INTERPRETING AND BOOSTING DROPOUT FROM A GAME-THEORETIC VIEW

Hao Zhang, Sen Li, Yinchao Ma, Mingjie Li, Yichen Xie, and Quanshi Zhang
Shanghai Jiao Tong University

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 theoretical 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 alleviating 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.

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). 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 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 the game theory, as follows. We consider each input variable as a player, and consider the trained DNN as a game. Let \( x \) denote the input, and let \( f(x) \) denote the output of the DNN. For each input variable \( i \), we can compute its importance value \( \phi(i) \), which measures the numerical contribution of variable \( i \) to the output \( f(x) \). We notice that the importance value of variable \( i \) would be different when we mask another variable \( j \) w.r.t. the case when we do not mask \( j \). Thus, the interaction between input variables \( i \) and \( j \) is measured as the difference \( \phi_{w/j}(i) - \phi_{w/o}(i) \).

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 the 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.

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) We design
a novel loss function to penalize the strength of interactions, which improves the performance of 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 dropping 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. (2013), 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 above views, the 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 variables w.r.t. variables, it is the unique unbiased metric that fairly allocates the numerical contribution of each player to the overall award. Given a set of players $N = \{1, 2, \ldots, n\}$, $2^N \triangleq \{S | S \subseteq N\}$ denotes all possible subsets of $N$. A game $f : 2^N \rightarrow \mathbb{R}$ 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 numerical contribution of each player to the overall award, as shown in Equation (1). The Shapley value of player $i$ in the game $f$, $\phi(i|N) = f(N)$, is computed as follows.

$$\sum_{i=1}^{n} \phi(i|N) = f(N) - f(\emptyset), \quad \phi(i|N) = \sum_{S \subseteq N \setminus \{i\}} \frac{P_{\text{Shapley}}(S|N \setminus \{i\})[f(S \cup \{i\}) - f(S)]}{2^{|M|}}$$ (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.

3 GAME-THEORETIC EXPLANATIONS OF DROPOUT

Preliminaries: Shapley values. The Shapley value was initially proposed by Shapley (1953) in the game theory. It is considered as a unique unbiased metric that fairly allocates the numerical contribution of each player to the overall award. Given a set of players $N = \{1, 2, \ldots, n\}$, $2^N \triangleq \{S | S \subseteq N\}$ denotes all possible subsets of $N$. A game $f : 2^N \rightarrow \mathbb{R}$ 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 numerical contribution of each player to the overall award, as shown in Equation (1). The Shapley value of player $i$ in the game $f$, $\phi(i|N) = f(N)$, is computed as follows.

$$\sum_{i=1}^{n} \phi(i|N) = f(N) - f(\emptyset), \quad \phi(i|N) = \sum_{S \subseteq N \setminus \{i\}} \frac{P_{\text{Shapley}}(S|N \setminus \{i\})[f(S \cup \{i\}) - f(S)]}{2^{|M|}}$$ (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 the game theory. In the game theory, some players may form a coalition to compete with other players to win an award \cite{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 score before the softmax layer corresponding to the true class as the score.

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 score $f(S)$, input variables in $N \setminus S$ are replaced by the reference value (e.g. being masked), and input variables in $S$ remain unchanged. Both \cite{Singh et al. 2019} and Appendix \[B\] introduce details about the reference 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 $S$. In particular, when we consider neural activations before dropout, such activations are usually non-negative after the ReLU operation. Then, their reference values are set to 0.

Interactions encoded by DNNs. In this section, we introduce how to use the interaction defined in the 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 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 \cite{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.

$$I(i, j) \equiv \phi(S_{ij}[N \setminus ij]) - [\phi(i[N \setminus \{j\}]) + \phi(j[N \setminus \{i\}])] = \sum_{S \subseteq N \setminus \{i, j\}} P_{\text{Shapley}}(S\setminus N \setminus \{i, j\}) \Delta f(S, i, j) \tag{2}$$

where $\Delta f(S, i, j) \equiv f(S \cup \{i, j\}) - f(S \cup \{j\}) - f(S \cup \{i\}) + f(S)$. $\phi(i[N \setminus \{j\}])$ and $\phi(j[N \setminus \{i\}])$ correspond to the contribution to the DNN output when $i$ and $j$ work individually. Theoretically, $I(i, j)$ is also equal to the difference between the Shapley value of pixel $i$ when we mask another pixel $j$ w.r.t. the case when we do not mask $j$. Please see Appendix \[C\] 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.

$$I(i, j) = \mathbb{E}_s \left[ I^{(s)}(i, j) \right] \quad I^{(s)}(i, j) \equiv \mathbb{E}_{S \subseteq N \setminus \{i, j\} : |S| = s} \left[ \Delta f(S, i, j) \right] \tag{3}$$

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 pixels $i$ 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 amount of input variables, i.e. the interaction depends on the context of a large scale.

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 $T \cup \{i, j\}$. For example, let $S$ represent the face, and let $T \cup \{i, j\}$ represent pixels
of an eye in the face. 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. Let $R^T(i, j)$ quantify the marginal award obtained from the inference pattern of an eye. All interaction effects from smaller coalitions $T' \subset T$ are removed from $R^T(i, j)$.

According to the above example, Let $T' \subset T$ 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 co-occurrence of all pixels in the eye $T \cup \{i, j\}$ can exclusively trigger the inference pattern $T \cup \{i, j\}$, but cannot trigger the smaller inference pattern $T' \cup \{i, j\}$. Thus, the $s$-order interaction can be decomposed into components w.r.t. all inference patterns $T \cup \{i, j\}, T \subseteq S$.

$$I^{(s)}(i, j) = \mathbb{E}_{S \subseteq N \setminus \{i, j\} \mid |S| = s} \left[ \sum_{T \subseteq S} R^T(i, j) \right] = \sum_{0 \leq q \leq s} \binom{s}{q} J^{(q)}(i, j)$$

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

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 $S' \subseteq S$ denotes the input variables that remain in the context $S$ after the dropout operation. Then, $I^{(s)}_{\text{dropout}}(i, j)$ can be computed as follows.

$$I^{(s)}_{\text{dropout}}(i, j) = \mathbb{E}_{S \subseteq N \setminus \{i, j\} \mid |S| = s} \left[ \mathbb{E}_{r \sim B(s, p)} \left[ \sum_{T \subseteq S' \mid |T| = r} R^T(i, j) \right] \right] = \mathbb{E}_{r \sim B(s, p)} \left[ \sum_{0 \leq q \leq r} \binom{r}{q} J^{(q)}(i, j) \right]$$

where $r$ is the number of units in $S'$. The interaction only comprises the marginal awards from the inference patterns consisting of at most $r \sim B(s, p)$ variables, where $B(s, p)$ is the binomial distribution with the sample number $s$ and the sample rate $p$. Please see Appendix E for the proof.

Since $r = |S'| \leq s$, we have

$$1 \geq \binom{r}{1} J^{(1)}(i, j) \geq \binom{r}{2} J^{(2)}(i, j) \geq \cdots \geq \binom{r}{r} J^{(r)}(i, j) \geq 0.$$  

Thus, when we apply the dropout, the $s$-order interaction will be suppressed as follows.

$$\frac{I^{(s)}_{\text{dropout}}(i, j)}{I^{(s)}(i, j)} = \frac{\sum_{0 \leq q \leq r} \binom{r}{q} J^{(q)}(i, j)}{\sum_{0 \leq q \leq s} \binom{s}{q} J^{(q)}(i, j)} \leq 1$$

Equation (7) proves that dropout can suppress interactions of each order encoded by the DNN. Equation (6) shows that inference patterns with more activation units (with higher orders) are more vulnerable to the dropout operation.

**Experimental verification:** Besides the theoretical proof, we also conducted experiments to illustrate how dropout suppresses the interaction modeled by DNNs, which is a verification of Equation (7). In experiments, we trained AlexNet (Krizhevsky et al. 2012) on CIFAR-10 (Krizhevsky &
Table 1: Comparison of the interaction strength between the over-fitted samples and ordinary samples. DNNs usually encodes more interaction for the over-fitted samples than ordinary samples, which reveals the relationship between the interaction strength with the over-fitting of DNNs. We measure the interaction strength in image pixels.

| Dataset        | Model     | Ordinary samples | Over-fitted samples |
|----------------|-----------|------------------|---------------------|
| MNIST          | ResNet-44 | 2.17e-3          | 3.64e-3             |
| Tiny-ImageNet  | ResNet-44 | 2.57e-3          | 2.89e-3             |
| CelebA         | ResNet-34 | 6.46e-3          | 1.17e-2             |

Hinton [2009] and AlexNet (Krizhevsky et al. [2012], VGG-16/19 (Simonyan & Zisserman 2015b) on CelebA (Liu et al. 2015) with and without the dropout. Figure 1(a) compares the strength of interactions encoded by DNNs, which are learned with or without the dropout operation. We averaged the strength of interactions over images, i.e. \( I = \mathbb{E} \left[ |E_{(i,j)}| \right] \), where \( I(i,j) \) is obtained according to Equation (2). Note that accurately computing the interaction of two input variables is an NP-hard problem. Thus, we applied a sampling-based method (Castro et al. 2009) to approximate the strength of interactions. Please see Appendix I for details. We found that dropout could effectively suppress the strength of the interactions, which verified our proof.

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 strength of interactions 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 randomly selected 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 (He et al., 2016) for the classification using the Tiny ImageNet dataset and the CelebA dataset, and trained ResNet-44 for object classification task using the MNIST dataset. We computed the average interaction strength on the over-fitted samples and ordinary samples, respectively.

As Table 1 shows, the over-fitted samples usually contained stronger interactions than ordinary samples. Please see Appendix I for details.

Understanding of dropout. We have proved that dropout can decrease the strength of interactions encoded in DNNs in Section 3. Besides, above 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 the 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.

\[
\text{Loss} = \text{Loss}_{\text{classification}} + \lambda \text{Loss}_{\text{interaction}},
\]

where \( \lambda > 0 \) is the weight of the interaction loss. The interaction loss is defined as follows. We regard 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 \( \text{Loss}_{\text{interaction}} = \mathbb{E}_{i,j \in N, i \neq j} \left[ |I(i,j)| \right] \). The inference value in the computation of \( I(i,j) \) is set to 0, as is explained in the paragraph understanding DNNs via the game theory, Section 3.

\[
\text{Loss}_{\text{interaction}} = \mathbb{E}_{i,j \in N, i \neq j} \left[ |I(i,j)| \right] = \mathbb{E}_{i,j \in N, i \neq j} \left[ \left| \sum_{S \subseteq N \backslash \{i,j\}} P_{\text{Shapley}}(S|N \backslash \{i,j\}) \Delta f(S, i, j) \right| \right]
\]

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 \( A, B \subseteq N, A \cap B = \emptyset \), 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. \( S \subseteq N \backslash A \backslash B \). Based on Equation (2),...
$S \sim P_{\text{Shapley}}(S \mid N \setminus A \setminus B)$. Therefore, we can get the following approximation of Loss_{interaction}. Please see Appendix H for the proof.

$$\text{Loss}_{interaction} = E_{A,B} E_{r \in N \setminus A \cap B = \emptyset, |A| = |B| = \alpha |N|} \left\{ E_r \left[ E_{S \subseteq N \setminus A \setminus B} \left| \Delta f(S, A, B)^2 \right| \right] \right\}$$

where $\Delta f(S, A, B) \overset{\text{def}}{=} f(S \cup A \cup B) - f(S \cup A) - f(S \cup B) + f(S)$.

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

**Advantage 1:** The interaction loss explicitly controls the penalty of the strength of interactions by adjusting the weight $\lambda$ in Equation (8). The explicit control of the interaction strength is crucial, because it is important to make a careful balance between the over-fitting and under-fitting problems during the learning of the DNN, thereby controlling the over-fitting level of the DNN. The weight $\lambda$ 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 $\lambda$ may lead to the under-fitting problem, while a small value of $\lambda$ may increase the risk of over-fitting. Unlike the interaction loss, it is difficult for people to explicitly control the strength of dropout during the learning process.

**Advantage 2:** Due to the disharmony between the 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 Loss_{interaction} using a new track, which is different from the track of computing Loss_{classification}. In other words, we do not update the parameter in batch normalization layers when we compute Loss_{interaction}. Please see Appendix H for more discussions.

**Experiments:** We trained DNNs for the classification based on the CIFAR-10 dataset (Krizhevsky & Hinton, 2009). We trained a total of seven DNNs, including AlexNet (Krizhevsky et al., 2012), VGG-11/13/16 (Simonyan & Zisserman, 2015a), and ResNet-20/32/44 (He et al., 2016).

- **Improving the performance of DNNs via the interaction loss.** We trained the DNNs with different weights of the interaction loss, and evaluated its testing accuracy. Table 2 compares their classification accuracy when we used different values of $\lambda$ to train DNNs. We found that the interaction loss could boost the performance of DNNs. In particular, because the dropout operation was not compatible with batch normalization (Li et al., 2019). The dropout operation would decrease the classification accuracy in VGG-11/13. In comparison, learning with the interaction loss did not suffer from the use of batch normalization.

| DNN      | Weight | AlexNet | VGG-11 | VGG-13 | VGG-16 | DNN      | Weight | ResNet-20 | ResNet-32 | ResNet-44 |
|----------|--------|---------|--------|--------|--------|----------|--------|-----------|-----------|-----------|
| 0.0      | 64.8   | 60.65   | 59.91  | 58.95  |        | 0.0      | 63.75  | 64.17     | 63.91     | 63.91     |
| 50.0     | 68.24  | 63.00   | 62.47  | 62.12  |        | 0.001    | 63.74  | 64.91     | 65.30     |
| 100.0    | 67.80  | 63.18   | 63.63  | 62.82  |        | 0.0003   | 63.4   | 64.38     | 65.89     |
| 200.0    | 69.6   | 63.85   | 65.16  | 62.78  |        | 0.01     | 64.66  | 64.9      | 64.97     |
| 500.0    | 68.56  | 64.47   | 64.84  | 61.73  | 64.56  | 0.003    | 64.19  | 65.59     |
| 1000.0   | 60.22  | 66.97   | 59.99  | 61.64  |        | 0.01     | 62.92  | 63.04     | 67.47     |
| Dropout  | -      | 59.65   | 59.39  | 59.51  | 59.65  | Dropout  | 68.56  | 64.82     | 65.53     |

Table 2: Performance of DNNs learned using different weights of interaction loss.

5 CONCLUSION

In this paper, we aim to 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 our 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 found that the interaction loss could reduce the strength of interaction effectively and further boost the performance of DNNs.
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.

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 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.

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 Suin 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.

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

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

Chandan Singh, W James Murdoch, and Bin Yu. Hierarchical interpretations for neural network predictions. International Conference on Learning Representations, 2019.

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  DETAILS ABOUT THE REFERENCE VALUE

This section provides details about the reference value mentioned in Section 3. When estimating the attribution of input variables and the interaction between input variables, we usually use a reference value to simulate the absence of variables. Many attribution methods require the definition of a reference value to indicate the absence of information. Setting an input variable to the reference value to simulate the absence of variables. Many attribution methods require the definition of a reference value to indicate the absence of information. Setting an input variable to the reference value to simulate the absence of variables.

C  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 pixel $i$ when we mask another pixel $j$ 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.

$$I(i, j) \triangleq \phi(S_{ij}|N_{ij}) - \left[ \phi(i|N \setminus \{j\}) + \phi(j|N \setminus \{i\}) \right] = \sum_{S \subseteq N \setminus \{i, j\}} \frac{(n - |S| - 2)! |S|!}{(n - 1)!} \Delta f(S, i, j) \quad (11)$$

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.
\[ I(i, j) \triangleq \phi_{\text{with}}(i) - \phi_{\text{without}}(i) \]

\[ = \sum_{S \in N \setminus \{i\}} \sum_{j \in S} \frac{(n-|S|-2)!|S|!}{(n-1)!} \left[ f(S \cup \{i\}) - f(S) \right] \]

\[ = \sum_{S \in N \setminus \{i\}} \sum_{j \in S} \frac{(n-|S|-2)!|S|!}{(n-1)!} \left[ f(S \cup \{i\}) - f(S) \right] \]

\[ = \sum_{S \in N \setminus \{i\}} \sum_{j \in S} \Delta f(S, i, j) \] (12)

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

Equation (11) and (12) prove the consistency of the two understandings towards the interaction mentioned in Section 3.

D Detailed Proof for Equation (4)

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

\[ f^{(s)}(i, j) = \mathbb{E}_{S \subseteq N \setminus \{i, j\}, |S| = s} \left[ \sum_{T \subseteq S} R^T(i, j) \right] \] (13)

\[ = \frac{1}{n^2} \sum_{S \subseteq N \setminus \{i, j\}, |S| = s} \left( \sum_{T \subseteq S} R^T(i, j) \right) \] (14)

\[ = \frac{1}{n^2} \sum_{0 \leq q \leq s} \left\{ \sum_{T \subseteq N \setminus \{i, j\}, |T| = q} \left[ \binom{n-q-2}{s-q} \binom{n-2}{q} R^T(i, j) \right] \right\} \] (15)

\[ = \frac{1}{n^2} \sum_{0 \leq q \leq s} \left[ \binom{n-q-2}{s-q} \binom{n-2}{q} \mathbb{E}_{T \subseteq N \setminus \{i, j\}, |T| = q} R^T(i, j) \right] \] (16)

\[ = \sum_{0 \leq q \leq s} \binom{s}{q} J^q(i, j) \] (17)

Note that from Equation (14) to Equation (15), we re-arrange the sum so that it become a summation of marginal awards from small inference patterns to large inference patterns. In Equation (17), \( J^q(i, j) = \mathbb{E}_{T \subseteq N \setminus \{i, j\}, |T| = q} [R^T(i, j)] \) denotes the average interaction between \( i \) and \( j \) given all potential inference patterns \( T \cup \{i, j\} \) with a fixed inference pattern size \( |T| = q \).

E Detailed Proof for Equation (5)

This section gives the detailed proof for Equation (5) in Section 3. The interaction of \( i \) and \( j \) with dropout, \( I^{(s)}_{\text{dropout}}(i, j) \), can be computed as
\[ I^{(s)}(i, j) = \mathbb{E}_{S \subseteq N \setminus \{i, j\}, |S| = s} \left\{ \mathbb{E}_{r \sim \mathcal{B}(s, p)} \left[ \mathbb{E}_{S' \subseteq S, |S'| = r} \left( \sum_{T \subseteq S'} R^T(i, j) \right) \right] \right\} \]

(18)

\[ = \mathbb{E}_{r \sim \mathcal{B}(s, p)} \left[ \frac{1}{r^2} \sum_{S' \subseteq S \setminus \{i, j\}} \left( \sum_{T \subseteq S'} R^T(i, j) \right) \right] \]

(20)

\[ = \mathbb{E}_{r \sim \mathcal{B}(s, p)} \left[ \frac{1}{r^2} \sum_{S' \subseteq N \setminus \{i, j\}} \left( \sum_{T \subseteq S'} R^T(i, j) \right) \right] \]

(22)

\[ = \mathbb{E}_{r \sim \mathcal{B}(s, p)} \left[ \sum_{0 \leq q \leq r} \binom{r}{q} J^q(i, j) \right] \]

(23)

Note that the derivation from Equation (22) to Equation (23) is similar to the proof in Appendix [D]

F  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 formed by \( T' \subseteq T \) are removed from \( R^T(i, j) \).

\[ R^T(i, j) = \Delta f(T, i, j) - \sum_{T' \subseteq T} R^T(i, j) \]

(24)

It is easy 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), since \( \sum_{T \subseteq S} R^T(i, j) = R^S(i, j) + \sum_{T \subseteq S} R^T(i, j) = f(S, i, j) \). Thus, according to Equation (3), this equality holds.

G  Computation of the Strength of Interactions

In this section, we aim to introduce implementation details to compute the strength of interactions between input variables modeled by the DNN mentioned in Section [3]. Let \( i, j \) 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 strength of interactions, as follows.

\[ I = \mathbb{E}_{i,j} \left[ \mathbb{E}_{s} \left[ \mathbb{E}_{S \subseteq N \setminus \{i, j\}, |S| = s} [\Delta f(S, i, j)] \right] \right] \]

(25)

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

In order to further boost the computation efficiency, we divided each image into \( 16 \times 16 \) grids in experiments. We consider each grid as an input variable, and computed the interaction between two
We further assume that \( \mu \) away from zero to reduce \( \Delta f(S, i, j) \). Thus, we set the coefficient of \( \Delta f(S, i, j) \) a small value close to zero. In this way, \( \Delta f(S, i, j) \) has the same sign with \( \mu \). In this case, we need to push \( \Delta f(S, i, j) \) towards to zero. In comparison, if \( P(\mu|\Delta f(S, i, j)) \approx 1 \), then it is highly possible that \( \Delta f(S, i, j) \) has the same sign with \( \mu \). In this case, we need to push \( \Delta f(S, i, j) \) away from zero to reduce \( \mu \). Thus, we set the coefficient of \( \Delta f(S, i, j) \) as \( 2P(\mu|\Delta f(S, i, j)) - 1 \). We further assume that \( \Delta f(S, i, j) \) is a small value close to zero. In this way, \( 2P(\mu|\Delta f(S, i, j)) - 1 \) \( \Delta f(S, i, j) \approx \frac{\mu}{4\sigma^2} \Delta f(S, i, j)^2 \).

Thus, the interaction loss can be rewritten as follows.

\[
\text{Loss}_{\text{interaction}} \approx \mathbb{E}_{i,j \in N} \left[ \mathbb{E}_s \left[ \mathbb{E}_{|S| = s, S \subseteq N} \left[ \frac{\mu}{4\sigma^2} \Delta f(S, i, j)^2 \right] \right] \right] \\
= \mathbb{E}_{A,B \subseteq \mathcal{N}, A \cap B = \emptyset, |A| = |B| = |S|} \left\{ \mathbb{E}_{i \in A, j \in B} \left[ \mathbb{E}_{|S| = s, S \subseteq N} \left[ \frac{\mu}{4\sigma^2} \Delta f(S, i, j)^2 \right] \right] \right\} \\
\approx \mathbb{E}_{A,B \subseteq \mathcal{N}, A \cap B = \emptyset, |A| = |B| = |S|} \left\{ \mathbb{E}_{|S| = s, S \subseteq N} \left[ \mathbb{E}_{i \in A, j \in B} \left[ \frac{\mu}{4\sigma^2} \Delta f(S, i, j)^2 \right] \right] \right\} \\
\approx \mathbb{E}_{A,B \subseteq \mathcal{N}, A \cap B = \emptyset, |A| = |B| = |S|} \left\{ \mathbb{E}_{|S| = s, S \subseteq N} \left[ \frac{\mu}{4|A||B|\sigma^2} \Delta f(S, A, B)^2 \right] \right\} \\
\]

Therefore, we can use Equation (10) to approximate the interaction loss. As the Equation (10) 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.
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, the statistic mean and the statistic var of the batch normalization operation can be updated. In comparison, when we compute the interaction loss, we only use the statistic mean and the statistic var of the batch normalization operation, yet without updating their values. In this way, parameters of the batch normalization operation are not affected by the sampling process of the interaction loss.

\section{Details of the Datasets and Training Settings in Section 4}

In experiments 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 used 10\% images for training the DNN. In this case, each category contains 600 images. For the CelebA dataset, we used 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 over-fitted samples. In comparison, samples with correct labels were ordinary samples.