The Thirty-Ninth AAAI Conference on Artificial Intelligence (AAAI-25)

Monitoring Primitive Interactions During the Training of DNNs

Jie Ren[1], **Xinhao Zheng**[1], **Jiyu Liu**[1,2*], **Andrew Lizarraga**[3],
Ying Nian Wu[3], **Liang Lin**[4], **Quanshi Zhang**[1†]

[1]Shanghai Jiao Tong University
[2]Dartmouth College
[3]University of California, Los Angeles
[4]Sun Yat-Sen University

Abstract

This paper focuses on the newly emerged research topic, *i.e.*, whether the complex decision-making logic of a DNN can be mathematically summarized into a few simple logics. Beyond the explanation of a static DNN, in this paper, we hope to show that the seemingly complex learning dynamics of a DNN can be faithfully represented as the change of a few primitive interaction patterns encoded by the DNN. Therefore, we redefine the interaction of principal feature components in intermediate-layer features, which enables us to concisely summarize the highly complex dynamics of interactions throughout the learning of the DNN. The mathematical faithfulness of the new interaction is experimentally verified. From the perspective of learning efficiency, we find that the interactions naturally belong to five groups (*reliable, withdrawn, forgotten, betraying*, and *fluctuating interactions*), each representing a distinct type of dynamics of an interaction being learned and/or being forgotten. This provides deep insights into the learning process of a DNN.

1 Introduction

In the field of interpretable artificial intelligence, one of the fundamental objectives of a theory system is *to let the seemingly extremely complex decision-making logic of a deep neural network (DNN) be faithfully explained as a small set of simple logics.* Unlike other explanation methods (Elhage et al. 2021; Meng et al. 2022; Zhao et al. 2022; Park et al. 2022; Olsson et al. 2022; Fel et al. 2023), this is a newly emerging mathematical problem in recent years, because it aims to answer whether the essential logic of a DNN is simple enough to be explained to human beings, *i.e.*, the existence of human-understandable explanation for a DNN.

Towards this problem, an interaction-based theory system has been built up recently, containing about 20 papers (surveyed by Ren et al. (2024)). Typically, Ren et al. (2023a); Li and Zhang (2023) discovered and Ren et al. (2024) proved[1] that we can always use the numerical utility of a few symbolic interactions between input variables to accurately explain all subtle changes of network outputs under a massive number of input variations. It is also found (Zhou et al. 2024) that the complexity of interactions could explain the generalization power of DNNs.

Beyond the above explanation of a static (trained) DNN, in this paper, we hope to explore whether the entire learning dynamics of a DNN, which is believed to be much more complex than a static DNN, can also be concisely explained as symbolic interactions. The explanation of the complex learning dynamics is mentioned by several previous studies (Zhou et al. 2024; Li and Zhang 2023; Ren et al. 2024; Chen et al. 2024; Cheng et al. 2024), and they all considered this as the last piece of the puzzle of the interaction-based explanation system, and also one of the biggest challenges that has hampered the field for years.

The challenges of explaining learning dynamics come from the high-dimensional changes in network parameters, which are complex and even chaotic. However, we hope

- to summarize the highly complex learning dynamics of a DNN into the dynamics of a few interactions;
- to explain the learning efficiency of a DNN, *i.e.*, answering how many interactions are learned from the beginning of the training and how many interactions are discarded later;
- to clarify whether the DNN learns all primitive patterns simultaneously.

Originally, the interaction metric was used to quantify the non-linear relationship encoded by a DNN. For example, Figure 1(b) shows a DNN implicitly encodes an interaction between two eye patches and a nose patch, and this interaction makes a numerical utility on the classification score of *cat*. Masking any one among three patches will invalidate this interaction and remove its utility from the output score.

However, in this study, we hope to use a few salient interactions (interactions with large interaction effects) to explain even more complex learning dynamics. Thus, how to reduce

[*] This work was done when Jiyu Liu was at Shanghai Jiao Tong University.

[†] Quanshi Zhang is the corresponding author. He is with Department of Computer Science and Engineering, John Hopcroft Center, at Shanghai Jiao Tong University, China. zqs1022@sjtu.edu.cn

[1] Sparsity of interactions is proven by Ren et al. (2024) under three common conditions for smooth inference on masked samples. Please see Appendix C for details.

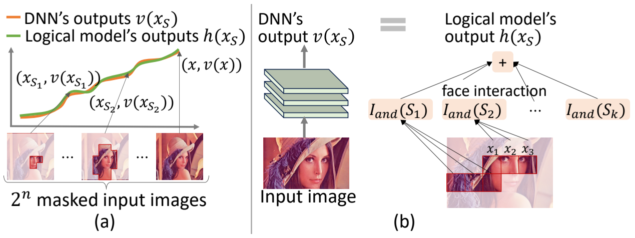


Figure 1: (a) We construct a logical model to mimic the DNN's outputs on randomly masked inputs. (b) The output of the logical model is the sum of all interaction utilities in the input encoded by the DNN.

the complexity of the explanation and concisely summarize dynamics is the key point of this study.

Therefore, we redefine interactions on feature components in intermediate layers to obtain concise interactions for explanation. We consider the top-ranked principal feature components as basic "input variables" for interactions. Experiments have shown that the newly defined interaction enables us to use much sparser interactions between much fewer (less than 10) principal feature components to explain most information of a DNN's learning dynamics (see Figure 2 and Figure 3) without losing explanation fidelity.

Surprisingly, we find that all interactions naturally belong to the following five categories in terms of learning efficiency. (1) *Reliable* interactions are stably learned throughout the entire training process of the DNN. (2) *Withdrawn* interactions are learned in early epochs and then discarded in later epochs. (3) *Forgotten* interactions are initially salient but gradually forgotten in the following epochs. (4) *Betraying* interactions are learned to represent a certain classification utility (toward a specific category) in early epochs, but later shifted to an opposite classification utility in later epochs. (5) *Fluctuating* interactions keep fluctuating during the training of the DNN. In conclusion, we can consider reliable and forgotten interactions as efficiently learned knowledge, while betraying and withdrawn interactions reflect the trial-and-error process during learning.

Although interactions have encoded mixed semantics, the decomposition of the complex learning dynamics of DNNs into a few concise interactions still provides a new perspective to understanding the learning behavior of a DNN.

Contributions of this study are as follows. (1) We redefine interactions, which enable us to concisely summarize the highly complex learning dynamics of a DNN into the change of a few interactions. (2) We find that all interactions naturally belong to five types, which reflect the DNN's distinctive learning behavior of different inference patterns. (3) Various experiments have verified the mathematical faithfulness of the interaction-based explanation.

2 Primitive Interactions in DNNs

Preliminary: Interactions

In this section, let us introduce the interaction, as well as a set of properties of interactions (Li and Zhang 2023; Ren et al. 2023a, 2024), as mathematical guarantees for the faithfulness of interaction-based explanation.

Defintion of AND-OR interactions. Given a trained DNN v, let $x \in \mathbb{R}^n$ denote an input sample with n input variables (*e.g.,* an image with n image patches and a sentence with n words), indexed by $N = \{1, 2, \ldots, n\}$. The DNN's output is denoted by $v(x) \in \mathbb{R}$. For example, in multi-category classification, $v(x)$ is usually defined as the following confidence score (Deng et al. 2022).

$$v(x) = p(y = y^*|x)/(1 - p(y = y^*|x)) \tag{1}$$

where y^* denotes the ground-truth label of the input x. Given the trained model $v(\cdot)$ and a set $S \subseteq N (S \neq \emptyset)$ of input variables, numerical utilities of the AND interaction $I_{\text{and}}(S|x)$ and the OR interaction $I_{\text{or}}(S|x)$ between these variables can be computed as follows.

$$\begin{aligned} I_{\text{and}}(S|x) &= \sum\nolimits_{L \subseteq S} (-1)^{|S|-|L|} v_{\text{and}}(x_L), \\ I_{\text{or}}(S|x) &= -\sum\nolimits_{L \subseteq S} (-1)^{|S|-|L|} v_{\text{or}}(x_{N \setminus L}) \end{aligned} \tag{2}$$

where $v_{\text{and}}(x_L) = 0.5v(x_L) + \gamma_L$ and $v_{\text{or}}(x_L) = 0.5v(x_L) - \gamma_L$ represent the output component for AND interactions and the output component for OR interactions, respectively. x_L denotes a masked input sample, in which the variables in $N \setminus L$ are masked.[2] Then, $v_{\text{and}}(x_L) \in \mathbb{R}$ denotes the output on the masked input. γ_L is a learnable parameter to decompose $v(x_L)$ into $v_{\text{and}}(x_L)$ and $v_{\text{or}}(x_L)$.

In this way, the computation of $v_{\text{and}}(x_L)$ and $v_{\text{or}}(x_L)$ is implemented by learning the parameter γ_L via a LASSO-like sparisity loss for interactions, *i.e.,* $\min_{\{\gamma_L\}} \sum_{S \subseteq N, S \neq \emptyset} [|I_{\text{and}}(S|x)| + |I_{\text{or}}(S|x)|]$.

Understanding AND-OR interactions via the *universal-matching property* of interactions. Each AND interaction $I_{\text{and}}(S|x)$ represents a non-linear relationship between variables in S, *i.e.*, the co-appearance of all variables in S will add a utility $I_{\text{and}}(S|x)$ to the model output. On the other hand, each OR interaction $I_{\text{or}}(S|x)$ represents the OR relationship encoded by the model. The appearance of any variables in S will add $I_{\text{or}}(S|x)$ to the output score. Figure 1(b) shows an example of an AND interaction $I_{\text{and}}(S|x)$ of a face in the surrogate model. $I_{\text{and}}(S|x)$ is triggered only when x_1, x_2, x_3 all appear in the input image. In comparison, an OR interaction $I_{\text{or}}(S|x)$ is triggered when any input variable in S appears.

According to Theorem 2.1, we construct a surrogate logical model based on all AND-OR interactions. Given each masked input x_S, the surrogate model first identifies a set of interactions triggered by x_S based on the AND-OR logic rule. Then, utilities of all these interactions are summed up as the output $h(x_S)$. Theorem 2.1 proves that no matter how the input is randomly masked, the model output on the masked sample can always be approximated by the surrogate model based on utilities of a few interactions.

[2] People usually mask input variables in $N \setminus L$ using baseline values $\{b_i\}$ (also called reference values) (Ancona, Öztireli, and Gross 2019; Covert, Lundberg, and Lee 2020) to replace the original values in these input variables, *i.e.,* setting $x_i = b_i$ if $i \in N \setminus L$.

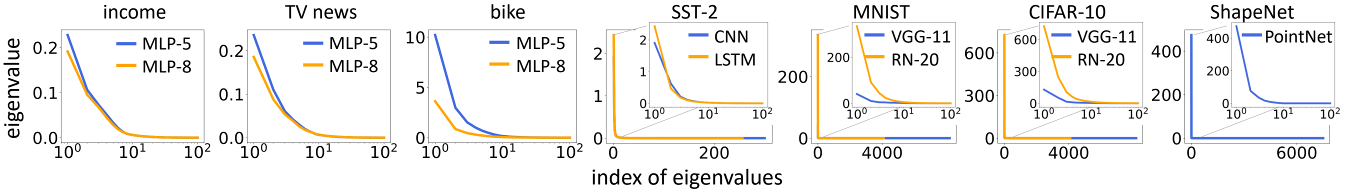


Figure 2: Significance (eigenvalue λ_i) of feature components in a descending order. The plot on the top-right side of each subfigure zooms in the range of 1st−100th eigenvalues for better visualization.

Theorem 2.1 (Universal-approximation property of interactions, proved in Appendix E)**.** *Given an input sample x, let $\Omega_{salient}$ denote the set of salient interactions. We consider interactions* w.r.t. $|I_{and}(S|x)| \geq \tau$ *or* $|I_{or}(S|x)| \geq \tau$ *as salient interactions. We construct the surrogate model $h(\cdot)$ to use AND-OR interactions extracted from the DNN $v(x_S)$ for inference,* $h(x_S) = \sum_{L\subseteq N, L\neq\emptyset} I_{and}(L|x) \cdot \mathbb{1}(x_S \textit{ triggers the AND relation } L) + \sum_{L\subseteq N, L\neq\emptyset} I_{or}(L|x) \cdot \mathbb{1}(x_S \textit{ triggers the OR relation } L) + v(x_\emptyset) = \sum_{L\subseteq S, L\neq\emptyset} I_{and}(L|x) + \sum_{L\cap S\neq\emptyset} I_{or}(L|x) + v(x_\emptyset) \approx \sum_{L\subseteq S, L\neq\emptyset, L\in\Omega_{salient}} I_{and}(L|x) + \sum_{L\cap S\neq\emptyset, L\in\Omega_{salient}} I_{or}(L|x) + v(x_\emptyset)$. *$v(x_\emptyset)$ is a constant that represents the model output when all input variables are masked. No matter how we arbitrarily mask the variables in x to obtain the masked inputs x_S* w.r.t. *a random subset $S \subseteq N$, the surrogate model $h(x_S)$ can always mimic the DNN output $v(x_S)$ on the masked input x_S,* i.e., $\forall S \subseteq N, h(x_S) = v(x_S)$.

Sparsity property of salient interactions. Let us enumerate all 2^n subsets $S \subseteq N$ and compute their interaction utilities. Ren et al. (2024) have proven[1] that DNNs usually encode very sparse salient interactions, *i.e.,* the number of salient interactions is $O(n^\delta)$ ($\delta \in [1.9, 2.2]$ empirically), which is extremely sparse *w.r.t.* all 2^n subsets.

Generalization property. Li and Zhang (2023) have discovered the generalization ability of interactions. That is, people can extract a common set of interactions from different (but similar) inputs or different models, and these interactions are discriminative for classification.

The above sparsity, universal approximation, and generalization properties of interactions ensure that the interactions can be considered as primitive inference patterns for the model inference.

Primitive Interactions on Features

Although we usually extract a few interactions from a fixed DNN, tracking the dynamics of interactions in all intermediate DNNs through the entire training process may significantly complicate the explanation. This is because DNNs trained after different epochs may generate fully different interactions. Therefore, the first challenge is to redefine the interaction to simplify the explanation, and meanwhile, the newly defined interaction should be powerful enough to faithfully reflect major changes in all training epochs.

Therefore, instead of taking raw pixels/words/3D points as input variables, we redefine interactions on principal feature components shared by all intermediate DNNs. Let us train a DNN, and collect the DNN trained after K different checkpoints (epochs). Given an input sample, we extract the feature from a certain intermediate layer of the DNNs at these K checkpoints, denoted by $f^{(1)}, f^{(2)}, \dots, f^{(K)} \in \mathbb{R}^m$. Subsequently, we conduct principal component analysis (PCA) on the K features to compute the top r principal directions (eigenvectors) $q_1, q_2, \dots, q_r \in \mathbb{R}^m$ corresponding to the largest r eigenvalues. In this way, we extract feature components along the top r principal directions, so as to use these feature components as basic "input variables" to define interactions. Specifically, for the intermediate-layer feature $f^{(k)}$ extracted after k epochs, we decompose the feature $f^{(k)}$ into the following $(r+2)$ feature components.

Model	Dataset	Using raw $f^{(k)}$	Using $\sum_{i=1}^{10} f_i + \bar{f}$
MLP-5	income	0.92	0.94
MLP-5	TV news	0.86	0.85
MLP-8	income	0.95	0.90
ResNet	MNIST	1.00	1.00
ResNet	CIFAR-10	0.89	0.89
VGG	MNIST	1.00	1.00
VGG	CIFAR-10	0.98	0.97

Table 1: Classification accuracy when using the raw feature and using the top 10 feature components.

$$f^{(k)} = \sum\nolimits_{i\in N_{\text{feature}}} f_i + \bar{f} + \epsilon \qquad (3)$$

where $N_{\text{feature}} = \{1, 2, \dots, r\}$ denotes the indices of top r principal feature components. $f_i = q_i q_i^T (f^{(k)} - \bar{f}) \in \mathbb{R}^m$ represents the i-th principal feature component. $\bar{f} = \sum_{k=1}^K f^{(k)}/K$ denotes the average feature during the learning process. $\epsilon = f^{(k)} - \bar{f} - \sum_{i\in N_{\text{feature}}} f_i$ represents the overall effect of the remaining $m-r$ feature components in $f^{(k)}$.

In this way, if we consider $\bar{f} + \epsilon$ as a constant background, we can regard the r feature components in $f^{(k)}$ as the variables involved in interactions. *I.e.,* each interaction $S \subseteq N_{\text{feature}}$ represents the collaborative relationship between feature components in S. Here, because $f^{(k)}$ can be extracted from any epoch, we ignore the superscript (k). Then, for a subset $L \subseteq N_{\text{feature}}$, f_L represents the masked feature when we mask feature components in $N_{\text{feature}} \setminus L$,[3] *i.e.,*$f_L = \sum_{i\in L} f_i + \sum_{i\in N_{\text{feature}}, i\notin L} b_i + \bar{f}$. We use $b_i \overset{\text{def}}{=}$

[3] Given that the components in ϵ in Eq. (3) are usually very small (see Figure 2), we ignore these components.

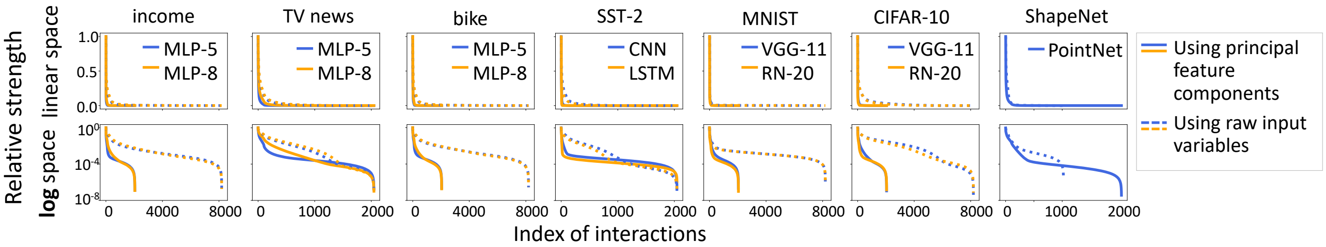


Figure 3: Comparison of the relative strength of interactions on raw input variables and that of interactions on principal feature components. For clarity, AND and OR interactions were put together and sorted in descending order of relative strength. Using principal feature components significantly enhanced the sparsity of interactions.

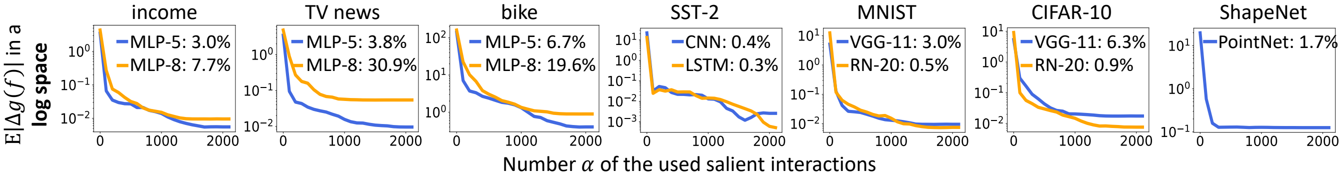


Figure 4: Explanation power of using different numbers of interactions. The monotonically decreasing matching error when we use increasing numbers α of salient interactions to mimic the model output. The numbers on the top-right corner show the average ratio of the minimum number $\mathbb{E}_f[\hat{\alpha}/2^{r+1}]$ of the most salient interactions (*i.e.*, the top $\hat{\alpha}$ interactions) among all $2^{(r+1)}$ interactions that achieve the matching quality $|g(f) - \hat{g}_{\hat{\alpha}}(f)|/|g(f)| \leq 0.1$.

$q_i q_i^T(f|_{\mathbb{E}[x]} - \bar{f})$ to represent the masked state (or namely *the baseline value*) of the i-th feature component. $f|_{\mathbb{E}[x]}$ denotes the feature when the average value $\mathbb{E}[x]$ of all input samples in the training set is fed to the model. b_i represents the i-th feature component in the feature $f|_{\mathbb{E}[x]}$. The mean value over different samples is a widely-used setting for baseline values (Dabkowski and Gal 2017), which alleviates the out-of-the-distribution problem in practice.

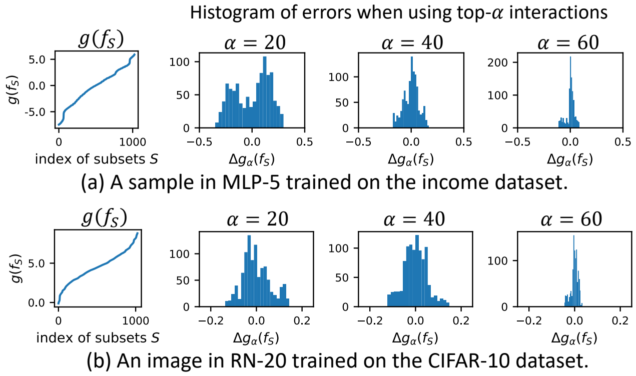
The DNN output $v(x)$ can be regarded as a function of the feature f, *i.e.,* $v(x) = g(f)$, where $g(\cdot)$ denotes subsequent layers upon the feature f. $g(f_L)$ denotes the DNN output on the masked feature. Thus, we can directly use Eq. (2) to compute interactions $I_{\text{and}}(S|f)$ and $I_{\text{or}}(S|f)$ on feature components by replacing $v(x_L)$ with $g(f_L)$.

Computational cost of interactions between feature components. Compared to interactions on raw input variables, interactions on feature components present a much smaller computational cost. For the input $x \in \mathbb{R}^n$, the computational cost of interactions on the n input variables in x is 2^n. When we define interactions on top r feature components ($r \ll n$ in most cases), the computational cost of interactions is reduced to 2^r, which is much less than 2^n.

Experimental settings. We trained a 5-layer MLP (Ren et al. 2023b) (namely *MLP-5*) and an 8-layer MLP (Ren et al. 2023b) (namely *MLP-8*) on three datasets (Dua and Graff 2017), including the census income (namely *income*), TV News channel commercial detection (namely *TV news*), and bike sharing (namely *bike*) datasets. We also followed (Li and Zhang 2023) to train a CNN and a three-layer unidirectional LSTM on the SST-2 dataset (Socher et al. 2013). Besides, we trained VGG-11 (Simonyan and Zisserman 2014) and ResNet-18/20 (He et al. 2016) (namely *RN-18/20*) on the MNIST (LeCun et al. 1998), CIFAR-10 (Krizhevsky 2012), and Tiny ImageNet (Le and Yang 2015) datasets, and trained PointNet (Charles et al. 2017) on the ShapeNet (Yi et al. 2016) dataset. For each neural network, we analyzed features extracted from the (roughly) half depth, which well balanced the informativeness of the feature and the conciseness of the explanation. Please see Appendix F for the detailed experimental settings.

Justification of using principal feature components: how many principal feature components are needed as input variables? We conducted two experiments. In the first experiment, we verified that the used top-ranked feature components represented most signals in f. For each DNN, we fed an input sample x to the DNNs trained after K different epochs, and extracted K feature vectors $f^{(1)}, \ldots, f^{(K)}$ from these DNNs. Using the feature vectors collected from different samples at K different epochs, we conducted PCA to compute eigenvalues in Figure 2. We found that in most DNNs, the top 10 eigenvalues were significantly larger than the rest. The long-tail components with very tiny eigenvalues did not reflect essential signals for the task. Therefore, we set $r = 10$ in all experiments.

In the second experiment, we compared the classification accuracy of using the entire feature f with the classification accuracy of using the top 10 components of the feature $\bar{f} + \sum_{i=1}^{10} f_i$. To this end, we masked other feature components in ϵ to obtain $f' = \sum_{i=1}^{10} f_i + \bar{f}$, according to Eq. (3), and fed f' back to the network for inference. We conducted experiments on four datasets, including the census, commercial, MNIST, and CIFAR-10 datasets. For each dataset, we randomly sample 100 samples and evaluate the classification accuracy of the network based on the original feature



(a) A sample in MLP-5 trained on the income dataset.

(b) An image in RN-20 trained on the CIFAR-10 dataset.

Figure 5: Faithfulness of the explanation. The histograms show the approximation error $\Delta g_\alpha(f_S)$ when using the sum of interactions to match the ground truth $g(f_S)$. Using less than $\alpha = 60$ interactions can well approximate the network outputs on almost masked inputs x_S.

f. Table 1 shows that using the top 10 feature components did not significantly change the classification accuracy. In other words, the top 10 feature components had already represented most of the knowledge learned by the model.

Sparsity of Interactions Theorem 2.1 shows that the network output on an input sample can always be explained by a small set of interactions, no matter how we randomly mask the input sample. Then the principle of Occam's Razor suggests that we can consider such interactions as primitive inference patterns encoded by the DNN. However, the proof of the sparsity (Ren et al. 2024) of interaction is conducted under three assumed common conditions[1], which are difficult to examine in real DNNs. Besides, unlike (Ren et al. 2024), we use OR interactions. Therefore, we need to verify the sparsity of interactions on feature components.

We compared the sparsity of interactions on feature components with the sparsity of interactions on raw input variables. To extract interactions on raw input variables, we followed Ren et al. (2023b) to divide each input image in the MNIST and CIFAR-10 datasets into 7 × 7 and 8 × 8 patches, respectively. Then, we randomly sampled twelve image patches as input variables to compute interactions. For the ShapeNet dataset, we took the manually annotated parts provided by Li and Zhang (2023) as input variables. To compute interactions on feature components, we followed Appendix F to extract principal feature components. For simplicity, we concatenated strength $|I_{\text{and}}(S|x)|$ of 2^r AND interactions and strength $|I_{\text{or}}(S|x)|$ of 2^r OR interactions to construct a 2^{r+1}-dimensional vector $\boldsymbol{I}$. The strength was further normalized by $\boldsymbol{I} \leftarrow \boldsymbol{I} / \max_i \boldsymbol{I}_i$. Figure 3 shows the curve of relative interaction strength sorted in descending order, which was averaged over different input samples. **Using principal feature components could significantly enhance the sparsity of interactions.**

Examining Faithfulness of Interactions In this section, we conducted two experiments to use interactions to mimic the entire model output $g(f)$, so as to evaluate the faithfulness of the interaction-based explanation. In the first experiment, we measured the matching error when we used salient interactions to match the model output. We followed Appendix F to extract AND-OR interactions. Let Ω_α denote the set of α salient interactions with the highest values of $|I_{\text{and/or}}(S|f)|$. We computed the matching error $\mathbb{E}_x|\Delta g(f)| = \mathbb{E}_f[|g(f) - \hat{g}_\alpha(f)|]$, *w.r.t.* $\hat{g}_\alpha(f) = g(f_\emptyset) + \sum_{S\in\Omega_\alpha} I_{\text{and}}(S|f) + \sum_{S\in\Omega_\alpha} I_{\text{or}}(S|f)$. We used different numbers α of salient interactions to compute the corresponding matching errors. Furthermore, we computed the least number $\hat{\alpha}$ of interactions that were required to cover 90% of the network output $g(f)$, *i.e.*, $\hat{\alpha} = \min \alpha$ s.t. $(|g(f) - \hat{g}_\alpha(f)|)/|g(f)| \leq 0.1$. Figure 4 reports the average matching error over different samples and the average ratio of the minimum interaction number ($E_f[\hat{\alpha}/2^{r+1}]$). The network outputs were usually well matched by only using less than 10% salient interactions.

The second experiment demonstrated that the sum of a few interactions could well approximate various network outputs on randomly masked features $\{g(f_S)\}_S$. Specifically, we used different numbers ($\alpha \in \{20, 40, 60\}$) of salient interactions to approximate the model outputs on 2^n masked features of an input sample. Then, for each masked feature f_S, we computed $\Delta g_\alpha(f_S) = g(f_S) - \hat{g}_\alpha(f_S)$ as the approximation error on f_S, where $\hat{g}_\alpha(f_S) = g(f_\emptyset) + \sum_{L\in\Omega_\alpha, \emptyset\neq L\subseteq S} I_{\text{and}}(L|f) + \sum_{L\in\Omega_\alpha, L\cap S\neq\emptyset} I_{\text{or}}(L|f)$. Figure 5 shows network outputs on all 2^n masked features of an input sample in ascending order and the approximation errors. For visualization, we averaged the approximation error over 50 neighboring masked features for smoothing. The results show that a small number (usually less than 60) of interactions could well approximate the varying network outputs on different masked features.

3 Emergence of Primitive Interactions

Emergence of Interactions During Training

Five types of interactions. In this section, we analyze a DNN's learning efficiency based on its learning dynamics of interactions during the learning process.

For an interaction pattern S, let $\nabla_t I(S|x, \theta_t) = \frac{\partial I(S|x,\theta_t)}{\partial t}$ denote the slope of the interaction curve at the t epoch. We observe the phenomena *w.r.t.* the values of $I(S|x, \theta_t)$ and $\nabla_t I(S|x, \theta_t)$ in Table 2, and categorize them into five groups. Specifically, we categorize the curves of numerical utilities of different salient interactions S into the following five types, which *reflect distinctive behaviors of a DNN learning different types of primitive inference patterns.* Please see Figure 6 for the curve of $I_{\text{and/or}}(S|x)$ across different epochs for each salient interaction S.

(a) Figure 6 (a) shows interactions belonging to the first group. The strength of these interactions increases throughout the learning process in a relatively stable manner. Thus, we consider such interactions to be stably learned by the DNN, and we call them *reliable interactions*.

(b) In the second group, utilities of interactions in Figure 6 (b) are usually close to zero in the beginning. Then, the strength of their utility first increases and then decreases, sometimes decreasing to almost zero. These interactions are referred to as *withdrawn interactions*.

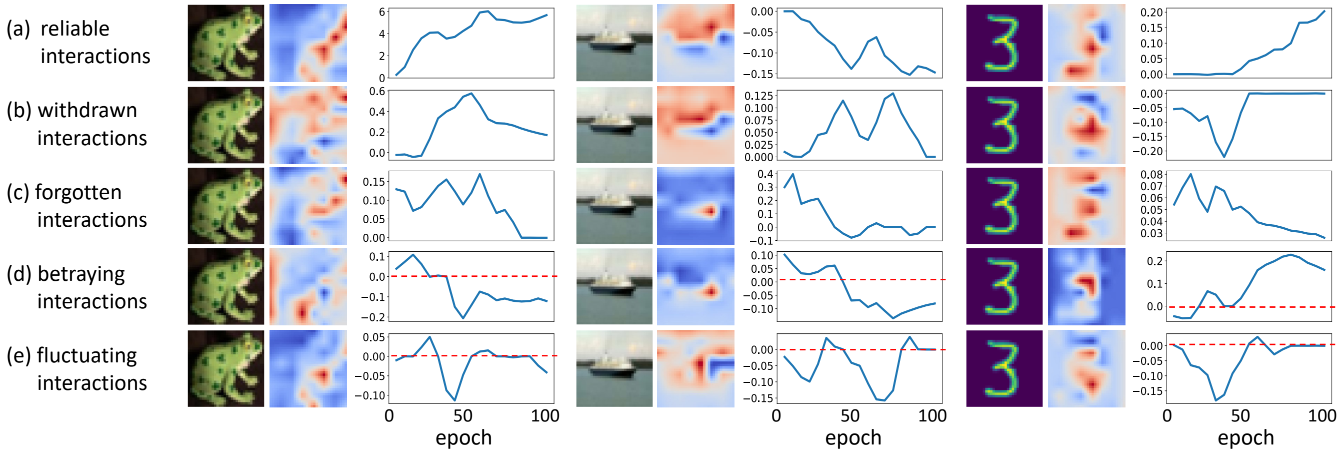


Figure 6: Curves of the utility of interactions during the learning of DNNs. These interactions can be categorized into five groups. Please refer to Appendix I for results on more samples.

Group	Phenomenon
reliable	$\forall t \in T, I(S \mid x, \theta_t) \cdot \nabla_t I(S \mid x, \theta_t) \geq 0$
withdrawn	$\exists t_{mid}$ s.t. when $t < t_{mid}$, $I(S \mid x, \theta_t) \cdot \nabla_t I(S \mid x, \theta_t) \geq 0$ when $t > t_{mid}$, $I(S \mid x, \theta_t) \cdot \nabla_t I(S \mid x, \theta_t) \leq 0$
forgotten	$\forall t \in T, I(S \mid x, \theta_t) \cdot \nabla_t I(S \mid x, \theta_t) \leq 0$
betraying	$\exists t_{mid}$ s.t. $\forall t_1 < t_{mid}, \forall t_2 > t_{mid}, I(S \mid x, \theta_{t_1}) \cdot I(S \mid x, \theta_{t_2}) \leq 0$
fluctuating	$I(S \mid x, \theta_t)$ and $\nabla_t I(S \mid x, \theta_t)$ oscillate around zero

Table 2: Categorization of five groups of interactions.

(c) As Figure 6 (c) shows, the initial utility of interactions in the third group is non-ignorable. However, the strength of these interactions keeps decreasing to zero. These interactions are gradually forgotten by the DNN. We call them *forgotten interactions*.
(d) Figure 6 (d) shows interactions in the fourth group. The interactions experience a gradual shift towards an interaction utility that is opposite to their initial utility. These interactions are called *betraying interactions*.
(e) For interactions in the fifth group, Figure 6 (e) has fluctuating interactive utilities throughout the learning process, thereby being called *fluctuating interactions*.

Different types of interactions reflect primitive inference patterns of different learning efficiency. (1) We can consider *reliable interactions* and *forgotten interactions* as stably and efficiently learned knowledge. (2) Some *Betraying interactions* and *withdrawn interactions* reflect the trial-and-error process during learning, while some are caused by a bad initialization of weights. (3) *Fluctuating interactions* correspond to the noise knowledge.

In particular, the goal of (Shwartz-Ziv and Tishby 2017) is quite similar to ours,*i.e.*, understanding the learning and forgetting of information throughout the training of a DNN. Shwartz-Ziv and Tishby (2017) discovers that the DNN usually first extracts information and then compresses information. In our study, we discover the existence of withdrawn interactions, which precisely explains what information is first learned and subsequently forgotten.

In addition, we have conducted an experiment on a toy dataset to demonstrate that interactions can successfully reveal betraying features learned during training of the DNN. Please refer to Appendix I for the experimental results.

The number and complexity order of interactions in each group help to understand the performance of DNNs. For each DNN and each sample, we selected 100 interactions whose maximum interaction strength $(\max_t |I_{\text{and}}(S|x,\theta_t)|$ and $\max_t |I_{\text{or}}(S|x,\theta_t)|)$ throughout the training process were ranked in top 100 among all interactions. Then, we counted the number of interactions belonging to each group among these 100 salient interactions. Table 3 reports the average number of interactions in each group over different samples. We found that compared to VGG-11, RN-20 learned more reliable and forgotten interactions, while having fewer betraying and fluctuating interactions. This might be because the residual connections in RN-20 made the features more stable. Besides, we also noticed that the DNNs trained on the MNIST dataset usually encoded more reliable interactions and less betraying and fluctuating interactions than the DNNs trained on the CIFAR-10 and Tiny ImageNet datasets. This result indicated that the dynamics of interactions also provided a new perspective to analyze the difficulty of training a DNN on a dataset.

We further studied the complexity (order) of interactions. Let the order of an interaction S be referred to as the number of variables in S, order$(S) = |S|$. Zhou et al. (2024) have found that *compared to high-order interactions, low-order interactions extracted from training samples are more likely to generalize to (appear in) testing samples.* Since the network output is the sum of all interactions, we can use the ratio of low-order interactions and high-order interactions to explain the generalization power of the DNN. Thus, We explored the order of interactions in each group. Figure 7 reports the average number of interactions of each order over different samples in each group. We found that the distribution of interactions over different orders was similar in

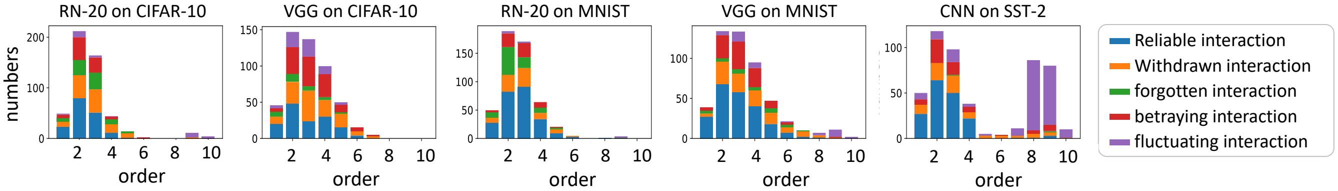


Figure 7: The number and complexity order of interactions of each order in each group.

Model	Dataset	reliable interactions	withdrawn interactions	forgotten interactions	betraying interactions	fluctuating interactions
VGG-11	CIFAR-10	28.4	**26.4**	6.0	**26.6**	**12.6**
RN-20	CIFAR-10	**33.4**	26.2	**16.6**	17.8	6.0
VGG-11	MNIST	44.2	**21.4**	5.8	**20.8**	**7.8**
RN-20	MNIST	**49.6**	18.2	**18.0**	12.6	1.6
RN-18	Tiny ImageNet	4.0	39.0	20.4	20.4	16.2
CNN	SST-2	33.6	14.4	0.4	12.8	38.8

Table 3: Average number of salient interactions within each group.

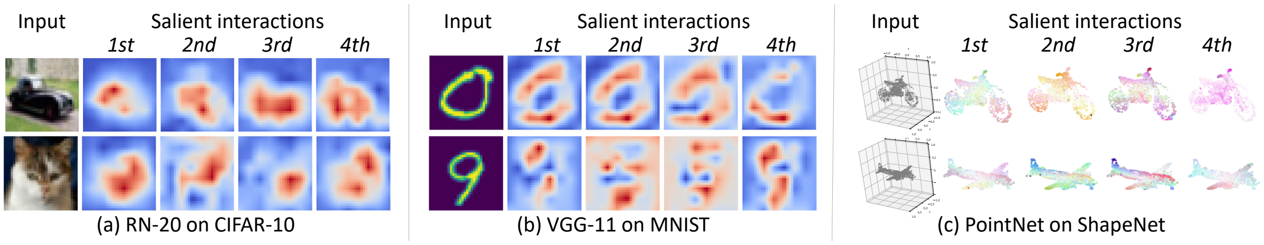


Figure 8: Visualization of salient interactions. Please see Appendix H for more results.

different models. Besides, we found that high-order interactions were usually fluctuating and withdrawn interactions, because high-order interactions usually represented complex and unstable features.

In addition, Appendix J showed that DNNs tended to use high-order interactions to classify abnormal samples (*e.g.*, samples with noisy labels) than normal samples.

What Does an Interaction Represent?

As a supplement to the mathematical explanation of the learning dynamics, we also visualize the primitive interactions in this subsection, although interactions toward mathematically concise explanation are not equivalent to semantically meaningful concepts.

We first visualize the attribution map of each top-ranked feature component f_i. Considering the distinctive properties of different tasks, we apply the projected influence attribution, the gradient-based attribution (Simonyan and Zisserman 2014), and the Shapley value (Shapley 1953) to estimate the attribution of input data (image data, the 3D point cloud data, and the language data) to each feature component f_i, respectively. Figure **??** shows examples of attribution maps of feature components, where the red color indicates that the corresponding regions in the input have a positive attribution to the principal feature component, while the blue color indicates a negative attribution. For the point cloud data, we use RGB color channels to visualize the three-dimensional attributions. Please see Appendix H for details of the visualization techniques and results.

Then, for each interaction $S \subseteq N_{\text{feature}}$ with a considerable utility $I_{\text{and/or}}(S|x)$, Figure 8 visualizes the attribution map of the interaction, which simply sums up the attribution maps of its compositional feature components $\{f_i\}_{i\in S}$.

4 Conclusion and Discussions

In this study, we have proposed a method to simplify and summarize a DNN's highly complex learning dynamics into the change of a few interaction primitives. We have extended the interaction defined on raw input variables by (Ren et al. 2023a; Li and Zhang 2023; Zhou et al. 2023), and have newly defined interactions on principal feature components. This extension greatly boosts the sparsity/simplicity of the interaction-based explanation of a DNN, which provides a new perspective to understand mechanical factors for learning efficiency. The mathematical faithfulness of the new interaction is experimentally verified. We have found that the dynamics of all salient interactions naturally belong to five groups, *i.e.,* reliable, withdrawn, forgotten, betraying, and fluctuating interactions, which provide new insights, *e.g.,* explaining how reliable inference patterns are gradually learned, how redundant patterns are first learned and later discarded, and how a DNN learns noisy patterns.

Acknowledgments

This work is partially supported by the National Science and Technology Major Project (2021ZD0111602), the National Nature Science Foundation of China (92370115, 62276165). Jie Ren is supported by Wu Wen Jun Honorary Doctoral Scholarship, AI Institute, Shanghai Jiao Tong University.

References

Ancona, M.; Öztireli, C.; and Gross, M. H. 2019. Explaining Deep Neural Networks with a Polynomial Time Algorithm for Shapley Value Approximation. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning, ICML*, volume 97 of *Proceedings of Machine Learning Research*, 272–281. PMLR.

Charles, R. Q.; Su, H.; Kaichun, M.; and Guibas, L. J. 2017. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 77–85.

Chen, L.; Lou, S.; Huang, B.; and Zhang, Q. 2024. Defining and Extracting generalizable interaction primitives from DNNs. *CoRR*, abs/2401.16318.

Cheng, X.; Cheng, L.; Peng, Z.; Xu, Y.; Han, T.; and Zhang, Q. 2024. Layerwise Change of Knowledge in Neural Networks. In *Forty-first International Conference on Machine Learning*.

Covert, I.; Lundberg, S. M.; and Lee, S. 2020. Understanding Global Feature Contributions Through Additive Importance Measures. *CoRR*, abs/2004.00668.

Dabkowski, P.; and Gal, Y. 2017. Real Time Image Saliency for Black Box Classifiers. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 6967–6976.

Deng, H.; Ren, Q.; Zhang, H.; and Zhang, Q. 2022. Discovering and Explaining the Representation Bottleneck of DNNS. In *The Tenth International Conference on Learning Representations, ICLR*.

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository.

Elhage, N.; Nanda, N.; Olsson, C.; Henighan, T.; Joseph, N.; Mann, B.; Askell, A.; Bai, Y.; Chen, A.; Conerly, T.; DasSarma, N.; Drain, D.; Ganguli, D.; Hatfield-Dodds, Z.; Hernandez, D.; Jones, A.; Kernion, J.; Lovitt, L.; Ndousse, K.; Amodei, D.; Brown, T.; Clark, J.; Kaplan, J.; McCandlish, S.; and Olah, C. 2021. A Mathematical Framework for Transformer Circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Fel, T.; Picard, A.; Bethune, L.; Boissin, T.; Vigouroux, D.; Colin, J.; Cadénc, R.; and Serre, T. 2023. CRAFT: Concept Recursive Activation FacTorization for Explainability. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2711–2721.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Krizhevsky, A. 2012. Learning Multiple Layers of Features from Tiny Images. *University of Toronto*.

Le, Y.; and Yang, X. S. 2015. Tiny ImageNet Visual Recognition Challenge.

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

Li, M.; and Zhang, Q. 2023. Does a Neural Network Really Encode Symbolic Concept? In *Proceedings of the 36th International Conference on Machine Learning, ICML*, Proceedings of Machine Learning Research. PMLR.

Meng, K.; Bau, D.; Andonian, A.; and Belinkov, Y. 2022. Locating and Editing Factual Associations in GPT. In *NeurIPS*.

Olsson, C.; Elhage, N.; Nanda, N.; Joseph, N.; DasSarma, N.; Henighan, T.; Mann, B.; Askell, A.; Bai, Y.; Chen, A.; Conerly, T.; Drain, D.; Ganguli, D.; Hatfield-Dodds, Z.; Hernandez, D.; Johnston, S.; Jones, A.; Kernion, J.; Lovitt, L.; Ndousse, K.; Amodei, D.; Brown, T.; Clark, J.; Kaplan, J.; McCandlish, S.; and Olah, C. 2022. In-context Learning and Induction Heads. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Park, H.; Lee, S.; Hoover, B.; Wright, A. P.; Shaikh, O.; Duggal, R.; Das, N.; Hoffman, J.; and Chau, D. H. 2022. ConceptEvo: Interpreting Concept Evolution in Deep Learning Training. *CoRR*, abs/2203.16475.

Ren, J.; Li, M.; Chen, Q.; Deng, H.; and Zhang, Q. 2023a. Defining and Quantifying the Emergence of Sparse Concepts in DNNs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, https://arxiv.org/pdf/2111.06206v5.pdf.

Ren, Q.; Deng, H.; Chen, Y.; Lou, S.; and Zhang, Q. 2023b. Bayesian neural networks tend to ignore complex and sensitive concepts. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, Proceedings of Machine Learning Research. PMLR.

Ren, Q.; Gao, J.; Shen, W.; and Zhang, Q. 2024. Where We Have Arrived in Proving the Emergence of Sparse Interaction Primitives in AI Models. In *The Twelfth International Conference on Learning Representations*.

Shapley, L. S. 1953. *17. A Value for n-Person Games*, 307–318. Princeton: Princeton University Press. ISBN 9781400881970.

Shwartz-Ziv, R.; and Tishby, N. 2017. Opening the Black Box of Deep Neural Networks via Information. *CoRR*, abs/1703.00810.

Simonyan, K.; and Zisserman, A. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv e-prints*, arXiv:1409.1556.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A.; and Potts, C. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.

Yi, L.; Kim, V. G.; Ceylan, D.; Shen, I.-C.; Yan, M.; Su, H.; Lu, C.; Huang, Q.; Sheffer, A.; and Guibas, L. 2016. A Scalable Active Framework for Region Annotation in 3D Shape Collections. *SIGGRAPH Asia*.

Zhao, Z.; Xu, P.; Scheidegger, C.; and Ren, L. 2022. Human-in-the-loop Extraction of Interpretable Concepts in Deep Learning Models. *IEEE Trans. Vis. Comput. Graph.*, 28(1): 780–790.

Zhou, H.; Tang, H.; Li, M.; Zhang, H.; Liu, Z.; and Zhang, Q. 2023. Explaining How a Neural Network Play the Go Game and Let People Learn. *arXiv e-prints*, arXiv:2310.09838.

Zhou, H.; Zhang, H.; Deng, H.; Liu, D.; Shen, W.; Chan, S.; and Zhang, Q. 2024. Explaining Generalization Power of a DNN Using Interactive Concepts. In *AAAI*, 17105–17113. AAAI Press.