banzhaf}}(S|N\setminus\{i\}) = 0.5^{n-1}$ is the likelihood of S being sampled. The form of the Banzhaf value is similar to that of the Shapley value in Equation (1), but the sampling weight of the Banzhaf value $P_{\text{Banzhaf}}(S|N\setminus\{i\})$ is different from $P_{\text{Shapley}}(S|N\setminus\{i\})$ of the Shapley value. For dropout with the dropout rate 0.5, let n be the number of input variables, and let S be the units not dropped in $N\setminus\{i\}$. Then, the likelihood of S not being dropped is given as $P_{\text{dropout}}(S|N\setminus\{i\}) = 0.5^{|S|}0.5^{n-|S|-1} = P_{\text{Banzhaf}}(S|N\setminus\{i\})$. In this way, dropout usually generates activation units S following $P_{\text{Banzhaf}}(S|N\setminus\{i\})$. Therefore, the frequent inference patterns in the computation of the Banzhaf value are also frequently generated by dropout, thereby being reliably learned by the DNN. This makes that $f(S\cup\{i\}) - f(S)$ in the computation of the Banzhaf value can be sophisticatedly modeled without lots of outlier values. Thus, the dropout can be considered as a smooth factor in terms of the Banzhaf value. Please see Appendix F for the analysis of dropout with other dropout rates.

Grabisch & Roubens (1999) defined the interaction based on the Banzhaf value as $I_{\text{Banzhaf}}(i,j) = \sum_{S\subseteq N\setminus\{i,j\}} P_{\text{Banzhaf}}\big(S\big|N\setminus\{i,j\}\big)\Delta f(S,i,j)$, just like Equation (2). Thus, dropout can also be considered as a smooth factor in the computation of the Banzhaf interaction. Figure 3(right) shows that the target interaction used in this study is closely related to the Banzhaf interaction, which potentially connects the target interaction based on the Shapley value to the dropout operation.

4 Utility of Dropout & Improvement of the Dropout Utility

Close relationship between the strength of interactions and over-fitting. In this section, we conducted various experiments to explore the relationship between the interaction strength and the over-fitted samples. We noticed that in the classification task, the over-fitted samples were usually outliers. To this end, we assigned 5% training samples with random incorrect labels in MNIST (Lecun et al., 1998), CelebA(Liu et al., 2015), and Tiny ImageNet (Le & Yang, 2015). When the DNN was well-trained on such training data with an almost zero training loss, then we took samples with incorrect labels as the over-fitted samples. We trained ResNet-34 (RN-34) (He et al., 2016) for the classification task using the Tiny ImageNet dataset and the CelebA dataset, and trained ResNet-44 (RN-44) for the classification task using the MNIST dataset. For each trained DNN, we divided input images into grids, and computed the average interaction strength between grids on the over-fitted samples and ordinary samples, respectively. As Table 1 shows, the over-fitted samples usually contained stronger interactions[1] than ordinary samples.

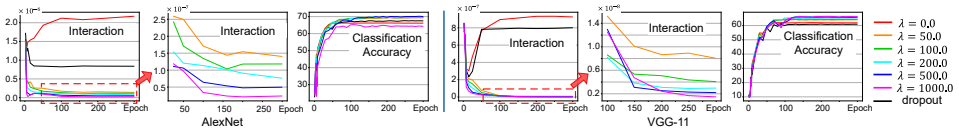

Figure 4: Curves of interaction strength encoded by DNNs trained using different weights of the interaction loss. The interaction strength decreases along with the increase of the weight λ. For fair comparison, we computed the interaction between activation units before the layer, in which we used the interaction loss or the dropout operation.

Understanding of dropout. We have proved that dropout can decrease the strength of interactions encoded in DNNs in Section 3. Besides, previous paragraphs have shown that the over-fitted samples usually encode more interactions by the DNN than other samples. Therefore, we consider the utility of the dropout operation is to decrease interaction strength to reduce the significance of over-fitting.

Further improvement of the utility of dropout using the interaction loss. Based on the above understanding, we develop the interaction loss as an improvement of the utility of dropout. To this end, we apply this loss to learn DNNs, in order to boost the performance, as follows.

$$\text{Loss} = \text{Loss}_{\text{classification}} + \lambda \text{Loss}_{\text{interaction}}, \tag{7}$$

where $\lambda > 0$ is the weight of the interaction loss. Let us focus on an intermediate-layer feature after the ReLU operation $\mathbf{h} \in \mathbb{R}^n$ as n activation units. We aim to suppress interactions between any two units $i, j \in N$. Thus, the interaction loss can be formulated as $\text{Loss}_{\text{interaction}} = \mathbb{E}_{i,j\in N, i\neq j}[|I(i,j)|]$. The baseline value in the computation of $I(i,j)$ is set to 0, as is explained in the paragraph *understanding DNNs via game theory*, Section 3.

$$\text{Loss}_{\text{interaction}} = \mathbb{E}_{i,j\in N,i\neq j}\left[|I(i,j)|\right] = \mathbb{E}_{i,j\in N,i\neq j}\left[\left|\sum\nolimits_{S\subseteq N\setminus\{i,j\}} P_{\text{Shapley}}(S|N\setminus\{i,j\})\left[\Delta f(S,i,j)\right]\right|\right] \tag{8}$$

However, it is extremely computational expensive to use the above interaction loss to train the DNN. Therefore, we propose the following approximation of the interaction loss, which is implemented in a batch manner, instead of averaging all pairs of activation units. Specifically, we sample disjoint subsets of units $A, B \subsetneqq N, A\cap B = \emptyset$, instead of sampling two single units i, j, to compute for the interaction loss. Here, we can regard A and B as batches of the sampled units $\{i\}$, $\{j\}$, respectively. Accordingly, the context S is disjoint with A and B, *i.e.* $S \subseteq N\setminus A\setminus B$. We can get the following approximation of $\text{Loss}_{\text{interaction}}$. Please see Appendix H for the proof.

$$\text{Loss}'_{\text{interaction}} = \mathbb{E}_{A,B\subsetneqq N, A\cap B=\emptyset, |A|=|B|=\alpha n}\left\{\mathbb{E}_r\left[\mathbb{E}_{|S|=r, S\subseteq N\setminus A\setminus B}\left[\Delta f(S,A,B)^2\right]\right]\right\} \tag{9}$$

where $\Delta f(S,A,B) \overset{\text{def}}{=} f(S\cup A\cup B) - f(S\cup A) - f(S\cup B) + f(S)$, and α is a small positive constant. In this paper, we set α=0.05. Figure 4(left) verifies the proposed approximated interaction loss can successfully reduce the interaction defined in Equation (2).

Advantages. Compared with dropout, the interaction loss exhibits following advantages.

Advantage 1: The interaction loss enables people to explicitly control the penalty of the interaction strength by adjusting the weight λ in Equation (7). The explicit control of the interaction strength is crucial, because it is important to make a careful balance between over-fitting and under-fitting during the learning of the DNN. The weight λ needs to be carefully set, since it is usually difficult to trade off between over-fitting and under-fitting in deep learning. A large value of λ may lead to under-fitting, while a small value of λ may increase the risk of over-fitting. Unlike the interaction loss, people cannot explicitly control the strength of dropout during the learning process.

Advantage 2: Due to the disharmony between dropout and the batch normalization (Li et al., 2019), people usually cannot apply dropout in the DNN with batch normalization. In comparison, the interaction loss is compatible with batch normalization. For example, we compute $\text{Loss}_{\text{interaction}}$ using a new track, which is different from the track of computing $\text{Loss}_{\text{classification}}$. In other words, we do not update the parameter in batch normalization layers when we compute $\text{Loss}_{\text{interaction}}$. Please see Appendix I for more discussions. Furthermore, Appendix J.2 verifies the disharmony in (Li et al., 2019), in which we added batch normalization layers to VGG-16/19, and compared the classification accuracy with and without dropout.

Experiments: We trained DNNs for the classification based on the CIFAR-10 dataset (Krizhevsky & Hinton, 2009), the classification based on the Tiny ImageNet dataset (Le & Yang, 2015), and

CIFAR-10 dataset

λ	AlexNet[2]	VGG-11[2]	VGG-13[2]	VGG-16[2]
0.0	66.2	61.9	60.8	62.0
50.0	69.2	63.9	64.0	63.8
100.0	69.6	64.3	65.4	64.5
200.0	69.6	65.3	65.9	64.7
500.0	**70.0**	65.9	**66.2**	**64.9**
1000.0	64.3	**66.3**	66.0	64.5
Dropout	67.5	60.9	60.9	63.0

Tiny ImageNet

λ	RN-18[2]	RN-34[2]	λ	VGG-16	VGG-19
0.0	48.8	45.6	0.0	33.4	37.6
0.001	50.0	48.4	50.0	38.4	38.2
0.003	49.6	49.0	100.0	38.0	38.6
0.01	**52.2**	**49.6**	200.0	38.2	39.0
0.03	50.4	48.8	500.0	**42.8**	41.8
			1000.0	40.8	**45.2**
Dropout	47.4	46.0	Dropout	36.8	32.6

Gender estimation

λ	VGG-13	VGG-16	λ	RN-18
0.0	94.6	93.7	0.0	92.7
5.0	94.8	93.8	0.001	93.0
10.0	94.7	**94.6**	0.003	**93.1**
20.0	**94.9**	94.1	0.01	93.0
50.0	94.7	94.08	0.03	92.9
100.0	94.7	94.3		
Dropout	94.6	92.4	Dropout	92.1

Over-fitting ↑ ↓ Under-fitting

Table 3: Classification accuracy when the DNNs are controlled from over-fitting to under-fitting.

the face attribution estimation based on the CelebA dataset (Liu et al., 2015). We trained six types of DNNs, including AlexNet[2] (Krizhevsky et al., 2012), VGG-11/13/16 (Simonyan & Zisserman, 2015), and ResNet-18/34 (RN-18/34) (He et al., 2016). Note that we only randomly sampled 10% training data in the CIFAR-10 dataset and the Tiny ImageNet dataset, and randomly sampled training data form the CelebA dataset. This was because a DNN usually showed a low significance of over-fitting, when it was learned on the complete dataset. We only sampled a small proportion of training images to better show the over-fitting problem.

• *Learning DNNs with the interaction loss.* We trained the DNNs with different weights of the interaction loss, and evaluated their testing accuracy. Table 3 shows that the accuracy of the DNN first increased and then decreased when we gradually increased the value of λ. This could be explained as the interaction loss first alleviated the significance of over-fitting, and then caused under-fitting. Besides, Figure 4 shows curves of the interaction during the learning process with different weights λ of interaction loss. We found that the interaction decreased along with the increase of the weight λ, which verified the effectiveness of the interaction loss. Besides, we found that the interaction loss mainly penalized interaction components with high orders, just like the dropout operation. Please see Appendix J.1 for more results. In particular, Table 3 shows that the performance of ResNets sometimes dropped when these DNNs were learned with dropout. This phenomenon also verified the conclusion in (Li et al., 2019), *i.e.* dropout was not compatible with batch normalization. In comparison, learning with the interaction loss did not suffer from the use of batch normalization.

• *Exploring the suitable position for the interaction loss and the dropout operation.* We learned the AlexNet and VGG-13/16 using the CIFAR-10 dataset, learned the VGG-19 using the Tiny ImageNet dataset, and learned the VGG-16 using the CelebA dataset. For each DNN, we put the dropout operation and the interaction loss in the low convolutional layer (before the 3rd/5th convolutional layer of the AlexNet/VGGs) and the high fully-connected layer (before the 2nd fully-connected layer), respectively, and compared their performance. As Table 2 shows, the interaction loss was better to be applied to the feature of a low convolutional layer. In comparison, we did not find out a clear principle about the suitable position for dropout.

Discussions about advantages of the interaction loss. As is mentioned above, the interaction loss exhibits two advantages over dropout. Table 3 shows that the interaction loss enabled people to explicitly control the interaction strength. When the λ value increased, DNNs were controlled from over-fitting to under-fitting. Both the over-fitting problem and the under-fitting problem would hurt the performance. However, this explicit control could not be achieved by dropout. Furthermore, unlike dropout, the interaction loss was compatible with batch normalization. As Table 3 shows, the use of dropout decreased the classification accuracy of ResNet-18 (RN-18) on the Tiny ImageNet dataset by 1.4%, and decreased the classification accuracy of ResNet-18 (RN-18) on the CelebA dataset by 0.6%. In comparison, the use of the interaction loss does not suffer from the batch normalization operation. Appendix J.2 reports extended results beyond Table 3. For fair comparison, the interaction loss and dropout were put in the same position of the DNN.

5 Conclusion

In this paper, we explain, model, and improve the utility of the dropout operation using game theory. We prove that dropout can reduce the strength of interactions, thereby improving the performance of DNNs. Experimental results have verified this conclusion. Furthermore, based on the close relationship between the interaction and over-fitting, we propose an interaction loss to directly penalize the strength of interactions, in order to improve the utility of dropout. We find that the interaction loss can reduce the interaction strength effectively and further boost the performance of DNNs.

[2]We slightly adjusted the structure of DNNs to adapt them to input images with sizes 32×32 and 64×64, according to the classic and standard implementation of (pytorch-cifar100, 2020).

REFERENCES

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

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

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

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

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: representing model uncertainty in deep learning. In *ICML*, pp. 1050–1059, 2016.

Hongchang Gao, Jian Pei, and Heng Huang. Demystifying dropout. In *ICML*, 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.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *In CVPR*, 2016.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv: Neural and Evolutionary Computing*, 2012.

Joseph D Janizek, Pascal Sturmfels, and Su-In 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.

Kishore Reddy Konda, Xavier Bouthillier, Roland Memisevic, and Pascal Vincent. Dropout as data augmentation. In *ICLR*, 2016.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *In Computer Science Department, University of Toronto, Tech. Rep*, 1, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *In NIPS*, 2012.

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7, 2015.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Xiang Li, Shuo Chen, Xiaolin Hu, and Jian Yang. Understanding the disharmony between dropout and batch normalization by variance shift. In *CVPR*, pp. 2682–2690, 2019.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *In ICCV*, 2015.

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

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

pytorch-cifar100. Proper implementation of DNNs on CIFAR-10 using PyTorch. `https://github.com/weiaicunzai/pytorch-cifar100`, 2020.

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

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *In ICLR*, 2015.

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

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

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

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

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

Michael Tsang, Dehua Cheng, Hanpeng Liu, Xue Feng, Eric Zhou, and Yan Liu. Feature interaction interpretability: A case for explaining ad-recommendation systems via neural interaction detection. In *ICLR*, 2020.

Stefan Wager, Sida I Wang, and Percy Liang. Dropout training as adaptive regularization. In *NIPS*, pp. 351–359, 2013.

Robert J Weber. Probabilistic values for games. *The Shapley Value. Essays in Honor of Lloyd S. Shapley*, pp. 101–119, 1988.

A FOUR DESIRABLE PROPERTIES OF THE SHAPLEY VALUE

In this section, we mainly introduce the four desirable properties of the Shapley value mentioned in Section 3, including *the linearity property, the dummy property, the symmetry property and the efficiency property* (Ancona et al., 2019).

Linearity property: Given three games f, g, and h. If the award of the game f satisfies $f(S) = g(S) + h(S)$, then the Shapley value of each player $i \in N$ in the game f is the sum of Shapley values of the player i in the game g and h, *i.e.* $\phi_f(i|N) = \phi_g(i|N) + \phi_h(i|N)$.

Dummy property: In a game f, If a player i satisfies $\forall S \subseteq N \setminus \{i\}, f(S \cup \{i\}) = f(S) + f(\{i\})$, then this player is defined as the dummy player. The dummy player i satisfies $\phi(i|N) = f(\{i\}) - f(\emptyset)$, *i.e.* the dummy player has no interaction with other players in N.

Symmetry property: Given two player i and j in a game f, if $\forall S \subseteq N \setminus \{i, j\}, f(S \cup \{i\}) = f(S \cup \{j\})$, then $\phi(i|N) = \phi(j|N)$.

Efficiency property: For a game f, the overall award can be distributed to each player in the game. *I.e.* $\sum_{i \in N} \phi(i|N) = f(N) - f(\emptyset)$, where $f(\emptyset)$ is the obtained award when no player participates in the game.

B THE CONSISTENCY OF TWO UNDERSTANDINGS TOWARDS THE INTERACTION

This section aims to give a proof for the consistency of the following two understandings towards the interaction, as is mentioned in Section 3.

1. $I(i, j)$ is defined as the contribution increase of S_{ij} when input variables i and j cooperate with each other *w.r.t.* the case when each input variable works individually (Grabisch & Roubens, 1999).

2. $I(i, j)$ can also be understood as the difference between the Shapley value of input variable i when we mask another input variable j *w.r.t.* the case when we do not mask j.

• According to the first understanding, the interaction of input variables i and j can be computed as follows.

$$I(i,j) \triangleq \phi(S_{ij}|N') - \Big[\phi(i|N\setminus\{j\}) + \phi(j|N\setminus\{i\})\Big] = \sum_{S \subseteq N\setminus\{i,j\}} \frac{(n-|S|-2)!|S|!}{(n-1)!} \Big[\Delta f(S,i,j)\Big] \quad (10)$$

where $\Delta f(S, i, j) = f(S \cup \{i, j\}) - f(S \cup \{i\}) - f(S \cup \{j\}) + f(S)$.

• According to the second understanding, the interaction of input variables i and j can be computed as follows.

$$\begin{aligned} I(i,j) &\triangleq \phi_{\text{w/ } j}(i) - \phi_{\text{w/o } j}(i) \\ &= \underbrace{\sum_{S \in N\setminus\{i,j\}} \frac{(n-|S|-2)!|S|!}{(n-1)!} \Big[f(S\cup\{i,j\}) - f(S\cup\{j\})\Big]}_{\text{the Shapley value of input variable } i \text{ when we do not mask } j} \\ &\quad - \underbrace{\sum_{S \in N\setminus\{i,j\}} \frac{(n-|S|-2)!|S|!}{(n-1)!} \Big[f(S\cup\{i\}) - f(S)\Big]}_{\text{the Shapley value of input variable } i \text{ when we mask } j} \\ &= \sum_{S \subseteq N\setminus\{i,j\}} \frac{(n-|S|-2)!|S|!}{(n-1)!} \Big[\Delta f(S,i,j)\Big] \end{aligned} \quad (11)$$

where $\Delta f(S, i, j) = f(S \cup \{i, j\}) - f(S \cup \{i\}) - f(S \cup \{j\}) + f(S)$.

Equation (10) and (11) prove the consistency of the two understandings towards the interaction mentioned in Section 3.

C PROOF OF EQUATION (4)

This section provides the detailed proof for Equation (4) in Section 3. The s-order interaction can be decomposed as follows.

$$
\begin{aligned}
I^{(s)}(i,j) &= \mathop{\mathbb{E}}_{\substack{S\subseteq N\setminus\{i,j\}\\|S|=s}}\left[\sum_{T\subseteq S} R^T(i,j)\right] && (12)\\
&= \frac{1}{\binom{n-2}{s}}\sum_{\substack{S\subseteq N\setminus\{i,j\}\\|S|=s}}\left(\sum_{T\subseteq S} R^T(i,j)\right) && (13)\\
&= \frac{1}{\binom{n-2}{s}}\sum_{\substack{S\subseteq N\setminus\{i,j\}\\|S|=s}}\left[\sum_{0\le q\le s}\sum_{\substack{T\subseteq S\\|T|=q}} R^T(i,j)\right] && (14)\\
&= \frac{1}{\binom{n-2}{s}}\sum_{0\le q\le s}\left[\sum_{\substack{S\subseteq N\setminus\{i,j\}\\|S|=s}}\sum_{\substack{T\subseteq S\\|T|=q}} R^T(i,j)\right] && (15)\\
&= \frac{1}{\binom{n-2}{s}}\sum_{0\le q\le s}\left[\sum_{\substack{T\subseteq N\setminus\{i,j\}\\|T|=q}}\sum_{\substack{T'\subseteq N\setminus\{i,j\}\setminus T\\|T'|=s-q}} R^T(i,j)\right] \quad \text{\% let } S=T\cup T' \text{ and } T\cap T'=\emptyset. && (16)\\
&= \frac{1}{\binom{n-2}{s}}\sum_{0\le q\le s}\left\{\sum_{\substack{T\subseteq N\setminus\{i,j\}\\|T|=q}}\left[\binom{n-2-q}{s-q}R^T(i,j)\right]\right\} && (17)\\
&= \frac{1}{\binom{n-2}{s}}\sum_{0\le q\le s}\left[\binom{n-q-2}{s-q}\binom{n-2}{q}\underbrace{\mathop{\mathbb{E}}_{\substack{T\subseteq N\setminus\{i,j\}\\|T|=q}} R^T(i,j)}_{J^q(i,j)}\right] && (18)\\
&= \sum_{0\le q\le s}\binom{s}{q}J^q(i,j) = \sum_{0\le q\le s}\Gamma^{(q)}(i,j|s) && (19)
\end{aligned}
$$

In Equation (19), $J^q(i,j)=\mathbb{E}_{T\subseteq N\setminus\{i,j\},|T|=q}[R^T(i,j)]$ denotes the average interaction between i and j given all potential inference patterns $T\cup\{i,j\}$ with a fixed inference pattern size $|T|=q$.

D PROOF OF EQUATION (5) AND EQUATION (6, LEFT)

This section gives the detailed proof for Equation (5) and Equation (6, left) in Section 3. We first prove Equation (5). The interaction of i and j with dropout, $I^{(s)}_{\text{dropout}}(i,j)$, can be computed as

$$
\begin{aligned}
I^{(s)}_{\text{dropout}}(i,j) &= \mathop{\mathbb{E}}_{\substack{S\subseteq N\setminus\{i,j\}\\|S|=s}}\left[\mathop{\mathbb{E}}_{\substack{S'\subseteq S\\|S'|=r}}\left(\sum_{T\subseteq S'} R^T(i,j)\right)\right] && (20)\\
&= \mathop{\mathbb{E}}_{\substack{S\subseteq N\setminus\{i,j\}\\|S|=s}}\left[\mathop{\mathbb{E}}_{\substack{S'\subseteq S\\|S'|=r}}\left(\sum_{T\subseteq S'} R^T(i,j)\right)\right] && (21)\\
&= \frac{1}{\binom{n-2}{s}}\sum_{\substack{S\subseteq N\setminus\{i,j\}\\|S|=s}}\frac{1}{\binom{s}{r}}\sum_{\substack{S'\subseteq S\\|S'|=r}}\left(\sum_{T\subseteq S'} R^T(i,j)\right) && (22)\\
&= \frac{1}{\binom{n-2}{s}}\frac{1}{\binom{s}{r}}\sum_{\substack{S'\subseteq N\setminus\{i,j\}\\|S'|=r}}\sum_{\substack{S''\subseteq N\setminus\{i,j\}\setminus S'\\|S''|=s-r}}\left(\sum_{T\subseteq S'} R^T(i,j)\right) \quad \text{\% similar to Equation (15)} \sim \text{Equation (18).} && (23)
\end{aligned}
$$

$$= \frac{1}{\binom{n-2}{s}} \frac{1}{\binom{s}{r}} \binom{n-2-r}{s-r} \sum_{\substack{S' \subseteq N \setminus \{i,j\} \\ |S'|=r}} \left(\sum_{T \subseteq S'} R^T(i,j) \right) \tag{24}$$

$$= \frac{1}{\binom{n-2}{r}} \sum_{\substack{S' \subseteq N \setminus \{i,j\} \\ |S'|=r}} \left(\sum_{T \subseteq S'} R^T(i,j) \right) \tag{25}$$

$$= \mathop{\mathbb{E}}_{\substack{S' \subseteq N \setminus \{i,j\} \\ |S'|=r}} \left(\sum_{T \subseteq S'} R^T(i,j) \right) \tag{26}$$

$$= \sum_{0 \le q \le r} \binom{r}{q} J^q(i,j) \qquad \text{\% similar to the proof of Equation (16).} \tag{27}$$

Thus, Equation (5) holds. In order to prove Equation (6, left), we aim to prove $\forall r \le s$ and $1 \le q \le r-1$, we have $1 \ge \frac{\Gamma^{(q)}(i,j|r)}{\Gamma^{(q)}(i,j|s)} \ge \frac{\Gamma^{(q+1)}(i,j|r)}{\Gamma^{(q+1)}(i,j|s)} \ge 0$. Next, we prove the three inequalities step by step.

First, $\frac{\Gamma^{(q)}(i,j|r)}{\Gamma^{(q)}(i,j|s)} = \frac{\binom{r}{q} J^{(q)}(i,j)}{\binom{s}{q} J^{(q)}(i,j)} = \frac{r!(s-q)!q!}{(r-q)!q!s!} = \frac{r!(s-q)!}{(r-q)!s!} = \frac{r \cdot (r-1) \cdot \cdots \cdot (r-q+1)}{s \cdot (s-1) \cdot \cdots \cdot (s-q+1)} \le 1$.

Next, $\frac{\Gamma^{(q)}(i,j|r)}{\Gamma^{(q)}(i,j|s)} \Big/ \frac{\Gamma^{(q+1)}(i,j|r)}{\Gamma^{(q+1)}(i,j|s)} = \frac{r \cdot (r-1) \cdot \cdots \cdot (r-q+1)}{s \cdot (s-1) \cdot \cdots \cdot (s-q+1)} \Big/ \frac{r \cdot (r-1) \cdot \cdots \cdot (r-(q+1)+1)}{s \cdot (s-1) \cdot \cdots \cdot (s-(q+1)+1)} = \frac{s-q}{r-q} \ge 1$.

Then, $\frac{\Gamma^{(q+1)}(i,j|r)}{\Gamma^{(q+1)}(i,j|s)} = \frac{r-q}{s-q} \ge 0$.

Thus, we prove Equation (6, left).

E THE COMPUTATION OF $R^T(i,j)$

This section gives a recursive formulation of $R^T(i,j)$ mentioned in Section 3. $R^T(i,j)$ measures the marginal award of the inference pattern $T \cup \{i,j\}$, where all interaction effects from smaller inference patterns $T' \subsetneqq T$ are removed from $R^T(i,j)$. In this way, $R^T(i,j)$ can be computed as follows.

$$R^T(i,j) \stackrel{\text{def}}{=} \Delta f(T,i,j) - \sum_{T' \subsetneqq T} R^{T'}(i,j) \tag{28}$$

Next, we prove the correctness of the formulation of $R^T(i,j)$. In other words, we aim to prove $I^{(s)}(i,j) = \mathbb{E}_{S \subseteq N \setminus \{i,j\}, |S|=s} \left[\sum_{T \subseteq S} R^T(i,j) \right]$ in Equation (4).

$$\text{RHS} = \mathbb{E}_{S \subseteq N \setminus \{i,j\}, |S|=s} \Bigg[\underbrace{R^S(i,j)}_{\substack{\text{the marginal reward obtained from} \\ \text{the inference pattern } S \cup \{i,j\}}} + \underbrace{\sum_{T \subsetneqq S} R^T(i,j)}_{\substack{\text{the marginal reward obtained from the} \\ \text{inference patterns smaller than } S \cup \{i,j\}}} \Bigg] \tag{29}$$

$$= \mathbb{E}_{S \subseteq N \setminus \{i,j\}, |S|=s} \left[\Delta f(S,i,j) \right] \qquad \text{\% according to Equation (28).} \tag{30}$$

$$= I^{(s)}(i,j) = \text{LHS} \qquad \text{\% according to Equation (3)} \tag{31}$$

Thus, we prove the correctness of the recursive formulation of $R^T(i,j)$.

F RELATIONSHIP BETWEEN BANZHAF VALUES AND DROPOUT

In this paper, we mainly focus on the dropout operation with the dropout rate 0.5. However, in practice, the dropout rate can be different from 0.5. In order to reveal a general relationship between

dropout and the Banzhaf value, we extend the definition of the Banzhaf value to the *Bernoulli-p Banzhaf value*. The Bernoulli-p Banzhaf value is computed under the assumption that each player independently participates in the game with a probability p following a ditribution of Bernoulli(p). Therefore, the likelihood of the subset S being sampled is computed as follows.

$$P_{p\text{-Banzhaf}}(S|p) = p^{|S|}(1-p)^{n-|S|-1} \tag{32}$$

Thus, the Bernoulli-p Banzhaf value can be computed as follows.

$$\psi^p(i|N) = \sum\nolimits_{S\subseteq N\setminus\{i\}} P_{p\text{-Banzhaf}}(S|p)\big[f(S\cup\{i\}) - f(S)\big] \tag{33}$$

The sampling process of dropout is exactly the same as the sampling in the computation of the Bernoulli-p Banzhaf value. Let n be the number of units and S be units that are not dropped in $N\setminus\{i\}$. If the dropout rate is $(1-p)$, then the probability distribution of S can be expressed as follows.

$$P_{\text{dropout}}(S|p) = p^{|S|}(1-p)^{n-|S|-1} = P_{p\text{-Banzhaf}}(S|p) \tag{34}$$

In this way, we reveal the relationship between dropout and the Bernoulli-p Banzhaf value, *i.e.* dropout usually generates a set of activation units S following the distribution of $P_{p\text{-Banzhaf}}(S|p)$.

G THE COMPUTATION OF THE STRENGTH OF INTERACTIONS

In this section, we introduce implementation details to compute the strength of interactions between input variables modeled by the DNN mentioned in Section 3. Let i and j denote two arbitrary input variables, and N represents the set of all input variables. We use f to represent the DNN. Note that the computation based on Equation (2) is NP-hard. Thus, we apply a sampling-based method (Castro et al., 2009) to approximate the interaction between i and j, as follows.

$$\begin{aligned}
I'(i,j) &\overset{\text{def}}{=} \mathbb{E}_s\Big[\mathbb{E}_{S\subseteq N\setminus\{i,j\},|S|=s}\big[\Delta f(S,i,j)\big]\Big] \\
&= \sum_{S\subseteq N\setminus\{i,j\}} [P(S|N\setminus\{i,j\})\Delta f(S,i,j)] \qquad \text{\% Here } P(S|N\setminus\{i,j\}) = \frac{1}{n-1}\frac{1}{\binom{n-2}{s}}. \\
&= \sum_{S\subseteq N\setminus\{i,j\}} [\frac{1}{n-1}\frac{1}{\binom{n-2}{s}}\Delta f(S,i,j)] \\
&= \sum_{S\subseteq N\setminus\{i,j\}} [\frac{s!(n-2-s)!}{(n-1)!}\Delta f(S,i,j)] \\
&= \sum_{S\subseteq N\setminus\{i,j\}} [P_{\text{Shapley}}(S|N\setminus\{i,j\})\Delta f(S,i,j)] \\
&= I(i,j)
\end{aligned} \tag{35}$$

where $\Delta f(S,i,j) = f(S\cup\{i,j\}) - f(S\cup\{i\}) - f(S\cup\{j\}) + f(S)$, $P(S|N\setminus\{i,j\})$ is the likelihood of the subset S being sampled. According to above equation, we first uniformly sample an integer s from the range $[0, n-2]$, then sample the subset S, $|S| = s$ from $N\setminus\{i,j\}$. Thus, we have $P(S|N\setminus\{i,j\}) = \frac{1}{n-1}\frac{1}{\binom{n-2}{s}}$. $I'(i,j)$ can be approximated by the sampling-based method, as follows.

$$I'(i,j) \approx \mathbb{E}_{s\sim\text{Uniformly sampling}[0,n-2]}\big[\mathbb{E}_{S\sim\text{Sampling } s \text{ units from } N\setminus\{i,j\}}[\Delta f(S,i,j)]\big] \tag{36}$$

According to Section 3, the strength of the interaction $I'(i,j)$ is measured by the absolute value $|I'(i,j)|$. In this way, the overall interaction encoded by the DNN can be approximated as follows.

$$I \approx \mathbb{E}_{\text{image}}\Big[\mathbb{E}_{(i,j)\in\text{image}}\big[|I'(i,j)|\big]\Big] \tag{37}$$

In implementation, we randomly sampled 10 images for the computation. We divided each image into 16×16 grids in experiments to further boost the computation efficiency. We consider each grid as an input variable, and computed the interaction between two grids. We randomly sampled 80 pairs of grids for computation of the interaction. For each pair of grids, we sampled s in Equation (36) for 500 times. For each value of s, we sampled one subset S of all input variables to approximate the

interaction. We averaged the strength of interaction over all sampled pairs of grids as the result of the interaction. Besides, for fair comparison of interactions among different DNNs, we normalized the value of interaction using the range of output score of DNNs. *I.e.* $I_{\text{norm}} = I/Y$, where Y was the normalization term that reflected the range of the output score of the DNN. If the DNN was learned for multi-category classification, then $Y = |\mathbb{E}_{x\in X}[f(x)_{y^*} - \mathbb{E}_{y\neq y^*}[f(x)_y]]|$, where x was sampled from the training set X, y^* was the ground-truth label of x, and $f(x)_y$ denoted the value of the y-th dimension in the output score $f(x)$. If the DNN was learned for binary classification and the output score was a scalar, then $Y = |\mathbb{E}_{x\in X_{\text{positive}}}[f(x)] - \mathbb{E}_{x\in X_{\text{negative}}}[f(x)]|$, where X_{positive} and X_{negative} represent the set of positive samples and the set of negative samples, respectively.

H Proof of the Approximation of the Interaction Loss

This section gives the proof for the trustworthiness of the approximated interaction loss $\text{Loss}'_{\text{interaction}}$ in Equation (9). The relationship between Equation (8) and Equation (9) is proved as follows.

According to Equation (3), Equation (8), and Equation (35), the interaction loss can be approximated by sampling-based method, *i.e.*

$$\text{Loss}_{\text{interaction}} = \mathbb{E}_{i,j\in N}[|I(i,j)|] = \mathbb{E}_{i,j\in N}[|I'(i,j)|] = \mathbb{E}_{i,j\in N}\left[\left|\mathbb{E}_s\left[\mathbb{E}_{|S|=s,S\subseteq N}[\Delta f(S,i,j)]\right]\right|\right] \tag{38}$$

Let μ denote the strength of the interaction $I(i,j)$, *i.e.* $|I(i,j)| = \mu \geq 0$. We assume that $\Delta f(S,i,j)$ follows the Gaussian distribution, *i.e.* $\Delta f(S,i,j) \sim \text{Gaussian}(\hat{\mu}, \sigma^2)$. $\hat{\mu} = I(i,j)$. In this way, given a specific pair of units i and j, we have

$$\begin{aligned} P(\Delta f(S,i,j),\mu) &= C\cdot\exp(-\frac{(\Delta f(S,i,j)-\mu)^2}{2\sigma^2})p(\mu) \\ P(\Delta f(S,i,j),-\mu) &= C\cdot\exp(-\frac{(\Delta f(S,i,j)+\mu)^2}{2\sigma^2})p(-\mu), \end{aligned} \tag{39}$$

where $C = \frac{1}{\sqrt{2\pi}\sigma}$ is a constant value. Thus, given the value of $\Delta f(S,i,j)$, the likelihood of $I(i,j) = +\mu$ is given as

$$\begin{aligned} P(\mu|\Delta f(S,i,j)) &= \frac{P(\Delta f(S,i,j),\mu)}{P(\Delta f(S,i,j),\mu) + P(\Delta f(S,i,j),-\mu)} \\ &= \frac{C\cdot\exp(-\frac{(\Delta f(S,i,j)-\mu)^2}{2\sigma^2})p(\mu)}{C\cdot\exp(-\frac{(\Delta f(S,i,j)-\mu)^2}{2\sigma^2})p(\mu) + C\cdot\exp(-\frac{(\Delta f(S,i,j)+\mu)^2}{2\sigma^2})p(-\mu)} \quad \%\ p(\mu) = p(-\mu) \\ &= \frac{\exp[-\frac{1}{2\sigma^2}(\Delta f(S,i,j)-\mu)^2]}{\exp[-\frac{1}{2\sigma^2}(\Delta f(S,i,j)-\mu)^2] + \exp[-\frac{1}{2\sigma^2}(\Delta f(S,i,j)+\mu)^2]} \\ &= \frac{1}{1+\exp(-\frac{2\Delta f(S,i,j)\mu}{\sigma^2})} \\ &= \text{sigmoid}\left(\frac{2\Delta f(S,i,j)\mu}{\sigma^2}\right) \end{aligned} \tag{40}$$

During the learning process, if $P(\mu|\Delta f(S,i,j)) \approx 1$, then it is highly possible that $I(i,j) = \mu > 0$. In this case, we need to decrease $\Delta f(S,i,j)$ to decrease $|I(i,j)|$. In comparison, if $P(\mu|\Delta f(S,i,j)) \approx 0$, then it is possible that $I(i,j) = -\mu < 0$, and we need to increase $\Delta f(S,i,j)$ to reduce $|I(i,j)|$. Thus, we set the coefficient of $\Delta f(S,i,j)$ as $2P(\mu|\Delta f(S,i,j)) - 1$. Further-

more, this coefficient can be approximated as

$$
\begin{aligned}
2P(\mu|\Delta f(S,i,j)) - 1 &= 2 \cdot \text{sigmoid}\left(\frac{2\Delta f(S,i,j)\mu}{\sigma^2}\right) - 1 \\
&= \frac{2}{1+\exp(-\frac{2\Delta f(S,i,j)\mu}{\sigma^2})} - 1 \quad \text{\% We assume that } \Delta f(S,i,j) \text{ is close to zero} \\
&\approx 2\left(\frac{1}{2} + \frac{\mu}{2\sigma^2}\Delta f(S,i,j)\right) - 1 \\
&= \frac{\mu}{\sigma^2}\Delta f(S,i,j).
\end{aligned} \tag{41}
$$

Thus, the interaction loss can be rewritten as follows.

$$
\begin{aligned}
\text{Loss}_{\text{interaction}} &\approx \mathbb{E}_{i,j\in N}\left[\mathbb{E}_s\left[\mathbb{E}_{|S|=s,S\subseteq N}\left[\frac{\mu}{\sigma^2}\Delta f(S,i,j)^2\right]\right]\right] \\
&\approx \mathbb{E}_{A,B\subsetneq N,A\cap B=\emptyset,|A|=|B|=\alpha n}\left\{\mathbb{E}_{i\in A,j\in B}\left[\mathbb{E}_s\left[\mathbb{E}_{|S|=s,S\subseteq N}\left[\frac{\mu}{\sigma^2}\Delta f(S,i,j)^2\right]\right]\right]\right\} \\
&= \mathbb{E}_{A,B\subsetneq N,A\cap B=\emptyset,|A|=|B|=\alpha n}\left\{\mathbb{E}_s\left[\mathbb{E}_{|S|=s,S\subseteq N}\left[\frac{\mu}{\alpha^2 n^2\sigma^2}\Delta f(S,A,B)^2\right]\right]\right\}
\end{aligned} \tag{42}
$$

where $\frac{\mu}{\alpha^2 n^2 \sigma^2}$ is a positive constant. Therefore, we can use Equation (9) to approximate the interaction loss.

I Details of the Interaction and the Interaction Loss

Implementation of the interaction loss: In this section, we aim to introduce the implementation details of the interaction loss. Since the computation of the interaction loss in Equation (8) is expensive, we propose Equation (9) as an approximation of the interaction loss.

As the Equation (9) shows, we need to sample subsets of units A, B, and S for the computation of the interaction loss. For A and B, we randomly select 5% units from the intermediate-layer feature, respectively, and $A \cap B = \emptyset$. As for the subset S, we sampled units from $N \setminus A \setminus B$. Specifically, we first uniformly select a sampling rate from the range $[0, 1]$, and sample the subset S with this sampling rate. When we apply the interaction loss to DNNs with batch normalization layers (such as ResNets), we need to compute the interaction loss using a different track with the computation of the classification loss. *I.e.* when we compute the classification loss, parameters in the batch normalization layers are normally updated. In comparison, when we compute the interaction loss, we directly use the parameters in batch normalization layers without updating their values. In this way, we can apply the interaction loss to DNNs without causing disharmony with batch normalization layers.

The selection of the output score: In this paper, we consider a DNN f as a game, and take the output score of the DNN as the award score. Since the award score is usually a scalar value, for the DNN with a vectorized output, we usually take the output score before the softmax layer corresponding to the true class as the award score. Specifically, for DNNs with batch normalization layers, such as ResNets, we take the output score after the softmax layer corresponding to the true class as the award score.

Details of the datasets and training settings in Section 4: In the first experiment of Section 4, we used datasets with incorrect labels to reveal the relationship between the interaction and the over-fitting of DNNs. For the Tiny ImageNet dataset, we used all images from the first ten categories for the classification task, and each category contained 500 images. For the MNIST dataset, we randomly selected 10% images for training the DNN. In this case, each category contains 600 images. For the CelebA dataset, we randomly selected 1% training samples for the estimation of the gender. In order to build datasets with the over-fitted samples, we randomly selected 5% samples of the training samples, and replaced their labels with a randomly selected incorrect labels. In this case, when the DNN was well-learned and had a training loss close to zero, samples with incorrect labels could be considered as the over-fitted samples. In comparison, samples with correct labels were ordinary samples. Besides, in this experiment, the strength of interaction was normalized by the normalization term of each individual image.

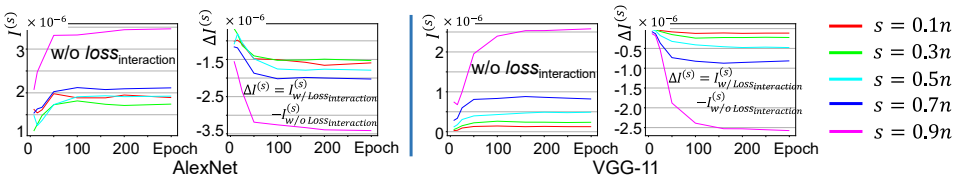

Figure 5: Curves of interaction component with different orders during the learning process. The interaction loss mainly penalized the interaction component with high order, which made the interaction component with low order became the main part of the interaction. All interaction were computed between activation units before the layer that we used interaction loss. $\Delta I^{(s)} = I^{(s)}_{\text{w/ Loss}_{\text{interaction}}} - I^{(s)}_{\text{w/o Loss}_{\text{interaction}}}$ denotes the change of interaction strength.

In the following experiment of Section 4, we explored the utility of the interaction loss. We trained DNNs using the CIFAR-10 dataset, the Tiny ImageNet dataset and the CelebA dataset. In order to make the DNN suffer from the over-fitting problem, for the CIFAR-10 dataset and the Tiny ImageNet dataset, we randomly generated 10% training samples to train the DNN. More specifically, we only used the first 10 categories for the classification task. Besides, for the CelebA dataset, we used 1% training samples to train the DNN for gender estimation. In these experiments, the strength of interactions was normalized by the normalization term over all training samples.

For fair comparison, the interaction loss and dropout were put in the same position of the DNN. Specifically, for AlexNet, we put the interaction loss or dropout before the third convolutional layer. For VGGs, we put the interaction loss or dropout behind the second block, *i.e.* behind the second ReLU layer of VGG-11 and behind the forth ReLU layer of VGG-13/16/19. We put the interaction loss or dropout behind the first block for ResNet-18 (RN-18), and behind the second block for ResNet-34 (RN-34). When we put the interaction loss or dropout at a high layer, we always put the interaction loss or dropout behind the first fully-connected layer.

The baseline value: In this section, we introduce details about the baseline value, which was proposed by (Ancona et al., 2019). Ancona et al. (2019) proposed a polynomial-time algorithm for Shapley values approximation to explain deep neural networks. When we estimate the attribution of input variables and the interaction between input variables, we usually use a baseline value to simulate the absence of variables. Many methods of estimating attributions require the definition of a baseline value as the anchor value corresponding to the absence of the variable. Setting an input variable to the baseline value indicates this input variable is toggled off. Let us focus on the task of object classification. We follow the settings in Ancona et al. (2019). If we aim to measure the interactions between input pixels, the baseline value is defined as the average pixel value among all samples. When we measure the interaction between activation units before the dropout operation, the baseline value is defined as zero.

J More Results of the Interaction Loss

J.1 Interaction Components with Different Orders for DNNs Learned with the $\text{Loss}_{\text{interaction}}$

In this section, we aimed to analyze which interaction component was more vulnerable to the interaction loss. We trained AlexNet and VGG-11 using the CIFAR-10 dataset. The weight λ of the interaction loss was set as 50. We computed the interaction components with different orders by setting $s = 0.1n, 0.3n, ..., 0.9n$. As Figure 5 shows, the interaction component with higher order was more suppressed than interaction components with other orders by the interaction loss, which was similar to the utility of dropout.

J.2 The Comparison between Dropout and $\text{Loss}_{\text{interaction}}$ when using with batch normalization

This section investigates the compatibility between dropout with batch normalization, and investigates the compatibility between the interaction loss with batch normalization. Preliminary results

	λ						Dropout
	0.0	0.5	1.0	2.0	5.0	10.0	
VGG-16+BN	44.6	45.6	**47.0**	43.6	43.8	45.8	35.2
VGG-19+BN	47.8	46.0	47.4	45.2	**48.6**	46.0	38.2

Table 4: Classification accuracy of VGG with batch normalization layers. The dropout operation decreased the classification accuracy due to the disharmony with batch normalization layers. In comparison, the interaction loss could further boost the performance of DNNs.

have been shown in Table 3. We added a batch normalization layer following each convolutional layer in VGG-16 and VGG-19, and compared the classification accuracy after putting dropout or the interaction loss in the DNN. For fair comparison, either the interaction loss or dropout was added following the fourth ReLU layer of VGG-16/19. Table 4 shows that the disharmony between dropout and batch normalization would hurt the performance of VGG-16 and VGG-19 more seriously. However, the use of the interaction loss would boost the classification accuracy. Specifically, the use of dropout decreased the classification accuracy of VGG-16 and VGG-19 by 9.4% and 9.6%, respectively, on the Tiny ImageNet dataset. Whereas, the use of the interaction loss boosted the classification accuracy of VGG-16 and VGG-19 by 2.4% and 0.8%, respectively. This phenomenon verified that dropout may hurt the performance of VGG-16/19 with batch normalization. The interaction loss did not suffer from the use of batch normalization.

K Details for Visualizing Interactions

This section provides the implementation details to visualize interactions modeled by DNNs. We used the CelebA dataset for visualization. There were 224×224 pixels in images of CelebA, which made it expensive to compute and visualize the interaction between any pairs of pixels. Thus, we divided the image into 16×16 grids, each grid contained 14×14 pixels. We considered these grids as input variables, and computed the interaction between any two adjacent grids. We sampled 100 times to approximate the interaction between two grids. For a specific grid, we averaged the interactions between the grid with its adjacent grids, and used the strength of the averaged interaction as the attribution value of this grid.