Bayesian Neural Networks Avoid Encoding Complex and Perturbation-Sensitive Concepts

Qihan Ren [*1] **Huiqi Deng** [*1] **Yunuo Chen** [1] **Siyu Lou** [1] **Quanshi Zhang** [1 2]

Abstract

In this paper, we focus on mean-field variational Bayesian Neural Networks (BNNs) and explore the representation capacity of such BNNs by investigating which types of concepts are less likely to be encoded by the BNN. It has been observed and studied that a relatively small set of interactive concepts usually emerge in the knowledge representation of a sufficiently-trained neural network, and such concepts can faithfully explain the network output. Based on this, our study proves that compared to standard deep neural networks (DNNs), it is less likely for BNNs to encode complex concepts. Experiments verify our theoretical proofs. Note that the tendency to encode less complex concepts does not necessarily imply weak representation power, considering that complex concepts exhibit low generalization power and high adversarial vulnerability. The code is available at `https://github.com/sjtu-xai-lab/BNN-concepts`.

1. Introduction

Unlike standard deep neural networks (DNNs), Bayesian neural networks (BNNs) represent network weights as probability distributions. Therefore, BNNs exhibit distinctive representation capacities from standard DNNs. Existing studies (Blundell et al., 2015; Gal & Smith, 2018; Kristiadi et al., 2020; Carbone et al., 2020; Wenzel et al., 2020; Krishnan et al., 2020; Zhang et al., 2022) usually analyzed BNNs in terms of generalization power, adversarial robustness, and optimization.

In contrast to the above studies, this paper proposes a new perspective to investigate the representation capacity of BNNs, *i.e.*, we discover and theoretically prove that BNNs are less likely to encode complex and perturbation-sensitive concepts than standard DNNs. In fact, such a property brings specific advantages to feature representations of BNNs. To be precise, we limit our study to the scope of **mean-field variational BNNs** (Blundell et al., 2015), which is one of the most commonly used BNNs. Thus, in this paper, we just use the term *BNN* to refer to mean-field variational BNNs.

Common phenomenon of concept emergence in various neural networks. Although it is well-known that a neural network does not explicitly encode concepts like graphical models, recent studies have discovered (Ren et al., 2023a; Li & Zhang, 2023) and theoretically proved (Ren et al., 2023c) a common concept-emerging phenomenon that neural networks usually *implicitly* encode a small number of interactive concepts for inference, which have been observed in different neural networks for various tasks. Specifically, each interactive concept represents an AND relationship among a set of input variables.

For example, we can use $I(S = \{\text{eyes, nose, mouth}\}) = U_S \cdot \text{exist(eyes)} \cdot \text{exist(nose)} \cdot \text{exist(mouth)}$ to illustrate the AND relationship for the face concept in image classification. If any image patch in the set $S = \{\text{eyes, nose, mouth}\}$ is masked, then the face concept will be deactivated, and the numerical effect of this concept is removed ($I(S) = 0$) and no longer influences the network output.

More importantly, interactive concepts can be considered as faithful inference patterns encoded by the neural network. It is because Ren et al. (2023a) has proved that people can use a relatively small number of interactive concepts to well mimic the inference logic of the neural network on a certain input sample. That is, numerical effects of these concepts always well predict diverse network outputs, no matter how the input sample is masked.

BNNs ignore complex and perturbation-sensitive concepts. Based on the interactive concepts, we discover and theoretically prove that *compared to standard DNNs, it is more difficult for a neural network to encode complex interactive concepts, as long as it has weight uncertainty.* The

[*]Equal contribution [1]Shanghai Jiao Tong University, Shanghai, China [2]Quanshi Zhang is the corresponding author. He is with the Department of Computer Science and Engineering, the John Hopcroft Center, at the Shanghai Jiao Tong University, China. Correspondence to: Quanshi Zhang <zqs1022@sjtu.edu.cn>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

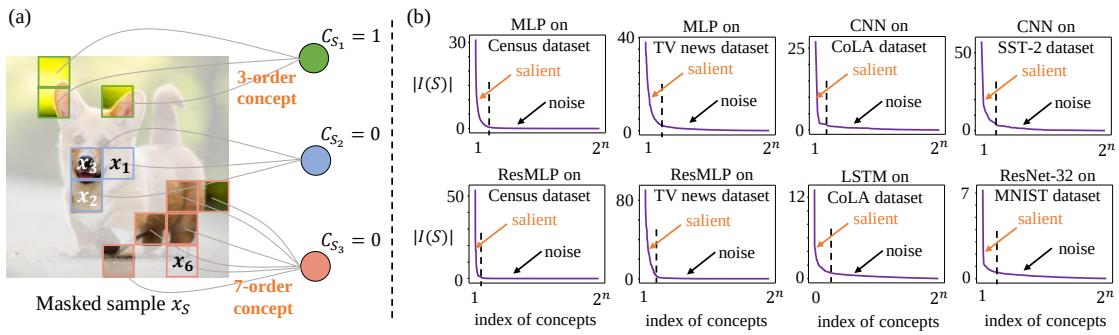

Figure 1. (a) Illustration of interactive concepts encoded by a neural network. Each interactive concept S corresponds to an AND relationship among a specific set S of input variables (image patches). C_S represents the activation state of the concept S. The patch x_1 is masked ($\text{unmask}(x_1) = 0$), so the concept S_2 is deactivated, *i.e.*, $C_{S_2} = \bigwedge_{i=1}^{3} \text{unmask}(x_i) = 0$. Similarly, x_6 is masked, so that the concept S_3 is deactivated, and $C_{S_3} = 0$. (b) Experiments demonstrate the common concept-emerging phenomenon. Neural networks with various architectures all encode sparse interactive concepts. In other words, most interactive concepts have near-zero effects, *i.e.*, $I(S) \approx 0$, and can be considered as noises; only a relatively small number of interactive concepts have significant effects. For better visualization, the interactive concepts are sorted by strengths in descending order.

complexity of an interactive concept S is defined as the number of variables in the set S, *i.e.*, $\text{complexity}(S) = |S|$. Here, $|S|$ is also termed the *order* of the interactive concept.

We prove the above conclusion through three steps. First, it is difficult to theoretically analyze interactive concepts encoded by BNNs, because BNNs represent network weights as probability distributions. To this end, we find that we can usually use a *surrogate DNN model*, which is constructed by adding perturbations to both the input and low-layer features of a standard DNN, to approximate feature representations of a BNN. In this way, we can directly analyze the surrogate DNN model with feature uncertainty, instead of investigating the BNN with weight uncertainty.

Second, we prove that in the surrogate DNN model, high-order interactive concepts are more sensitive to random perturbations than low-order interactive concepts.

Third, we prove that the sensitivity makes high-order interactive concepts difficult to be learned when features are perturbed. In this way, we can conclude that high-order interactive concepts are also less likely to be learned by the BNN when its weights are perturbed.

In addition, experiments showed that the strength of high-order (complex) interactive concepts encoded by BNNs was weaker than those encoded by standard DNNs, which verified the above theoretical conclusion.

Note that our proof does NOT mean that a BNN has limited representation capacity. Instead, we just demonstrate the distinctive tendency of avoiding encoding complex (high-order) interactive concepts, when weight uncertainty is introduced into the neural network. This does not mean that BNNs have weaker representation power than standard DNNs. If the task loss requires to encode complex concepts, then our research indicates that the BNN must reduce its weight uncertainty, to some extent.

Practical values and advantages of avoiding encoding complex concepts. Although we prove that BNNs tend to avoid encoding complex concepts, it is not necessarily a disadvantage of the BNN, compared to standard DNNs. On the contrary, it has been found that compared to simple (low-order) concepts, complex (high-order) concepts encoded by a neural network usually have poorer generalization ability (Lengerich et al., 2022) and are more vulnerable to adversarial attacks (Ren et al., 2021). Thus, encoding less complex concepts might be an advantage.

2. BNNs ignore complex and perturbation-sensitive concepts

Unlike standard DNNs, a BNN represents each weight in the network as a probability distribution, instead of a scalar. In this paper, we limit the scope of our study to mean-field variational BNNs (Blundell et al., 2015), where all weights $\boldsymbol{W}$ are formulated as a Gaussian distribution $\mathcal{N}(\boldsymbol{W}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$, and the covariance matrix $\boldsymbol{\Sigma}$ is diagonal. Other types of BNNs (*e.g.*, BNNs based on the Monte Carlo Dropout (Gal & Ghahramani, 2016)) are not discussed. The BNN learns parameters $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and we use $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ to represent the weight distribution. Let us consider a classification task with the training data $\mathcal{D} = \{(\boldsymbol{x}^{(1)}, y^{(1)}), \ldots, (\boldsymbol{x}^{(n)}, y^{(n)})\}$. Training a BNN is to minimize the Kullback-Leibler (KL) divergence between the distribution $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ and the posterior distribution $p(\boldsymbol{W}|\mathcal{D})$.

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg\min_{\boldsymbol{\theta}} \text{KL}[q_{\boldsymbol{\theta}}(\boldsymbol{W}) \| p(\boldsymbol{W}|\mathcal{D})] \\ &= \arg\min_{\boldsymbol{\theta}} -\mathbb{E}_{\boldsymbol{W}\sim q_{\boldsymbol{\theta}}(\boldsymbol{W})}[\log p(\mathcal{D}|\boldsymbol{W})] + \text{KL}[q_{\boldsymbol{\theta}}(\boldsymbol{W}) \| p(\boldsymbol{W})],\end{aligned} \tag{1}$$

where the first term is the classification loss, and the second term is the KL divergence between $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ and the prior distribution $p(\boldsymbol{W})$, which is usually formulated as a Gaussian distribution $\mathcal{N}(\boldsymbol{W}; \mathbf{0}, \boldsymbol{I})$. In addition, given a testing sample $\boldsymbol{x}$, the inference of the BNN is conducted as follows. First, network weights are sampled from the weight distribution $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ to construct multiple neural networks. Then, each network is used to conduct inference on the sample $\boldsymbol{x}$, and the final inference result $p(y|\boldsymbol{x})$ is computed as the average classification probability of all the networks,

$$p(y|\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{W}\sim q_{\boldsymbol{\theta}}(\boldsymbol{W})}[p(y|\boldsymbol{x}, \boldsymbol{W})]. \tag{2}$$

2.1. Preliminaries: emergence of sparse concepts

The learning of neural networks is usually regarded as a fitting problem between the ground-truth label and the model prediction, without explicit learning of specific concepts. However, recent studies have empirically discovered (Ren et al., 2023a; Li & Zhang, 2023) and theoretically proved (Ren et al., 2023c) that sparse AND relationships between input variables were usually implicitly encoded by a neural network when it was sufficiently trained. As shown in Figure 1(a), these AND relationships can be viewed as specific types of *interactive concepts*, which will be introduced in the **interactive concepts** paragraph.

Although counter-intuitive, this concept-emerging phenomenon does exist in various neural networks. Furthermore, such interactive concepts have been used to prove the representation bottleneck of the neural network (Deng et al., 2022) and obtain optimal masking states for attribution methods (Ren et al., 2023b). **We also verify the trustworthiness of using interactive concepts to explain neural networks in experiments (see the end of this section).**

Interactive concepts. Ren et al. (2021) proposed the interaction effect $I(S)$ to study the emergence of concepts. Let us consider a pre-trained neural network v and an input sample $\boldsymbol{x} = [x_1, \ldots, x_n]$ with n input variables indexed by $N = \{1, \ldots, n\}$. Let Ω denote a set of interactive concepts extracted from the network. Each interactive concept $S \in \Omega$ corresponds to the collaboration (AND relationship) between input variables in a specific set $S \subseteq N$, thus $\Omega \subseteq 2^N = \{S|S \subseteq N\}$. For instance, as Figure 1(a) shows, a concept $S = \{x_1, x_2, x_3\}$ is formed due to the co-occurrence of the three image patches. The concept will be activated and make a certain interaction effect $I(S)$ on the network output, only if the patches x_1, x_2, x_3 are all present. In contrast, the absence (masking) of any patch among x_1, x_2, and x_3 will deactivate the concept and remove the interaction effect, *i.e.*, $I(S|\boldsymbol{x}^{\text{mask}}) = 0$.

Specifically, the interaction effect $I(S|\boldsymbol{x})$ on the sample $\boldsymbol{x}$ is computed by the Harsanyi dividend (Harsanyi, 1963).

$$I(S|\boldsymbol{x}) = \sum\nolimits_{T\subseteq S}(-1)^{|S|-|T|} \cdot v(\boldsymbol{x}_T). \tag{3}$$

If $I(S|\boldsymbol{x})$ has a significant value, then the neural network is considered to encode an interactive concept S; otherwise, if $I(S|\boldsymbol{x}) \approx 0$, the concept S does not exist. Here, $\boldsymbol{x}_T$ denotes the masked input sample, where variables in $N \setminus T$ are masked and variables in T are kept unchanged. Besides, $v(\boldsymbol{x}_T) \in \mathbb{R}$ can be computed as a scalar output of the neural network on the masked sample $\boldsymbol{x}_T$ (*e.g.*, the confidence score of classifying the input sample $\boldsymbol{x}_T$ to the ground-truth category $v(\boldsymbol{x}_T) = \log \frac{p(y=y_{\text{truth}}|\boldsymbol{x}_T)}{1-p(y=y_{\text{truth}}|\boldsymbol{x}_T)}$).

Faithfulness of interactive concepts. Given an input sample $\boldsymbol{x}$ with n variables, we have 2^n different ways to mask the sample $\boldsymbol{x}$ and obtain the masked sample $\boldsymbol{x}_T$ *w.r.t.* all subsets $T \subseteq N$. To this end, Ren et al. (2021) proved that

$$\exists\, \Omega \subseteq 2^N,\ s.t.\ \forall\, T \subseteq N, v(\boldsymbol{x}_T) = \sum\nolimits_{S\in\Omega, S\subseteq T} I(S|\boldsymbol{x}), \tag{4}$$

where $2^N = \{S|S \subseteq N\}$. The equation indicates that interactive concepts in Ω can well mimic network outputs on all the 2^n masked samples. Thus, we can consider that all interactive concepts in the set Ω as faithful inference patterns encoded by the neural network.

Sparsity of interactive concepts. More crucially, extensive experiments (Ren et al., 2023a; Li & Zhang, 2023) discovered that interactive concepts emerging in a neural network are usually very sparse. Figure 1(b) shows that most interactive concepts have near-zero interaction effects ($|I(S|\boldsymbol{x})| \approx 0$), thus having negligible influence on the network output. Only a few salient interactive concepts have significant effects $I(S|\boldsymbol{x})$ on the network output. In this way, the network output can be mimicked by only a few salient interactive concepts in Ω_{salient}.

$$\forall\, T \subseteq N,\ \ v(\boldsymbol{x}_T) = \sum\nolimits_{S\in\Omega_{\text{salient}}} I(S|\boldsymbol{x}) + \epsilon \tag{5}$$

The above equation decomposes the output $v(\boldsymbol{x}_T)$ into two parts: (1) effects of all salient interactive concepts in Ω_{salient}, and (2) a small residual term ϵ containing negligible effects of all non-salient interactive concepts.

Empirically verifying the sparsity of concepts. Based on Eq. (5), in the following analysis, *only salient interactive concepts in $\Omega_{salient}$ are regarded as valid concepts encoded by a neural network.* We empirically verify the emergence of sparse concepts in various neural networks, including multi-layer perceptrons (MLPs), residual multi-layer perceptrons (ResMLPs) (Touvron et al., 2022), long short-term memory (LSTM) (Hochreiter & Schmidhuber, 1997), and convolutional neural networks (CNNs), and on different datasets, including tabular data (Census dataset and TV news dataset (Dua & Graff, 2017)), language data (CoLA (Warstadt et al., 2019) and SST-2 (Socher et al., 2013)), and image data (MNIST (LeCun et al., 1998)). Figure 1(b) verifies that concepts encoded by various neural networks are all sparse.

Complexity of a neural network representing a concept. In many previous studies (Deng et al., 2022; Wang et al., 2021; Zhang et al., 2021), the complexity of an interactive concept S was measured by the number of variables in the set S (also termed the *order* of the interactive concept), *i.e.*, $\text{complexity}(S) = \text{order}(S) = |S|$. Then, a low-order concept represents a simple collaboration among a few input variables, while a high-order concept represents a complex collaboration among many input variables.

2.2. Approximating weight uncertainty by adding input perturbations

In this paper, we aim to prove that compared to standard DNNs, it is more difficult to encode high-order (complex) interactive concepts as long as the network has weight uncertainty. Note that previous studies (Lengerich et al., 2022; Ren et al., 2021) found that a DNN encoding less complex concepts was **NOT** necessarily equivalent to a weak representation capacity. Instead, it usually boosts the generalization power and adversarial robustness. In addition, as discussed in the last two paragraphs of the introduction, the BNN can still encode complex concepts when it learns small variances.

Unlike standard DNNs, a BNN formulates each weight as a probability distribution, which boosts the difficulty of theoretically analyzing interactive concepts encoded in a BNN. Therefore, in this subsection, we first discover that introducing uncertainty to weights in the BNN can be approximated by adding perturbations to input variables and low-layer features in experiments. In other words, we add random perturbations to both input variables and low-layer features of a standard DNN, and we demonstrate that such a perturbed DNN performs as a *surrogate DNN model*, which well approximates feature representations of a BNN.

Let us consider a feed-forward BNN, which has L cascaded linear layers and ReLU layers. Given an input sample $\boldsymbol{x} \in \mathbb{R}^{D_0}$ ($D_0 = n$), the feature of the l-th layer $\boldsymbol{h}^{(l)} \in \mathbb{R}^{D_l}$ ($1 \le l \le L$) is computed as follows.

$$\boldsymbol{h}^{(l)} = \boldsymbol{W}^{(l)}(\cdots \boldsymbol{\Phi}^{(1)}(\boldsymbol{W}^{(1)}\boldsymbol{x} + \boldsymbol{b}^{(1)})\cdots) + \boldsymbol{b}^{(l)}, \quad (6)$$

where $\boldsymbol{W}^{(l)} \in \mathbb{R}^{D_l \times D_{l-1}}$ and $\boldsymbol{b}^{(l)} \in \mathbb{R}^{D_l}$ denote the weight matrix and bias of the l-th linear layer, respectively. In the BNN, $W_{ij}^{(l)} \sim \mathcal{N}(\overline{W}_{ij}^{(l)}, (\sigma_{ij}^{(l)})^2)$ is independently sampled from Gaussian distributions. We use $\boldsymbol{\mu}_{\boldsymbol{W}^{(l)}} = [\overline{W}_{ij}^{(l)}] \in \mathbb{R}^{D_l \times D_{l-1}}$ to denote the mean of the weight matrix. Besides, $\boldsymbol{b}^{(l)} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{b}^{(l)}}, \boldsymbol{\Sigma}_{\boldsymbol{b}^{(l)}})$, where $\boldsymbol{\Sigma}_{\boldsymbol{b}^{(l)}}$ is a diagonal matrix. The diagonal matrix $\boldsymbol{\Phi}^{(l)} = \text{diag}(\phi_1^{(l)}, \cdots, \phi_{D_l}^{(l)}) \in \{0,1\}^{D_l \times D_l}$ denotes binary gating states of the l-th ReLU layer.

Then, we construct the surrogate DNN model with the same architecture as the BNN, to approximate the BNN's feature distribution. Parameters of this surrogate DNN model $\boldsymbol{\psi}$ are set as the mean of the weight distribution and the mean of the bias distribution in the BNN, *i.e.*, $\boldsymbol{\psi} = \{\boldsymbol{\mu}_{\boldsymbol{W}^{(l)}}, \boldsymbol{\mu}_{\boldsymbol{b}^{(l)}}\}_{l=1}^{L}$. Given an input sample $\boldsymbol{x}$, we add perturbations $\Delta\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_{\Delta\boldsymbol{x}})$ to input variables and perturbations $\Delta\boldsymbol{h}^{(l')} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_{\Delta\boldsymbol{h}^{(l')}})$ to features between the first layer and the $(l-1)$-th layer in the surrogate DNN model ($1 \le l' \le l-1$). In this way, we can obtain the distribution of the l-th layer feature $\tilde{h}^{(l)}$ in the surrogate DNN model, denoted as $p_{\text{DNN}}(\tilde{\boldsymbol{h}}^{(l)}|\boldsymbol{\Delta} = \{\boldsymbol{\Sigma}_{\Delta\boldsymbol{x}}, \boldsymbol{\Sigma}_{\Delta\boldsymbol{h}^{(1)}}, \cdots, \boldsymbol{\Sigma}_{\Delta\boldsymbol{h}^{(l-1)}}\})$, and we use $p_{\text{DNN}}(\tilde{\boldsymbol{h}}^{(l)}|\boldsymbol{\Delta})$ to mimic the feature distribution $p_{\text{BNN}}(\boldsymbol{h}^{(l)})$ in the BNN. Thus, the objective function is formulated as minimizing the following KL divergence.

$$\forall\, 1 \le l \le L, \quad \min_{\boldsymbol{\Delta}} \text{KL}(p_{\text{BNN}}(\boldsymbol{h}^{(l)}) \| p_{\text{DNN}}(\tilde{\boldsymbol{h}}^{(l)}|\boldsymbol{\Delta})), \quad (7)$$

where we set $\boldsymbol{\Sigma}_{\Delta\boldsymbol{x}}, \boldsymbol{\Sigma}_{\Delta\boldsymbol{h}^{(l')}} \in \boldsymbol{\Delta}$ as diagonal matrices.

However, it is difficult to directly optimize Eq. (7). Instead, we learn the covariance matrices in a layer-wise manner, as follows. First, we learn the covariance matrix $\boldsymbol{\Sigma}_{\Delta\boldsymbol{x}}$ on input variables to match the first-layer feature of the surrogate DNN model to the first-layer feature of the BNN, *i.e.*, $\min_{\boldsymbol{\Sigma}_{\Delta\boldsymbol{x}}} \text{KL}(p_{\text{BNN}}(\boldsymbol{h}^{(1)}) \| p_{\text{DNN}}(\tilde{\boldsymbol{h}}^{(1)}|\boldsymbol{\Sigma}_{\Delta\boldsymbol{x}}))$. Then, we fix the learned covariance matrix $\boldsymbol{\Sigma}_{\Delta\boldsymbol{x}}$ (note that it is not to fix the perturbation $\Delta\boldsymbol{x}$), and learn the covariance matrix $\boldsymbol{\Sigma}_{\Delta\boldsymbol{h}^{(1)}}$ on the first-layer feature to fit feature distributions of the second layer by minimizing $\text{KL}(p_{\text{BNN}}(\boldsymbol{h}^{(2)}) \| p_{\text{DNN}}(\tilde{\boldsymbol{h}}^{(2)}|\boldsymbol{\Sigma}_{\Delta\boldsymbol{x}}, \boldsymbol{\Sigma}_{\Delta\boldsymbol{h}^{(1)}}))$. We recursively learn the covariance matrix of an upper layer by fixing the covariance matrices in all lower layers, until the last layer.

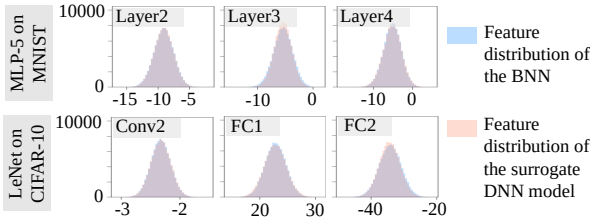

Figure 2. Comparison between the feature distribution of the BNN and the feature distribution of the surrogate DNN model. We randomly selected a feature dimension from each layer of the network. Each sub-figure compares feature distributions between the BNN and the surrogate DNN model in the selected dimension. Please see Appendix I for results on tabular datasets.

Table 1. Approximation error of the surrogate model and approximation error of the baseline distribution. The approximation error was measured using features of the last layer of the network.

	MLP-5 on MNIST	LeNet on CIFAR-10	MLP-8 on Census	MLP-8 on TV news
surrogate	**0.16**	**0.06**	**0.11**	**0.16**
baseline	21.38	19.68	4.79	4.50

Experimental verification. We trained BNNs on image datasets and tabular datasets to verify the quality of using the surrogate DNN model to approximate the feature distribution of the BNN. For image datasets, we tested BNNs with two architectures. For the MNIST dataset (LeCun et al., 1998), we constructed a BNN with the architecture of a 5-layer MLP. We also tested a BNN with the LeNet architecture (LeCun et al., 1998), which was trained on the CIFAR-10 dataset (Krizhevsky et al., 2009). We used two tabular datasets, including the UCI TV news dataset (termed *TV news*) and the UCI census income dataset (termed *Census*) (Dua & Graff, 2017). We constructed BNNs with an 8-layer MLP architecture for these tabular datasets. All MLPs contained 100 neurons in each hidden layer. For each BNN, we constructed a corresponding surrogate DNN model. Please see Appendix H for implementation details.

Figure 2 shows that the feature distribution of the surrogate DNN model well matched the feature distribution of the BNN. Furthermore, we used the KL divergence $\mathrm{KL}(p_{\text{BNN}}(\boldsymbol{h}^{(l)})\|p_{\text{DNN}}(\tilde{\boldsymbol{h}}^{(l)}|\boldsymbol{\Delta}))$ in Eq. (7) to measure the approximation error. To compare with $\mathrm{KL}(p_{\text{BNN}}(\boldsymbol{h}^{(l)})\|p_{\text{DNN}}(\tilde{\boldsymbol{h}}^{(l)}|\boldsymbol{\Delta}))$, we further constructed a simple baseline distribution of the features $p_{\text{base}}(\boldsymbol{h}^{(l)}) = \mathcal{N}(\hat{\mu}\mathbf{1}, \hat{\sigma}^2\boldsymbol{I})$, where $\hat{\mu}$ and $\hat{\sigma}^2$ denote the mean and the variance over all feature dimensions of the BNN, respectively. We computed $\mathrm{KL}(p_{\text{BNN}}(\boldsymbol{h}^{(l)})\|p_{\text{base}}(\boldsymbol{h}^{(l)}))$ for comparison. Table 1 shows that the approximation error of the surrogate DNN model was significantly smaller than the approximation error of the baseline distribution.

Experimental results showed that the weight uncertainty in a BNN could be well approximated by adding random perturbations to both input variables and low-layer features.

2.3. High-order concepts are sensitive to perturbations

In the previous subsection, we have demonstrated that adding random perturbations to input variables and low-layer features can successfully approximate the feature distribution in a BNN with weight uncertainty. In this way, proving the difficulty of BNNs in encoding high-order interactive concepts can be converted into the proof of the following two steps. First, in this subsection, we prove that high-order interactive concepts are more sensitive to perturbations than low-order interactive concepts, which is inspired by the proof in Zhou et al. (2023). Then, in the next subsection, we will prove that perturbation-sensitive concepts are difficult to be learned by a neural network.

Note that according to Section 2.2, introducing the weight uncertainty in a BNN can be approximated by adding random perturbations to both input variables and features of different layers. However, simultaneously adding perturbations to features of multiple layers significantly boosts the difficulty of analysis. Fortunately, adding perturbations to output features of the l-th layer can be considered as perturbing input variables of the $(l+1)$-th layer. Hence, in this subsection, we just analyze interactive concepts in a simple case where we perturb input variables in a certain layer, instead of analyzing the complex case of simultaneously perturbing features of different layers.

To prove that high-order interactive concepts are more sensitive to input perturbations than low-order interactive concepts, let us first derive the analytical form of the interaction effect $I(S)$ of an interactive concept.

Lemma 2.1. *Given a neural network v and an arbitrary input sample $\boldsymbol{x}' \in \mathbb{R}^n$, the network output can be decomposed using the Taylor expansion $v(\boldsymbol{x}') = \sum_{S\subseteq N}\sum_{\boldsymbol{\pi}\in Q_S} U_{S,\boldsymbol{\pi}} \cdot J(S,\boldsymbol{\pi}|\boldsymbol{x}')$. In this way, according to Eq. (3), the interaction effect $I(S|\boldsymbol{x}')$ on the sample $\boldsymbol{x}'$ can be reformulated as*

$$I(S|\boldsymbol{x}') = \sum\nolimits_{\boldsymbol{\pi}\in Q_S} U_{S,\boldsymbol{\pi}} \cdot J(S,\boldsymbol{\pi}|\boldsymbol{x}'), \tag{8}$$

where $J(S,\boldsymbol{\pi}|\boldsymbol{x}') = \prod_{i\in S}\left(\operatorname{sign}(x'_i - r_i)\cdot\frac{x'_i-r_i}{\tau}\right)^{\pi_i}$ denotes an expansion term of the degree $\boldsymbol{\pi}$, $\boldsymbol{\pi} \in Q_S = \{[\pi_1,\ldots,\pi_n]|\forall i \in S, \pi_i \in \mathbb{N}^+; \forall i \notin S, \pi_i = 0\}$. $U_{S,\boldsymbol{\pi}} = \frac{\tau^m}{\prod_{i=1}^n \pi_i!}\frac{\partial^m v(\boldsymbol{x}_\emptyset)}{\partial x_1^{\pi_1}\cdots\partial x_n^{\pi_n}}\cdot\prod_{i\in S}[\operatorname{sign}(x'_i - r_i)]^{\pi_i}$, $m = \sum_{i=1}^n \pi_i$.

Lemma 2.1 provides a new perspective to analyze the sensitivity of the interaction effect $I(S)$. In particular, just like in Ren et al. (2023a) and Ren et al. (2023b), we mask the input variable x_i by setting it to its reference value $x_i \leftarrow r_i$. The reference value r_i is designed as follows. Let $\mathbb{E}_{\boldsymbol{x}}[x_i]$ denote the average value of the input variable x_i over all input samples, which is usually regarded as a no-information state of this input variable (Ancona et al., 2019). In this paper, we remove the information from the input variable x_i by pushing x_i by a large enough distance τ towards its mean value. In other words, if $x_i > \mathbb{E}_{\boldsymbol{x}}[x_i]$, we set the reference value $r_i = x_i - \tau$[1]; otherwise, $r_i = x_i + \tau$. Here, $\tau \in \mathbb{R}$ is a pre-defined constant. In this way, compared to setting $r_i = \mathbb{E}_{\boldsymbol{x}}[x_i]$, the above setting ensures comparable perturbation magnitudes over different input dimensions.

Furthermore, in order to simplify the proof, when we add a small Gaussian perturbation $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$ to the sample $\boldsymbol{x}$, we ignore the extremely low possibility of large perturbations $|\epsilon_i| \geq \tau$ because the variance σ^2 is small.

Let us start with a simple case in Lemma 2.1. Since people usually adopt low-order Taylor expansion for approximation in real implementations, we first approximate the interaction effect $I(S|\boldsymbol{x}')$ using the expansion term of the lowest degree, and analyze the influence of input perturbations on $I(S|\boldsymbol{x}')$.

Theorem 2.2. *Let $\hat{\boldsymbol{\pi}}$ denote the lowest degree of the expan-*

[1]We need to avoid the case of over-perturbation, by setting $r_i \leftarrow \max(r_i, \mathbb{E}_{\boldsymbol{x}}[x_i])$, if $x_i > \mathbb{E}_{\boldsymbol{x}}[x_i]$; $r_i \leftarrow \min(r_i, \mathbb{E}_{\boldsymbol{x}}[x_i])$, otherwise. However, such cases are not common in real applications, so we ignore such settings in the following analysis.

sion terms of the interaction effect $I(S|\boldsymbol{x}')$, i.e., $\forall i \in S, \hat{\pi}_i = 1; \forall i \notin S, \hat{\pi}_i = 0$. Let us consider the interaction effect $I(S|\boldsymbol{x}')$ only containing the expansion term of the lowest degree, i.e., $I(S|\boldsymbol{x}') = U_{S,\hat{\boldsymbol{\pi}}} \cdot J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}')$. In this way, the mean and variance of the interaction effect $I(S|\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\epsilon})$ over different perturbations $\boldsymbol{\epsilon}$ are given as

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})] &= U_{S,\hat{\boldsymbol{\pi}}}, \\ \mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})] &= U_{S,\hat{\boldsymbol{\pi}}}^2((1+(\sigma/\tau)^2)^{|S|}-1). \end{aligned} \tag{9}$$

Theorem 2.2 proves that *the variance $\mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]$ increases along with the order $|S|$ of the interactive concept in an exponential manner*. It indicates that high-order interactive concepts are much more sensitive to input perturbations than low-order concepts. Furthermore, as mentioned in Section 2.2, since we can add perturbations to a surrogate DNN model to well mimic feature representations of a BNN, **we can consider that high-order interactive concepts encoded by the BNN are much more sensitive to weight uncertainty in the BNN than low-order concepts.**

Theorem 2.3 (Proof in Appendix G.3)**.** *Let $\boldsymbol{\pi} \in Q_S = \{[\pi_1, \ldots, \pi_n] | \forall i \in S, \pi_i \in \mathbb{N}^+; \forall i \notin S, \pi_i = 0\}$ denote an arbitrary degree. Then, the mean and the variance of $J(S, \boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})$ over perturbations $\boldsymbol{\epsilon}$ are*

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathbb{E}_{\boldsymbol{\epsilon}}[\textstyle\prod_{i\in S}(1+\frac{\epsilon_i}{\tau})^{\pi_i}], \\ \mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathrm{Var}_{\boldsymbol{\epsilon}}[\textstyle\prod_{i\in S}(1+\frac{\epsilon_i}{\tau})^{\pi_i}] \end{aligned} \tag{10}$$

Theorem 2.3 extends Theorem 2.2 to a **general case**, where we use a higher-order Taylor expansion to represent $I(S|\boldsymbol{x}')$.

Theorem 2.4 (Proof in Appendix G.4)**.** *Let S and S' be two interactive concepts, such that $S \subsetneq S'$. Let us consider expansion terms $J(S, \boldsymbol{\pi})$ and $J(S', \boldsymbol{\pi}')$, where the term $J(S', \boldsymbol{\pi}')$ is extended from the term $J(S, \boldsymbol{\pi})$ with $\boldsymbol{\pi} \prec \boldsymbol{\pi}'$. I.e., (1) $\forall i \in S', \pi_i' \in \mathbb{N}^+$; otherwise, $\pi_i' = 0$. (2) Given $\boldsymbol{\pi}'$, $\forall j \in S, \pi_j = \pi_j'$; otherwise, $\pi_j = 0$. Then, we have*

$$\begin{aligned} &\frac{\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]}{\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]} > \textstyle\prod_{i\in S'\setminus S}\mathbb{E}_{\epsilon_i}^2[(1+\frac{\epsilon_i}{\tau})^{\pi_i'}], \\ &\frac{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]}{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]} \\ &\qquad < \frac{1}{\prod_{i\in S'\setminus S}\mathbb{E}_{\epsilon_i}[(1+\frac{\epsilon_i}{\tau})^{\pi_i'}]}, \end{aligned} \tag{11}$$

and we can also obtain $\mathbb{E}_{\epsilon_i}[(1+\frac{\epsilon_i}{\tau})^{\pi_i'}] \geq 1$.

Theorem 2.4 indicates that for an arbitrary degree $\boldsymbol{\pi}$ of the interactive concept $\boldsymbol{S}$, $\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S', \boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S, \boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]$ increases in an exponential manner along with $|S' \setminus S| = |S'| - |S|$. Therefore, we can roughly consider that $\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S, \boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]$ increases exponentially *w.r.t.* the order $|S|$. Furthermore, according to Lemma 2.1, $I(S|\boldsymbol{x}+\boldsymbol{\epsilon})$ can be re-written as the weighted sum of $J(S, \boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})$. Since coefficients $U_{S,\boldsymbol{\pi}}$ *w.r.t.* different S and $\boldsymbol{\pi}$ are usually chaotic, we can roughly consider that the sensitivity of $I(S|\boldsymbol{x}+\boldsymbol{\epsilon})$ also grows exponentially along with the order $|S|$ of the interactive concept S. In addition, Theorem 2.4 also proves the approximately exponential decrease of $\frac{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]}{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]}$ along with $|S'| - |S|$. Similarly, we can obtain that the relative stability $\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]$ decreases along with the order $|S|$.

Conclusions. Both Theorem 2.2 and Theorem 2.4 tell us that high-order interactive concepts are much more sensitive to input perturbations. Furthermore, combined with the conclusion in Section 2.2, **we can conclude that high-order interactive concepts encoded by the BNN are much more sensitive to the weight uncertainty in the BNN than low-order concepts.**

Experimental verification. We conducted experiments to verify the above conclusions. To verify the sensitivity to input perturbations, we added a random perturbation $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$ to a given input sample $\boldsymbol{x}$, where $\sigma^2 = 0.05^2$. Then, we used the following two metrics, $V_{\text{noise}}^{(s)} = \mathbb{E}_{\boldsymbol{x}}[\mathbb{E}_{|S|=s}[\mathrm{Var}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I})}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]]]$ and $K_{\text{noise}}^{(s)} = \mathbb{E}_{\boldsymbol{x}}[\mathbb{E}_{|S|=s}[\frac{|\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I})}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\mathrm{Var}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I})}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]}]]$, to measure the average variance and the average relative stability of the s-order interactive concepts *w.r.t.* the input perturbation $\boldsymbol{\epsilon}$. Then, a large $V_{\text{noise}}^{(s)}$ or a small $K_{\text{noise}}^{(s)}$ indicated that the s-order interactive concepts were sensitive to input perturbations.

Similarly, to verify the sensitivity to the weight uncertainty, we sampled different weights $\boldsymbol{W}$ from the weight distribution $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ of the BNN. Then, we used $V_{\text{BNN}}^{(s)} = \mathbb{E}_{\boldsymbol{x}}[\mathbb{E}_{|S|=s}[\mathrm{Var}_{\boldsymbol{W}\sim q_{\boldsymbol{\theta}}(\boldsymbol{W})}[I(S|\boldsymbol{x},\boldsymbol{W})]]]$ and $K_{\text{BNN}}^{(s)} = \mathbb{E}_{\boldsymbol{x}}[\mathbb{E}_{|S|=s}[\frac{|\mathbb{E}_{\boldsymbol{W}\sim q_{\boldsymbol{\theta}}(\boldsymbol{W})}[I(S|\boldsymbol{x},\boldsymbol{W})]|}{\mathrm{Var}_{\boldsymbol{W}\sim q_{\boldsymbol{\theta}}(\boldsymbol{W})}[I(S|\boldsymbol{x},\boldsymbol{W})]}]]$ to measure the average variance and the average relative stability of the s-order interactive concepts *w.r.t.* the weight uncertainty in the BNN. Therefore, a large value of $V_{\text{BNN}}^{(s)}$ or a small value of $K_{\text{BNN}}^{(s)}$ indicated that the s-order interactive concepts were sensitive to the weight uncertainty. We followed experimental settings in the *experiments* paragraph in Section 2.2 to train BNNs. Specifically, we trained BNNs with the MLP architecture on the MNIST dataset, the TV news dataset, and the Census dataset. We trained BNNs with the LeNet architecture on the CIFAR-10 dataset. Appendix H introduces how to efficiently compute $I(S|\boldsymbol{x})$ on images.

Figure 3 shows that the average variance $V_{\text{noise}}^{(s)}$ and $V_{\text{BNN}}^{(s)}$ increased exponentially along with the order s, while the relative stability $K_{\text{noise}}^{(s)}$ and $K_{\text{BNN}}^{(s)}$ both decreased along with the order. This demonstrated that high-order interactive concepts were much more sensitive to input perturbations and the weight uncertainty in the BNN, thereby verifying Theorem 2.2 and Theorem 2.4.

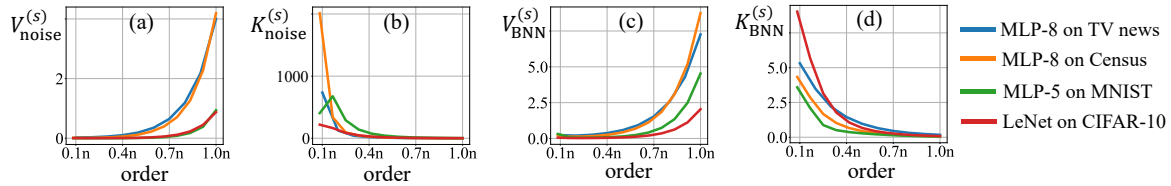

Figure 3. (a) The exponential increase of the average variance $V^{(s)}_{\text{noise}}$ and (b) the roughly exponential decrease of the average relative stability $K^{(s)}_{\text{noise}}$ along with the order s, under perturbations from a distribution $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, 0.05^2 \cdot \boldsymbol{I})$. (c) The exponential increase of the average variance $V^{(s)}_{\text{BNN}}$ and (d) the roughly exponential decrease of the average relative stability $K^{(s)}_{\text{BNN}}$ along with the order s, under weight uncertainty in the BNN.

2.4. Perturbation-sensitive concepts are difficult to learn

In this subsection, we prove that high-order interactive concepts, which are sensitive to input perturbations and weight uncertainty, are difficult to be learned by a BNN in a regression task. Specifically, we measure the learning effects of interactive concepts (denoted by U_S), and Theorems 2.5 and 2.6 prove the small learning effects of perturbation-sensitive concepts.

To facilitate the analysis, we first simplify the conceptual learning as a linear problem. Specifically, we first rewrite the interaction effect of an interactive concept S. Given an input sample $\boldsymbol{x}$, according to Eq. (8), the interaction effect of the concept S on the sample $\boldsymbol{x}'$ (obtained by applying some transformations on $\boldsymbol{x}$), $I(S|\boldsymbol{x}')$, can be rewritten as

$$I(S|\boldsymbol{x}') = U_S \cdot C_S(\boldsymbol{x}'), \tag{12}$$

where the constant $U_S = I(S|\boldsymbol{x})$ denotes the interaction effect of the concept S, and the function for the activation state is given as $C_S(\boldsymbol{x}') = \sum_{\boldsymbol{\pi}\in Q_S} U_{S,\boldsymbol{\pi}} J(S, \boldsymbol{\pi}|\boldsymbol{x}')/U_S$.

Understanding of $C_S(\boldsymbol{x}')$. Let us consider a sample $\boldsymbol{x}'$ where each input variable x'_i is either masked by the reference value r_i or kept unchanged as x_i. Then, the function $C_S(\boldsymbol{x}')$ defined above represents the binary activation state of the concept S in the sample $\boldsymbol{x}'$, which is an AND relationship between all variables in S:

$$C_S(\boldsymbol{x}') = \bigwedge\nolimits_{i\in S} \text{unmask}(x'_i), \tag{13}$$

where the binary function $\text{unmask}(x'_i) \in \{0, 1\}$ checks whether the i-th variable x'_i is masked in the sample $\boldsymbol{x}'$. If the i-th variable is masked, then $\text{unmask}(x'_i) = 0$; otherwise, $\text{unmask}(x'_i) = 1$.

Only when all input variables in S are not masked in the sample $\boldsymbol{x}'$, the concept S is activated, and $C_S(\boldsymbol{x}') = 1$. If any input variable in S is masked, then the concept S will not be activated ($C_S(\boldsymbol{x}') = 0$), yielding zero interaction effect $I(S|\boldsymbol{x}') = 0$.

Thus, we can extend Eq. (4) to a continuous version that explains the output as a **linear regression problem**.

$$v(\boldsymbol{x}') = \sum\nolimits_{S\in\Omega} U_S \cdot C_S(\boldsymbol{x}'), \tag{14}$$

where the activation state $C_S(\boldsymbol{x}')$ can be considered as an input dimension of the linear function, which reflects whether the input sample $\boldsymbol{x}'$ contains the concept S.

Therefore, the absolute value of the coefficient U_S can be considered as *the strength of the neural network in learning the interactive concept S*. According to Section 2.1 and Ren et al. (2023a), most interactive concepts have negligible coefficients $|U_S| \approx 0$, so we can consider that the neural network only encodes a few interactive concepts S with large absolute values $|U_S|$.

Let us facilitate the poof on a regression task. Based on the conclusion in Section 2.2, we can roughly consider that training a BNN on normal samples is equivalent to training a surrogate DNN model on perturbed input samples $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\epsilon}$. Then, according to Eq. (14), the learning of the BNN on a certain input sample can be roughly represented as $\min_{\{U_S|S\in\Omega\}} L(\{U_S\})$, and the loss is given by

$$\begin{aligned} L(\{U_S\}) &= \mathbb{E}_{\boldsymbol{\epsilon}}\left[(y^* - v(\boldsymbol{x}'))^2\right] \\ &= \mathbb{E}_{\boldsymbol{\epsilon}}[(y^* - \sum\nolimits_{S\in\Omega} U_S \cdot C_S(\boldsymbol{x}+\boldsymbol{\epsilon}))^2] \end{aligned} \tag{15}$$

where $\boldsymbol{x}$ and y^* denote the input sample and the ground-truth output, respectively, and $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\epsilon}$.

Theorem 2.5 (Proof in Appendix G.5)**.** *Given two random interactive concepts S and S', we can roughly assume that $C_S(\boldsymbol{x}+\boldsymbol{\epsilon})$ is independent of $C_{S'}(\boldsymbol{x}+\boldsymbol{\epsilon})$, because the two concepts S and S' usually have little overlap in most cases. Let $\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]$ and $\text{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]$ denote the mean and the variance of $C_S(\boldsymbol{x}+\boldsymbol{\epsilon})$ w.r.t. $\boldsymbol{\epsilon}$, respectively. Then, the solution to Eq. (15) satisfies the following property:*

$$\forall\, S \in \Omega, \quad |U^*_S| \propto |\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]/\text{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]| \tag{16}$$

Theorem 2.5 proves that the learning effect of an interactive concept S, measured by $|U^*_S|$, is proportional to the relative stability of the activation state of the interactive concept $|\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]/\text{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]|$ *w.r.t.* perturbations $\boldsymbol{\epsilon}$. This

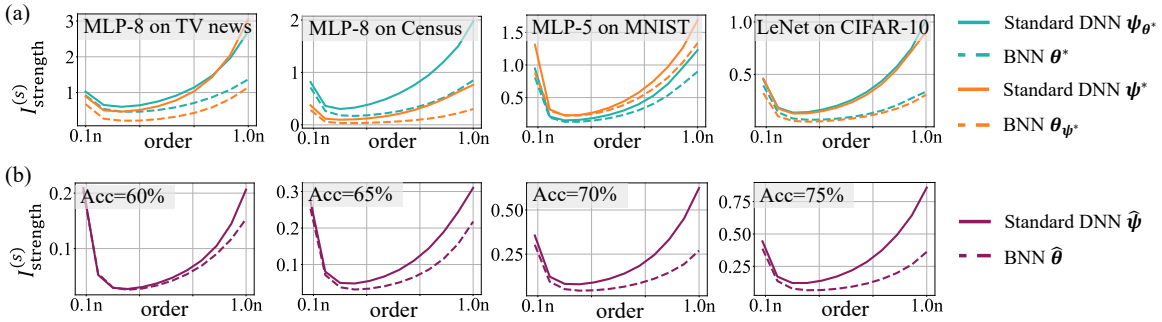

Figure 4. (a) Comparison of the strength of interactive concepts (i) between a trained BNN $\boldsymbol{\theta}^*$ and the constructed standard DNN $\psi_{\boldsymbol{\theta}^*}$, (ii) between a trained standard DNN ψ^* and the constructed BNN $\boldsymbol{\theta}_{\psi^*}$. (b) We trained a standard DNN $\hat{\psi}$ and a BNN $\hat{\boldsymbol{\theta}}$ with the LeNet architecture on the CIFAR-10 dataset, and compared the strength of interactive concepts between the two networks when the two networks were trained to have the same training accuracy.

indicates that perturbation-sensitive interactive concepts are more difficult to learn.

Theorem 2.6 (Proof in Appendix G.6). *Let* $A^{min} = \min_S |U_S|$ *and* $A^{max} = \max_S |U_S|$ *denote the lower bound and the upper bound of* $|U_S|$ *over all interactive concepts* S. *Then, for any* $S \subseteq N$, *we have*

$$A^{min} \cdot \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\text{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]} \le \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\text{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]} \le A^{max} \cdot \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\text{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]} \tag{17}$$

Theorem 2.6 proves that high-order (complex) interactive concepts have low relative stability *w.r.t.* perturbations $\boldsymbol{\epsilon}$. In fact, both Theorem 2.4 and Figure 3 have told us that $|\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]/\text{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]|$ significantly decreases along with the order $s = |S|$ of the interactive concept S. Therefore, both the lower bound and the upper bound of $|\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]/\text{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]|$ in Eq. (17) decrease along with the order s significantly. In this way, we can approximately consider that the strength of encoding a concept $|U^*_S| \propto |\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]/\text{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]|$ also decreases along with the order of interactive concepts. In other words, we prove that high-order interactive concepts are more difficult to be learned under perturbations $\boldsymbol{\epsilon}$. Combining the conclusion in Section 2.2, we also prove that high-order interactive concepts are more difficult to be learned by the BNN.

3. Experiments

In this section, we experimentally verified that compared to standard DNNs, BNNs were less likely to encode high-order (complex) interactive concepts. Specifically, we constructed three pairs of baseline networks for comparison.

(1) Given a trained BNN $\boldsymbol{\theta}^*$, we constructed a standard DNN by setting its weights to the mean value of the weight distribution of the BNN. The standard DNN was denoted by $\psi_{\boldsymbol{\theta}^*}$. Then, we compared the strength of all high-order interactive concepts between the BNN $\boldsymbol{\theta}^*$ and the standard DNN $\psi_{\boldsymbol{\theta}^*}$ without weight/feature uncertainty.

(2) Similarly, given a trained standard DNN ψ^*, we constructed a BNN $\boldsymbol{\theta}_{\psi^*}$ by setting the mean value of its weight distribution to the weights of the standard DNN. We set all weight dimensions in the l-th layer of the BNN to share the same variance σ_l^2, where σ_l^2 was computed as the average of variances of all weight dimensions in the l-th layer of the previous BNN $\boldsymbol{\theta}^*$. Then, we compared the strength of high-order interactive concepts between the standard DNN ψ^* and the BNN $\boldsymbol{\theta}_{\psi^*}$.

(3) We trained a standard DNN and a BNN with the same architecture. Then, we compared the strength of high-order interactive concepts between each pair of standard DNN $\hat{\psi}$ and the BNN $\hat{\boldsymbol{\theta}}$ when these two networks were trained to have the same training accuracy. We used the training accuracy to align the learning progress of the two networks for fair comparison.

Specifically, the average strength of the s-order interactive concepts was measured as $I^{(s)}_{\text{strength}} = \mathbb{E}_{\boldsymbol{x}}[\mathbb{E}_{S\subseteq N,|S|=s}[|I(S|\boldsymbol{x})|]]$. To compute the interaction effect $I(S|\boldsymbol{x})$, we set $v(\boldsymbol{x}_S) = \log \frac{p(y=y^*|\boldsymbol{x}_S)}{1-p(y=y^*|\boldsymbol{x}_S)} \in \mathbb{R}$, which reflected the confidence of classifying the masked input sample $\boldsymbol{x}_S$ into the ground-truth category y^*. For standard DNNs, $p(y=y^*|\boldsymbol{x}_S)$ referred to the classification probability of the ground-truth category on the masked sample $\boldsymbol{x}_S$. For BNNs, $p(y=y^*|\boldsymbol{x}_S)$ was computed according to Eq. (2), where we sampled ten neural networks from the weight distribution $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ of the BNN, and computed the average classification probability over all these networks.

We followed experimental settings in the *experiments* paragraph in Section 2.2 to train the networks. Specifically, we trained standard DNNs and BNNs with the MLP architec-

ture on the TV news dataset, the Census dataset, and the MNIST dataset. We trained standard DNNs and BNNs with the LeNet architecture on the CIFAR-10 dataset. Appendix H introduces how to efficiently compute $I(S|\boldsymbol{x})$ on images. Figure 4 shows that the strength of high-order interactive concepts of BNNs was much weaker than that of standard DNNs in all comparisons. This verified that BNNs were less likely to encode high-order (complex) interactive concepts than standard DNNs.

4. Conclusion and discussion

In this paper, we have proven the tendency of mean-field variational BNNs to avoid encoding high-order (complex) concepts. Many studies (Ren et al., 2023a; Li & Zhang, 2023; Ren et al., 2023c) have shown that there does exist a concept-emerging phenomenon when a neural network is sufficiently trained.

Besides, as discussed in the introduction, encoding less complex concepts does not mean that BNNs have weaker representation power than standard DNNs, because a standard DNN can be considered as a specific BNN with zero weight uncertainty. More crucially, Ren et al. (2021) and Lengerich et al. (2022) proved that high-order concepts are usually vulnerable to adversarial attacks and have weak generalization power. Thus, it is hard to say whether the tendency to avoid encoding complex concepts is a demerit or not.

Acknowledgements. This work is partially supported by the National Nature Science Foundation of China (62276165), National Key R&D Program of China (2021ZD0111602), Shanghai Natural Science Foundation (21JC1403800,21ZR1434600), National Nature Science Foundation of China (U19B2043). This work is also partially supported by Huawei Technologies Inc.

References

Ancona, M., Oztireli, C., and Gross, M. Explaining deep neural networks with a polynomial time algorithm for shapley value approximation. In *International Conference on Machine Learning*, pp. 272–281. PMLR, 2019.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural network. In *International conference on machine learning*, pp. 1613–1622. PMLR, 2015.

Carbone, G., Wicker, M., Laurenti, L., Patane, A., Bortolussi, L., and Sanguinetti, G. Robustness of bayesian neural networks to gradient-based attacks. *Advances in Neural Information Processing Systems*, 33:15602–15613, 2020.

Deng, H., Ren, Q., Zhang, H., and Zhang, Q. Discovering and Explaining the Representation Bottleneck of DNNs. In *International Conference on Learning Representations*, 2022.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Engstrom, L., Ilyas, A., Santurkar, S., Tsipras, D., Tran, B., and Madry, A. Adversarial robustness as a prior for learned representations, 2019.

Etmann, C., Lunz, S., Maass, P., and Schönlieb, C.-B. On the connection between adversarial robustness and saliency map interpretability, 2019.

Foong, A., Burt, D., Li, Y., and Turner, R. On the expressiveness of approximate inference in bayesian neural networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15897–15908. Curran Associates, Inc., 2020.

Fortuin, V., Garriga-Alonso, A., Ober, S. W., Wenzel, F., Ratsch, G., Turner, R. E., van der Wilk, M., and Aitchison, L. Bayesian neural network priors revisited. In *International Conference on Learning Representations*, 2022.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Gal, Y. and Smith, L. Sufficient conditions for idealised models to have no adversarial examples: a theoretical and empirical study with bayesian neural networks, 2018.

Grabisch, M. and Roubens, M. An axiomatic approach to the concept of interaction among players in cooperative games. *International Journal of game theory*, 28(4):547–565, 1999.

Harsanyi, J. C. A simplified bargaining model for the n-person cooperative game. *International Economic Review*, 4(2):194–220, 1963. ISSN 00206598, 14682354.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial examples are not bugs, they are features. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Janizek, J. D., Sturmfels, P., and Lee, S.-I. Explaining explanations: Axiomatic feature interactions for deep networks. *J. Mach. Learn. Res.*, 22:104–1, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

Krishnan, R., Subedar, M., and Tickoo, O. Specifying weight priors in bayesian deep neural networks with empirical bayes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4477–4484, 2020.

Kristiadi, A., Hein, M., and Hennig, P. Being bayesian, even just a bit, fixes overconfidence in relu networks. In *International conference on machine learning*, pp. 5436–5446. PMLR, 2020.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

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

Lengerich, B. J., Xing, E., and Caruana, R. Dropout as a regularizer of interaction effects. In Camps-Valls, G., Ruiz, F. J. R., and Valera, I. (eds.), *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 7550–7564. PMLR, 28–30 Mar 2022.

Li, M. and Zhang, Q. Does a Neural Network Really Encode Symbolic Concepts? *International Conference on Machine Learning*, 2023.

Lundberg, S. M., Erion, G. G., and Lee, S.-I. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Olah, C., Mordvintsev, A., and Schubert, L. Feature visualization. *Distill*, 2017. doi: 10.23915/distill.00007. https://distill.pub/2017/feature-visualization.

Ren, J., Zhang, D., Wang, Y., Chen, L., Zhou, Z., Chen, Y., Cheng, X., Wang, X., Zhou, M., Shi, J., and Zhang, Q. A Unified Game-Theoretic Interpretation of Adversarial Robustness. In *Advances in Neural Information Processing Systems*, volume 34, pp. 3797–3810. Curran Associates, Inc., 2021.

Ren, J., Li, M., Chen, Q., Deng, H., and Zhang, Q. Defining and Quantifying the Emergence of Sparse Concepts in DNNs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20280–20289, June 2023a.

Ren, J., Zhou, Z., Chen, Q., and Zhang, Q. Can We Faithfully Represent Masked States to Compute Shapley Values on a DNN? In *International Conference on Learning Representations*, 2023b.

Ren, Q., Gao, J., Shen, W., and Zhang, Q. Where We Have Arrived in Proving the Emergence of Sparse Symbolic Concepts in AI Models. *arXiv preprint arXiv:2305.01939*, 2023c.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Sundararajan, M., Dhamdhere, K., and Agarwal, A. The shapley taylor interaction index. In *International Conference on Machine Learning*, pp. 9259–9268. PMLR, 2020.

Touvron, H., Bojanowski, P., Caron, M., Cord, M., El-Nouby, A., Grave, E., Izacard, G., Joulin, A., Synnaeve, G., Verbeek, J., and Jégou, H. Resmlp: Feedforward networks for image classification with data-efficient training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–9, 2022. doi: 10.1109/TPAMI.2022.3206148.

Wang, X., Ren, J., Lin, S., Zhu, X., Wang, Y., and Zhang, Q. A Unified Approach to Interpreting and Boosting Adversarial Transferability. In *International Conference on Learning Representations*, 2021.

Warstadt, A., Singh, A., and Bowman, S. R. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Wenzel, F., Roth, K., Veeling, B. S., Swiatkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. How good is the bayes posterior in deep neural networks really? In *International conference on machine learning*, 2020.

Willink, R. Normal moments and hermite polynomials. *Statistics & Probability Letters*, 73(3):271–275, 2005. ISSN 0167-7152. doi: https://doi.org/10.1016/j.spl.2005.03.015.

Wu, A., Nowozin, S., Meeds, E., Turner, R. E., Hernandez-Lobato, J. M., and Gaunt, A. L. Deterministic variational inference for robust bayesian neural networks. In *International Conference on Learning Representations*, 2019.

Zhang, H., Li, S., Ma, Y., Li, M., Xie, Y., and Zhang, Q. Interpreting and Boosting Dropout from a Game-Theoretic View. In *International Conference on Learning Representations*, 2021.

Zhang, J., Hua, Y., Song, T., Wang, H., Xue, Z., Ma, R., and Guan, H. Improving bayesian neural networks by adversarial sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

Zhou, H., Zhang, H., Deng, H., Liu, D., Shen, W., Chan, S.-H., and Zhang, Q. Concept-Level Explanation for the Generalization of a DNN. *arXiv preprint arXiv:2302.13091*, 2023.

A. Discussion on literature in representation capacities of BNNs

Many studies investigated the representation capacity of BNNs from different perspectives. Gal & Smith (2018) and Carbone et al. (2020) proved that BNNs were robust to adversarial attacks. Kristiadi et al. (2020) proved that BNNs could mitigate the over-confidence problem in standard ReLU networks. Wenzel et al. (2020) considered that the poor performance of BNNs was due to the inappropriate prior distribution of weights in the BNN, and a series of studies (Wu et al., 2019; Krishnan et al., 2020; Fortuin et al., 2022) found that using carefully-designed prior distributions of weights could improve the performance of the BNN. Zhang et al. (2022) also showed that adding adversarial perturbations to weights during training could improve the performance of the BNN. Besides, Foong et al. (2020) proved that using either fully-factorized Gaussian distributions or dropout operations to approximate the posterior distribution of a BNN would lead to inaccurate uncertainty estimation of the network prediction. Unlike previous studies, we focus on the conceptual representation of BNNs, and theoretically prove that mean-field variational BNNs are less likely to encode complex interactive concepts than standard DNNs.

B. Discussion on literature in interactions in neural networks

Interactions in game theory are often used to explain neural networks and are closely related to the quantification of concepts. Grabisch & Roubens (1999) first proposed the Shapley interaction index, and Lundberg et al. (2018) later used this index to explain tree ensembles. Janizek et al. (2021) explained the pairwise feature interaction in DNNs, while Sundararajan et al. (2020) proposed the Shapley Taylor interaction index to quantify interactions among multiple input variables. Ren et al. (2023a) used game-theoretic interactions to analyze the emergence of concepts in the training of neural networks, and proved the faithfulness and sparsity of such formulation of concepts. In this paper, we follow the definition of concepts in Ren et al. (2023a), and prove BNNs' tendency to avoid encoding high-order (complex) concepts.

C. Discussion on literature in the connection between adversarial robustness and interpretability

Recent studies have shown that adversarial robustness is closely related to the interpretability of neural networks. Etmann et al. (2019) discovered and explained the phenomenon that adversarially robust models exhibit simpler and more human-interpretable saliency maps. Engstrom et al. (2019) demonstrated that adversarially robust models showed clear human-recognizable features when using the optimization-based feature visualization method (Olah et al., 2017), and the mapping from input images to intermediate features of the model is approximately invertible. Ilyas et al. (2019) demonstrated that adversarial samples can be attributed to the existence of non-robust features (features that are noisy and not interpretable to humans, but are highly predictive). Ren et al. (2021) showed that high-order (complex) interactive concepts encoded by neural networks are vulnerable to adversarial attacks, and that adversarially-trained DNNs encode more discriminative low-order (simple) interactive concepts than standard DNNs. In this paper, we prove that BNNs tend to avoid encoding high-order (complex) interactive concepts, which implies that BNNs may exhibit good adversarial robustness, from the perspective of conceptual representations.

D. Experiments on the connection between conceptual complexity and adversarial robustness

We show experimental results in (Ren et al., 2021) to demonstrate that high-order (complex) interactive concepts are more vulnerable to adversarial attacks, as illustrated in Figure 5. Although the interaction used in (Ren et al., 2021) was a bit different from the interaction used in this paper, we can prove that the Harsanyi dividend interaction in this paper is the elementary component of the multi-order interaction in (Ren et al., 2021). Thus, experimental results still reflect adversarial vulnerability of high-order interactive concepts. Please see Ren et al. (2021) for more details.

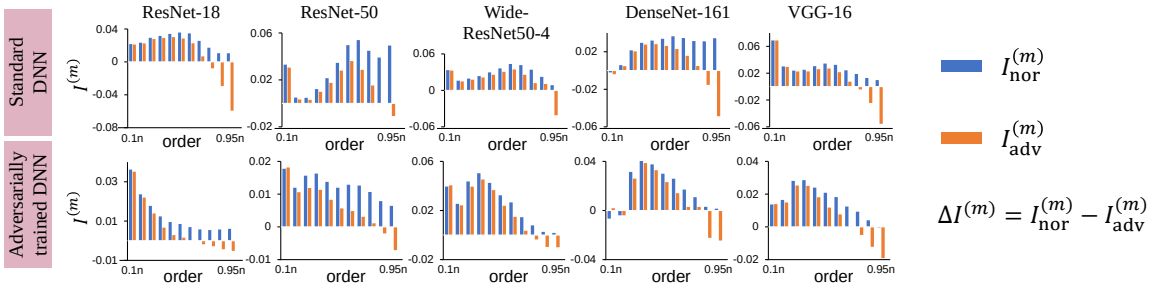

Figure 5. Adversarial attacks mainly affect high-order interactive concepts. Please refer to Ren et al. (2021) for more details.

E. Comparison of adversarial robustness between BNNs and standard DNNs

This experiment compares the adversarial robustness between BNNs and standard DNNs. Specifically, we train two BNNs with 8-layer MLP architecture on tabular datasets, including the Census dataset and the TV news dataset. All MLPs contain 100 neurons in each hidden layer. For each trained BNN $\boldsymbol{\theta}^*$, we compare this BNN with a standard DNN $\psi_{\boldsymbol{\theta}^*}$ that is constructed by following the experimental setting of **Comparison (1)** in Section 3. In other words, the standard DNN $\psi_{\boldsymbol{\theta}^*}$ is constructed by setting its weights to the mean value of the weight distribution of the BNN. Thus, with such experimental settings, the main difference between the BNN $\boldsymbol{\theta}^*$ and the DNN $\psi_{\boldsymbol{\theta}^*}$ is the weight uncertainty of BNNs, so that our experiment can faithfully reflect the impact of weight uncertainty of BNNs on the adversarial robustness.

We compare the classification accuracy on adversarial samples in the testing set between the BNN $\boldsymbol{\theta}^*$ and the standard DNN $\psi_{\boldsymbol{\theta}^*}$. To this end, for each pair of BNN and standard DNN, we adopt the untargeted PGD adversarial attack (Madry et al., 2018) based on the l_∞ norm, and accordingly obtain their accuracies on adversarial samples. In the PGD attack based on the l_∞ norm, an adversarial sample $\tilde{\boldsymbol{x}}$ is constrained wthin the l_∞-ball around the original sample $\boldsymbol{x}$, *i.e.*, $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_\infty \leq \epsilon$. We conduct the attack for 20 steps with $\epsilon = 0.1$, and set the step size to 0.01. Table 2 shows that BNNs exhibit higher adversarial accuracies than the corresponding DNNs, which indicates that BNNs are more robust to adversarial attacks.

Table 2. Adversarial accuracies of the BNN $\boldsymbol{\theta}^*$ and the standard DNN $\psi_{\boldsymbol{\theta}^*}$ constructed based on the BNN.

	MLP-8 on Census	MLP-8 on TV news
BNN $\boldsymbol{\theta}^*$	**77.51%**	**53.54%**
DNN $\psi_{\boldsymbol{\theta}^*}$	75.22%	50.54%

F. Experiments on the connection between conceptual complexity and generalization power.

Zhou et al. (2023) have investigated the connection between the generalization ability of an interactive concept encoded by a neural network and the complexity (order) of this concept.

The generalization ability of a concept S is defined as follows. For a generalizable concept S, if the concept is frequently extracted from training samples, then it is supposed to be also frequently extracted from testing samples and to make consistently positive (or consistently negative) effects to the classification of a certain category. Otherwise, this concept would not be considered generalizable. Thus, the generalization ability of a specific concept can be evaluated by whether this concept's interaction effects over training samples are similar to its interaction effects over testing samples. To this end, Zhou et al. (2023) quantified the *average generalization ability* $g^{(m)}$ over all m-order interactive concepts by the similarity

between interaction effects of m-order interactive concepts in training samples and those in testing samples:

$$g^{(m)} \overset{\text{def}}{=} \mathbb{E}_c \left[\text{sim} \left(I^{(m)}_{\text{train, c}}, I^{(m)}_{\text{test ,c}} \right) \right], \tag{18}$$

where the vector $I^{(m)}_{\text{train, c}} = \left[\mathbb{E}_{\boldsymbol{x}\in\text{train},c} \left[I\left(S_1|\boldsymbol{x}\right) \right], \mathbb{E}_{\boldsymbol{x}\in\text{train},c} \left[I\left(S_2|\boldsymbol{x}\right) \right], \ldots, \mathbb{E}_{\boldsymbol{x}\in\text{train},c} \left[I\left(S_d|\boldsymbol{x}\right) \right] \right]^\top \in \mathbb{R}^d$ denotes interaction effects of all m-order interactive concepts $[S_1, \cdots, S_d]$. The interaction effect of each concept $\mathbb{E}_{\boldsymbol{x}\in\text{train},c} \left[I\left(S_i|\boldsymbol{x}\right) \right]$ is averaged over different training samples in the category c. Accordingly, the vector $I^{(m)}_{\text{test, c}} = \left[\mathbb{E}_{\boldsymbol{x}\in\text{test},c} \left[I\left(S_1|\boldsymbol{x}\right) \right], \mathbb{E}_{\boldsymbol{x}\in\text{test},c} \left[I\left(S_2|\boldsymbol{x}\right) \right], \ldots, \mathbb{E}_{\boldsymbol{x}\in\text{test},c} \left[I\left(S_d|\boldsymbol{x}\right) \right] \right]^\top \in \mathbb{R}^d$ denotes interaction effects of all m-order interactive concepts, which are averaged over different testing samples in the category c.

In addition, the similarity is defined as the following Jaccard similarity between non-negative elements of $I^{(m)}_{\text{train, c}}$ and $I^{(m)}_{\text{test, c}}$.

$$\text{sim}\left(I^{(m)}_{\text{train},c}, I^{(m)}_{\text{test},c} \right) = \text{Jaccard sim}\left(\tilde{I}^{(m)}_{\text{train},c}, \tilde{I}^{(m)}_{\text{test},c} \right) = \frac{\left\| \min\left(\tilde{I}^{(m)}_{\text{train },c}, \tilde{I}^{(m)}_{\text{test},c} \right) \right\|_1}{\left\| \max\left(\tilde{I}^{(m)}_{\text{train},c}, \tilde{I}^{(m)}_{\text{test},c} \right) \right\|_1}, \tag{19}$$

where the $2d$-dimensional vector $\tilde{I}^{(m)}_{\text{train},c} = \left[\left(\max\left(I^{(m)}_{\text{train},c}, 0 \right) \right)^\top, -\left(\min\left(I^{(m)}_{\text{train},c}, 0 \right) \right)^\top \right]^\top$ is constructed to contain non-negative elements of $I_{\text{train},c}$. Similarly, $\tilde{I}_{\text{test},c}$ is constructed based on $I_{\text{test},c}$ to contain non-negative elements. Thus, a high Jaccard similarity indicates that most m-order interactive concepts can be well-generalized from training samples to testing samples.

In this experiment, we train 8-layer MLPs for tabular datasets, including the Census dataset and the TV news dataset. All MLPs contain 100 neurons in each hidden layer. For each DNN, we compute the interaction effects of all interactive concepts encoded by the network. Then, we follow Zhou et al. (2023) to evaluate the average generalization ability $g^{(m)}$ of interactive concepts of different complexities (orders) on the above-mentioned DNNs. Table 3 shows that complex (high-order) interactive concepts usually have poorer generalization power than simple (low-order) interactive concepts.

Table 3. Comparison of the generalization ability $g^{(m)}$ of interactive concepts of different orders.

	order=1	order=3	order=5	order=7	order=9
MLP-8 on Census	0.7989	0.6203	0.5505	0.4436	0.3758
MLP-8 on TV news	0.8156	0.5854	0.3860	0.3322	0.1522

G. Proof of Theorems

G.1. Proof of Lemma 2.1 in the main paper

Lemma 2.1. *Given a neural network v and an arbitrary input sample $\boldsymbol{x}' \in \mathbb{R}^n$, the network output can be decomposed using the Taylor expansion $v(\boldsymbol{x}') = \sum_{S\subseteq N}\sum_{\boldsymbol{\pi}\in Q_S} U_{S,\boldsymbol{\pi}} \cdot J(S,\boldsymbol{\pi}|\boldsymbol{x}')$. In this way, according to Eq. (3) in the main paper, the interaction effect $I(S|\boldsymbol{x}')$ on the sample $\boldsymbol{x}'$ can be reformulated as*

$$I(S|\boldsymbol{x}') = \sum\nolimits_{\boldsymbol{\pi}\in Q_S} U_{S,\boldsymbol{\pi}} \cdot J(S,\boldsymbol{\pi}|\boldsymbol{x}'), \tag{20}$$

where $J(S,\boldsymbol{\pi}|\boldsymbol{x}') = \prod_{i\in S}\left(\text{sign}(x_i'-r_i)\cdot\frac{x_i'-r_i}{\tau}\right)^{\pi_i}$ denotes an expansion term of the degree $\boldsymbol{\pi}$, $\boldsymbol{\pi}\in Q_S = \{[\pi_1,\ldots,\pi_n]|\forall i\in S,\pi_i\in\mathbb{N}^+;\forall i\notin S,\pi_i=0\}$. $U_{S,\boldsymbol{\pi}}=\frac{\tau^m}{\prod_{i=1}^n\pi_i!}\frac{\partial^m v(\boldsymbol{x}_\emptyset)}{\partial x_1^{\pi_1}\cdots\partial x_n^{\pi_n}}\cdot\prod_{i\in S}[\text{sign}(x_i'-r_i)]^{\pi_i}$, $m=\sum_{i=1}^n\pi_i$.

Proof. Let us denote the function on the right of Eq. (20) by $\tilde{I}(S|\boldsymbol{x}')$, *i.e.*,

$$\tilde{I}(S|\boldsymbol{x}') = \sum\nolimits_{\pi\in Q_S} U_{S,\pi} J(S,\pi|\boldsymbol{x}') \tag{21}$$

We need to prove that for any arbitrary input sample $\forall \boldsymbol{x}'\in\mathbb{R}^n$, $\tilde{I}(S|\boldsymbol{x}') = I(S|\boldsymbol{x}')$.

Actually, it has been proven in Grabisch & Roubens (1999) and Ren et al. (2023a) that the Harsanyi dividend $I(S|\boldsymbol{x}')$ is the **unique** metric satisfying the faithfulness requirement mentioned in the main paper, *i.e.*, satisfying

$$\forall\, T\subseteq N,\ v(\boldsymbol{x}'_T) = \sum\nolimits_{S\in\Omega,S\subseteq T} I(S|\boldsymbol{x}'). \tag{22}$$

Thus, as long as we can prove that $\tilde{I}(S|\boldsymbol{x}')$ also satisfies the above faithfulness requirement, we can obtain $\tilde{I}(S|\boldsymbol{x}') = I(S|\boldsymbol{x}')$.

To this end, we only need to prove $\tilde{I}(S|\boldsymbol{x}')$ also satisfies the faithfulness requirement in Eq. (22). Specifically, given an input sample $\forall \boldsymbol{x}'\in\mathbb{R}^n$, let us consider the Taylor expansion of the network output $v(\boldsymbol{x}_T)$ of an arbitrarily masked sample $\boldsymbol{x}'_{\boldsymbol{T}}(T\subseteq N)$, which is expanded at $\boldsymbol{x}'_\emptyset=[r_1,\ldots,r_n]$. Then, we have

$$\forall\, T\subseteq N,\quad v(\boldsymbol{x}'_T) = \sum_{\pi_1=0}^{\infty}\sum_{\pi_2=0}^{\infty}\cdots\sum_{\pi_n=0}^{\infty}\frac{1}{\prod_{i=1}^n\pi_i!}\frac{\partial^m v(\boldsymbol{x}'_\emptyset)}{\partial x_1^{\pi_1}\cdots\partial x_n^{\pi_n}}\cdot\prod_{i=1}^n[(\boldsymbol{x}'_T)_i-r_i]^{\pi_i}, \tag{23}$$

where $\boldsymbol{\pi}\in\{[\pi_1,\ldots,\pi_n]|\forall i\in N,\pi_i\in\mathbb{N}\}$ denotes the degree vector of Taylor expansion terms, and $m=\sum_{i=1}^n\pi_i$. In addition, r_i denotes the reference value to mask the input variable x_i.

According to the definition of the masked sample $\boldsymbol{x}'_T$, we have that all variables in T keep unchanged and other variables are masked to the reference value. That is, $\forall\, i\in T, (\boldsymbol{x}'_T)_i = x_i; \forall\, i\notin T, (\boldsymbol{x}'_T)_i = r_i$. Hence, we obtain $\forall i\notin T, [(\boldsymbol{x}'_T)_i - r_i]^{\pi_i}=0$. Then, among all Taylor expansion terms, only terms corresponding to degrees $\boldsymbol{\pi}$ in the set $P=\{[\pi_1,\ldots,\pi_n]|\forall i\in T,\pi_i\in\mathbb{N};\forall i\notin T,\pi_i=0\}$ may not be zero. Therefore, Eq. (23) can be re-written as

$$\forall\, T\subseteq N,\quad v(\boldsymbol{x}'_T) = \sum_{\boldsymbol{\pi}\in P}\frac{1}{\prod_{i=1}^n\pi_i!}\frac{\partial^m v(\boldsymbol{x}'_\emptyset)}{\partial x_1^{\pi_1}\cdots\partial x_n^{\pi_n}}\cdot\prod_{i\in T}(x_i'-r_i)^{\pi_i}. \tag{24}$$

We find that the set P can be divided into multiple disjoint sets as follows, $P=\cup_{S\subseteq T}Q_S$, where $Q_S=\{[\pi_1,\ldots,\pi_n]|\forall i\in S,\pi_i\in\mathbb{N}^+;\forall i\notin S,\pi_i=0\}$. Then, we can derive that

$$\begin{aligned}\forall\, T\subseteq N,\quad v(\boldsymbol{x}'_T) &= \sum_{S\subseteq T}\sum_{\boldsymbol{\pi}\in Q_S}\frac{1}{\prod_{i=1}^n\pi_i!}\frac{\partial^m v(\boldsymbol{x}'_\emptyset)}{\partial x_1^{\pi_1}\cdots\partial x_n^{\pi_n}}\cdot\prod_{i\in S}(x_i'-r_i)^{\pi_i}\\ &= \sum_{S\subseteq T}\sum_{\boldsymbol{\pi}\in Q_S}\underbrace{\frac{\tau^m}{\prod_{i=1}^n\pi_i!}\frac{\partial^m v(\boldsymbol{x}'_\emptyset)}{\partial x_1^{\pi_1}\cdots\partial x_n^{\pi_n}}\prod_{i\in S}(\delta_i)^{\pi_i}}_{\text{termed } U_{S,\pi}}\cdot\underbrace{\prod_{i\in S}(\delta_i\frac{x_i'-r_i}{\tau})^{\pi_i}}_{\text{termed } J(S,\pi|\boldsymbol{x}')},\end{aligned} \tag{25}$$

where $\tau\in\mathbb{R}$ is a pre-defined constant and $\delta_i=\text{sign}(x_i-r_i)\in\{-1,1\}$ is a sign function and it satisfies $\prod_{i\in S}(\delta_i)^{2\pi_i}=1$. Then, Eq. (25) can be re-written as

$$\forall\, T\subseteq N,\ v(\boldsymbol{x}'_T) = \sum_{S\subseteq T}\sum_{\boldsymbol{\pi}\in Q_S}U_{S,\boldsymbol{\pi}}\cdot J(S,\boldsymbol{\pi}|\boldsymbol{x}') = \sum_{S\subseteq T}\tilde{I}(S|\boldsymbol{x}'). \tag{26}$$

Thus, $\tilde{I}(S|\boldsymbol{x}')$ satisfies the faithfulness requirement in Eq. (22) when $\Omega = 2^N$.

Therefore, Lemma 1 holds. □

G.2. Proof of Theorem 2.2 in the main paper

Theorem 2.2. *Let $\hat{\boldsymbol{\pi}}$ denote the lowest degree of the expansion terms of the interaction effect $I(S|\boldsymbol{x}')$, i.e., $\forall i \in S, \hat{\pi}_i = 1; \forall i \notin S, \hat{\pi}_i = 0$. Let us consider the interaction effect $I(S|\boldsymbol{x}')$ only containing the expansion term of the lowest degree, i.e., $I(S|\boldsymbol{x}') = U_{S,\hat{\boldsymbol{\pi}}} \cdot J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}')$. In this way, the mean and variance of the interaction effect $I(S|\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\epsilon})$ over different perturbations $\boldsymbol{\epsilon}$ are given as*

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})] &= U_{S,\hat{\boldsymbol{\pi}}}, \\ \mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})] &= U_{S,\hat{\boldsymbol{\pi}}}^2((1+(\sigma/\tau)^2)^{|S|}-1). \end{aligned} \tag{27}$$

Proof. If we only consider Taylor expansion term of the lowest degree, then $I(S|\boldsymbol{x}') = U_{S,\hat{\boldsymbol{\pi}}} \cdot J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}')$, where $J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}') = \prod_{i\in S} \mathrm{sign}(x_i' - r_i) \cdot \frac{x_i'-r_i}{\tau}$.

Let us add a Gaussian perturbation $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$ to the input sample $\boldsymbol{x}$. In this way, we have

$$\begin{aligned} I(S|\boldsymbol{x}+\boldsymbol{\epsilon}) &\approx U_{S,\hat{\boldsymbol{\pi}}} \cdot J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}+\boldsymbol{\epsilon}) \\ J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}+\boldsymbol{\epsilon}) &= \prod_{i\in S} \mathrm{sign}(x_i+\epsilon_i-r_i) \cdot \frac{x_i+\epsilon_i-r_i}{\tau} \\ &= \prod_{i\in S} \left(\mathrm{sign}(x_i+\epsilon_i-r_i) \cdot \frac{x_i-r_i}{\tau} + \mathrm{sign}(x_i+\epsilon_i-r_i) \cdot \frac{\epsilon_i}{\tau} \right) \end{aligned} \tag{28}$$

According to the setting of the reference value in Section 2.3, we have $\forall i \in S, x_i - r_i \in \{-\tau, \tau\}$. Also in Section 2.3, we have assumed that the variance of the perturbation $\boldsymbol{\epsilon}$ is small, so that we can ignore the extremely low probability that the perturbation is large such that $|\epsilon_i| \geq \tau$. In this way, we have $\mathrm{sign}(x_i+\epsilon_i-r_i) = \mathrm{sign}(x_i-r_i)$, and we can obtain

$$\begin{aligned} J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}+\boldsymbol{\epsilon}) &= \prod_{i\in S} \left(\mathrm{sign}(x_i-r_i) \cdot \frac{x_i-r_i}{\tau} + \mathrm{sign}(x_i-r_i) \cdot \frac{\epsilon_i}{\tau} \right) \\ &= \prod_{i\in S} \left(1 + \mathrm{sign}(x_i-r_i) \cdot \frac{\epsilon_i}{\tau} \right) \end{aligned} \tag{29}$$

$$\begin{aligned} \Rightarrow \mathbb{E}_{\boldsymbol{\epsilon}}[J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathbb{E}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S} \left(1 + \mathrm{sign}(x_i-r_i) \cdot \frac{\epsilon_i}{\tau} \right) \right] \\ \mathrm{Var}_{\boldsymbol{\epsilon}}[J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathrm{Var}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S} \left(1 + \mathrm{sign}(x_i-r_i) \cdot \frac{\epsilon_i}{\tau} \right) \right] \end{aligned} \tag{30}$$

Since $\mathrm{sign}(x_i - r_i) \in \{-1, 1\}$, we have $1 + \mathrm{sign}(x_i - r_i) \cdot \frac{\epsilon_i}{\tau} \sim \mathcal{N}(1, (\sigma/\tau)^2), \forall i \in S$.

Proposition G.1. *If random variables $X_1, X_2, \cdots, X_k$ are independent of each other, then $\mathbb{E}[X_1X_2\cdots X_k] = \prod_{i=1}^k \mathbb{E}[X_i]$, and $\mathrm{Var}[X_1X_2\cdots X_k] = \prod_{i=1}^k (\mathbb{E}[X_i]^2 + \mathrm{Var}[X_i]) - \prod_{i=1}^k \mathbb{E}[X_i]^2$.*

According to the above proposition, we have

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\epsilon}}[J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \prod_{i\in S} 1 = 1 \\ \mathrm{Var}_{\boldsymbol{\epsilon}}[J(S, \hat{\boldsymbol{\pi}}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \prod_{i\in S} \left(1^2 + (\sigma/\tau)^2 \right) - \prod_{i\in S} 1^2 \\ &= \left(1 + (\sigma/\tau)^2 \right)^{|S|} - 1 \end{aligned} \tag{31}$$

Therefore,

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathbb{E}_{\boldsymbol{\epsilon}}[U_{S,\hat{\boldsymbol{\pi}}}\cdot J(S,\hat{\boldsymbol{\pi}}|\boldsymbol{x}+\boldsymbol{\epsilon})] = U_{S,\hat{\boldsymbol{\pi}}}\\ \mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathrm{Var}_{\boldsymbol{\epsilon}}[U_{S,\hat{\boldsymbol{\pi}}}\cdot J(S,\hat{\boldsymbol{\pi}}|\boldsymbol{x}+\boldsymbol{\epsilon})] = U_{S,\hat{\boldsymbol{\pi}}}^2\left(\left(1+(\sigma/\tau)^2\right)^{|S|}-1\right)\end{aligned} \tag{32}$$

□

G.3. Proof of Theorem 2.3 in the main paper

Theorem 2.3. *Let $\boldsymbol{\pi}\in Q_S=\{[\pi_1,\ldots,\pi_n]|\forall i\in S,\pi_i\in\mathbb{N}^+;\forall i\notin S,\pi_i=0\}$ denote an arbitrary degree. Then, the mean and the variance of $J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})$ over perturbations $\boldsymbol{\epsilon}$ are*

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathbb{E}_{\boldsymbol{\epsilon}}[\textstyle\prod_{i\in S}(1+\frac{\epsilon_i}{\tau})^{\pi_i}],\\ \mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathrm{Var}_{\boldsymbol{\epsilon}}[\textstyle\prod_{i\in S}(1+\frac{\epsilon_i}{\tau})^{\pi_i}]\end{aligned} \tag{33}$$

Proof. According to Lemma 2.1, given an arbitrary input sample $\boldsymbol{x}'$, we have

$$J(S,\boldsymbol{\pi}|\boldsymbol{x}') = \prod\nolimits_{i\in S}\left(\mathrm{sign}(x_i'-r_i)\cdot\frac{x_i'-r_i}{\tau}\right)^{\pi_i} \tag{34}$$

Let us add a Gaussian perturbation $\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I})$ to the input sample $\boldsymbol{x}$. In this way, we have

$$\begin{aligned}J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon}) &= \prod_{i\in S}\left(\mathrm{sign}(x_i+\epsilon_i-r_i)\cdot\frac{x_i+\epsilon_i-r_i}{\tau}\right)^{\pi_i}\\ &= \prod_{i\in S}\left(\mathrm{sign}(x_i+\epsilon_i-r_i)\cdot\frac{x_i-r_i}{\tau}+\mathrm{sign}(x_i+\epsilon_i-r_i)\cdot\frac{\epsilon_i}{\tau}\right)^{\pi_i}\end{aligned} \tag{35}$$

According to the setting of the reference value in Section 2.3, $\forall i\in S, x_i-r_i\in\{-\tau,\tau\}$. Also, in Section 2.3, we have assumed that the variance of the perturbation $\boldsymbol{\epsilon}$ is small, so that we can ignore the extremely low probability that the perturbation is large such that $|\epsilon_i|\geq\tau$. In this way, $\mathrm{sign}(x_i+\epsilon_i-r_i)=\mathrm{sign}(x_i-r_i)$, and we can obtain

$$\begin{aligned}J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon}) &= \prod_{i\in S}\left(\mathrm{sign}(x_i-r_i)\cdot\frac{x_i-r_i}{\tau}+\mathrm{sign}(x_i-r_i)\cdot\frac{\epsilon_i}{\tau}\right)^{\pi_i}\\ &= \prod_{i\in S}\left(1+\mathrm{sign}(x_i-r_i)\cdot\frac{\epsilon_i}{\tau}\right)^{\pi_i}\end{aligned} \tag{36}$$

$$\begin{aligned}\Rightarrow\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathbb{E}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S}\left(1+\mathrm{sign}(x_i-r_i)\cdot\frac{\epsilon_i}{\tau}\right)^{\pi_i}\right]\\ \mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathrm{Var}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S}\left(1+\mathrm{sign}(x_i-r_i)\cdot\frac{\epsilon_i}{\tau}\right)^{\pi_i}\right]\end{aligned} \tag{37}$$

Since $\forall i\in S$, ϵ_i is independent of each other, according to Proposition G.1 and Eq. (37), we have

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \prod_{i\in S}\mathbb{E}_{\epsilon_i}\left[\left(1+\mathrm{sign}(x_i-r_i)\cdot\frac{\epsilon_i}{\tau}\right)^{\pi_i}\right]\\ \mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \prod_{i\in S}\mathbb{E}_{\epsilon_i}\left[\left(1+\mathrm{sign}(x_i-r_i)\cdot\frac{\epsilon_i}{\tau}\right)^{2\pi_i}\right]-\prod_{i\in S}\left(\mathbb{E}_{\epsilon_i}\left[\left(1+\mathrm{sign}(x_i-r_i)\cdot\frac{\epsilon_i}{\tau}\right)^{\pi_i}\right]\right)^2\end{aligned} \tag{38}$$

Since $\mathrm{sign}(x_i-r_i)\in\{-1,1\}$, we have $\mathbb{E}_{\epsilon_i}\left[\left(1+\mathrm{sign}(x_i-r_i)\cdot\frac{\epsilon_i}{\tau}\right)^k\right]=\mathbb{E}_{\epsilon_i}\left[\left(1+\frac{\epsilon_i}{\tau}\right)^k\right],\forall k\in\mathbb{N}^+$. Therefore, we obtain

$$
\begin{aligned}
\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \prod_{i\in S}\mathbb{E}_{\epsilon_i}\left[\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi_i}\right] \\
&= \mathbb{E}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi_i}\right] \\
\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \prod_{i\in S}\mathbb{E}_{\epsilon_i}\left[\left(1+\frac{\epsilon_i}{\tau}\right)^{2\pi_i}\right] - \prod_{i\in S}\left(\mathbb{E}_{\epsilon_i}\left[\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi_i}\right]\right)^2 \\
&= \mathrm{Var}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi_i}\right].
\end{aligned}
$$

□

G.4. Proof of Theorem 2.4 in the main paper

Theorem 2.4. *Let S and S' be two interactive concepts, such that $S \subsetneq S'$. Let us consider expansion terms $J(S,\boldsymbol{\pi})$ and $J(S',\boldsymbol{\pi}')$, where the term $J(S',\boldsymbol{\pi}')$ is extended from the term $J(S,\boldsymbol{\pi})$ with $\boldsymbol{\pi} \prec \boldsymbol{\pi}'$. I.e., (1) $\forall i \in S', \pi'_i \in \mathbb{N}^+$; otherwise, $\pi'_i = 0$. (2) Given $\boldsymbol{\pi}'$, $\forall j \in S, \pi_j = \pi'_j$; otherwise, $\pi_j = 0$. Then, we have*

$$
\begin{aligned}
\frac{\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]}{\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]} &> \textstyle\prod_{i\in S'\setminus S}\mathbb{E}^2_{\epsilon_i}[(1+\frac{\epsilon_i}{\tau})^{\pi'_i}], \\
\frac{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]}{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]} &< \frac{1}{\prod_{i\in S'\setminus S}\mathbb{E}_{\epsilon_i}[(1+\frac{\epsilon_i}{\tau})^{\pi'_i}]},
\end{aligned}
\tag{39}
$$

and we can also obtain $\mathbb{E}_{\epsilon_i}[(1+\frac{\epsilon_i}{\tau})^{\pi'_i}] \geq 1$.

Proof. According to Theorem 2.3, we have

$$
\begin{aligned}
\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathrm{Var}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S'}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\right] \\
&= \mathrm{Var}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\prod_{i\in S'\setminus S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\right] \quad //S\subsetneq S' \\
&= \mathrm{Var}_{\boldsymbol{\epsilon}}\left[\underbrace{\prod_{i\in S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi_i}}_{A}\underbrace{\prod_{i\in S'\setminus S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}}_{B}\right] \quad //\forall i\in S, \pi'_i=\pi_i \\
&= \mathrm{Var}_{\boldsymbol{\epsilon}}[AB] \\
&= (\mathbb{E}^2_{\boldsymbol{\epsilon}}[A]+\mathrm{Var}_{\boldsymbol{\epsilon}}[A])(\mathbb{E}^2_{\boldsymbol{\epsilon}}[B]+\mathrm{Var}_{\boldsymbol{\epsilon}}[B]) - \mathbb{E}^2_{\boldsymbol{\epsilon}}[A]\mathbb{E}^2_{\boldsymbol{\epsilon}}[B] \\
&\quad //A \text{ and } B \text{ are independent; Proposition G.1} \\
&= \mathbb{E}^2_{\boldsymbol{\epsilon}}[A]\mathrm{Var}_{\boldsymbol{\epsilon}}[B]+\mathbb{E}^2_{\boldsymbol{\epsilon}}[B]\mathrm{Var}_{\boldsymbol{\epsilon}}[A]+\mathrm{Var}_{\boldsymbol{\epsilon}}[A]\mathrm{Var}_{\boldsymbol{\epsilon}}[B] \\
&> \mathbb{E}^2_{\boldsymbol{\epsilon}}[B]\mathrm{Var}_{\boldsymbol{\epsilon}}[A]+\mathrm{Var}_{\boldsymbol{\epsilon}}[A]\mathrm{Var}_{\boldsymbol{\epsilon}}[B]
\end{aligned}
\tag{40}
$$

Therefore, we can prove the first equality as follows.

$$\begin{aligned}\frac{\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]}{\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]} &= \frac{\mathrm{Var}_{\boldsymbol{\epsilon}}[AB]}{\mathrm{Var}_{\boldsymbol{\epsilon}}[A]} \\ &> \mathbb{E}^2_{\boldsymbol{\epsilon}}[B] + \mathrm{Var}_{\boldsymbol{\epsilon}}[B] \\ &> \mathbb{E}^2_{\boldsymbol{\epsilon}}[B] \\ &= \mathbb{E}^2_{\boldsymbol{\epsilon}}\left[\prod_{i\in S'\setminus S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\right] \\ &= \prod_{i\in S'\setminus S}\mathbb{E}^2_{\epsilon_i}\left[\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\right] \\ &\quad //\epsilon_i \text{ is independent of each other; Proposition G.1}\end{aligned} \tag{41}$$

Furthermore, we have

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})] &= \mathbb{E}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S'}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\right] \\ &= \mathbb{E}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\prod_{i\in S'\setminus S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\right] \quad //S\subsetneq S' \\ &= \mathbb{E}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi_i}\prod_{i\in S'\setminus S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\right] \quad //\forall i\in S,\pi'_i=\pi_i \\ &= \mathbb{E}_{\boldsymbol{\epsilon}}[AB]\end{aligned} \tag{42}$$

and also

$$\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})] = \mathbb{E}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi_i}\right] = \mathbb{E}_{\boldsymbol{\epsilon}}[A]. \tag{43}$$

Therefore, we have

$$\frac{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]}{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]} = \frac{\mathbb{E}_{\boldsymbol{\epsilon}}[AB]}{\mathbb{E}_{\boldsymbol{\epsilon}}[A]} = \mathbb{E}_{\boldsymbol{\epsilon}}[B]. \tag{44}$$

Then, we can prove the second inequality as follows.

$$\begin{aligned}&\frac{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S',\boldsymbol{\pi}'|\boldsymbol{x}+\boldsymbol{\epsilon})]}{\mathbb{E}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[J(S,\boldsymbol{\pi}|\boldsymbol{x}+\boldsymbol{\epsilon})]} \\ =&\frac{\mathbb{E}_{\boldsymbol{\epsilon}}[B]}{\mathrm{Var}_{\boldsymbol{\epsilon}}[AB]/\mathrm{Var}_{\boldsymbol{\epsilon}}[A]} \\ <&\frac{\mathbb{E}_{\boldsymbol{\epsilon}}[B]}{\mathbb{E}^2_{\boldsymbol{\epsilon}}[B]} \\ =&\frac{1}{\mathbb{E}_{\boldsymbol{\epsilon}}[B]} \\ =&\frac{1}{\mathbb{E}_{\boldsymbol{\epsilon}}\left[\prod_{i\in S'\setminus S}\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\right]} \\ =&\frac{1}{\prod_{i\in S'\setminus S}\mathbb{E}_{\epsilon_i}\left[\left(1+\frac{\epsilon_i}{\tau}\right)^{\pi'_i}\right]}\end{aligned} \tag{45}$$

Moreover, we can prove that $\mathbb{E}_{\epsilon_i}[(1+\frac{\epsilon_i}{\tau})^k]\geq 1, \forall k\in\mathbb{N}^+$, *i.e.*, $\mathbb{E}[X^k]\geq 1$, where $X\sim\mathcal{N}(1,(\sigma/\tau)^2)$.

For a random variable following a Gaussian distribution $\tilde{X} \sim \mathcal{N}(\tilde{\mu}, \tilde{\sigma}^2)$, Willink (2005) proved the following property:

$$\mathbb{E}\left[\tilde{X}^{k+1}\right] = \tilde{\mu}\mathbb{E}\left[\tilde{X}^{k}\right] + k\tilde{\sigma}^2\mathbb{E}\left[\tilde{X}^{k-1}\right] \tag{46}$$

Now let us consider $X \sim \mathcal{N}(1, (\sigma/\tau)^2)$. We have $\mathbb{E}\left[X^{k+1}\right] = \mathbb{E}\left[X^{k}\right] + k(\sigma/\tau)^2\mathbb{E}\left[X^{k-1}\right]$. By induction, it is easy to prove that $\mathbb{E}[X^k] \geq \mathbb{E}[X] = 1$. □

G.5. Proof of Theorem 2.5 in the main paper

Theorem 2.5. *Given two random interactive concepts S and S', we can roughly assume that $C_S(\boldsymbol{x}+\boldsymbol{\epsilon})$ is independent of $C_{S'}(\boldsymbol{x}+\boldsymbol{\epsilon})$, because the two concepts S and S' usually have little overlap in most cases. Let $\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]$ and $\mathrm{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]$ denote the mean and the variance of $C_S(\boldsymbol{x}+\boldsymbol{\epsilon})$ w.r.t. $\boldsymbol{\epsilon}$, respectively. Then, the solution to Eq. (15) in the main paper satisfies the following property:*

$$\forall\, S \in \Omega, \quad |U_S^*| \propto |\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]| \tag{47}$$

Proof. Let $p = |\Omega|$. Let $\boldsymbol{C}(\boldsymbol{x}+\boldsymbol{\epsilon}) = [C_{S_1}(\boldsymbol{x}+\boldsymbol{\epsilon}), \cdots, C_{S_p}(\boldsymbol{x}+\boldsymbol{\epsilon})]^\top$ denote the vector of all $C_S(\boldsymbol{x}+\boldsymbol{\epsilon}), S \in \Omega$, and let $\boldsymbol{U} = [U_{S_1}, \cdots, U_{S_p}]^\top$ denote the vector of all coefficients $U_S, S \in \Omega$. To further simplify the notation, we simply use $\boldsymbol{C}$ to denote the random vector $\boldsymbol{C}(\boldsymbol{x}+\boldsymbol{\epsilon})$. Besides, since we assume that each dimension of the vector $\boldsymbol{C}(\boldsymbol{x}+\boldsymbol{\epsilon})$ is independent of each other, we can use $\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}] = [\alpha_1, \cdots, \alpha_p]^\top \in \mathbb{R}^p$ and $\mathrm{Var}_{\boldsymbol{\epsilon}}[\boldsymbol{C}] = \mathrm{diag}(\beta_1^2, \cdots, \beta_p^2) \in \mathbb{R}^{p\times p}$ to denote the mean vector and covariance matrix of the random vector $\boldsymbol{C}(\boldsymbol{x}+\boldsymbol{\epsilon})$, respectively. We prove this theorem in three steps.

Step 1. We first prove that the optimal solution to Eq. (15) in the main paper is given by

$$\forall 1 \leq i \leq p, \quad U_{S_i}^* = \frac{1}{\det \boldsymbol{M}} \det(\boldsymbol{M}_1, \cdots, \boldsymbol{M}_{i-1}, \boldsymbol{\rho}, \boldsymbol{M}_{i+1}, \cdots, \boldsymbol{M}_p) \tag{48}$$

where $\boldsymbol{M} = \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]^\top + \mathrm{Var}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]$, $\boldsymbol{\rho} = y^*\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]$, and $\boldsymbol{M}_j$ denotes the j-th column of the matrix $\boldsymbol{M}$.

We can rewrite the objective function in Eq. (15) in the main paper as

$$\min_{\boldsymbol{U}} \mathbb{E}_{\boldsymbol{\epsilon}}[(y^* - \boldsymbol{U}^\top \boldsymbol{C}(\boldsymbol{x}+\boldsymbol{\epsilon}))^2] \tag{49}$$

To minimize the loss $L = \mathbb{E}_{\boldsymbol{\epsilon}}[(y^* - \boldsymbol{U}^\top \boldsymbol{C})^2]$, we set the gradient of the loss *w.r.t* $\boldsymbol{U}$ to zero, *i.e.*,

$$\begin{aligned}
\nabla_{\boldsymbol{U}} L &= \mathbb{E}_{\boldsymbol{\epsilon}}[2\boldsymbol{C}(\boldsymbol{U}^\top \boldsymbol{C} - y^*)] \\
&= 2\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}\boldsymbol{C}^\top \boldsymbol{U} - y^*\boldsymbol{C}] \\
&= 2\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}\boldsymbol{C}^\top]\boldsymbol{U} - 2y^*\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}] \\
&= 2(\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]^\top + \mathrm{Var}_{\boldsymbol{\epsilon}}[\boldsymbol{C}])\boldsymbol{U} - 2y^*\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}] = 0
\end{aligned} \tag{50}$$

$$\Rightarrow (\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]^\top + \mathrm{Var}_{\boldsymbol{\epsilon}}[\boldsymbol{C}])\boldsymbol{U} = y^*\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}] \tag{51}$$

Let $\boldsymbol{M} = \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]^\top + \mathrm{Var}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]$, and $\boldsymbol{\rho} = y^*\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]$. By Cramer's rule, we can obtain the solution to Eq. (51):

$$\forall 1 \leq i \leq p, \quad U_{S_i}^* = \frac{1}{\det \boldsymbol{M}} \det(\boldsymbol{M}_1, \cdots, \boldsymbol{M}_{i-1}, \boldsymbol{\rho}, \boldsymbol{M}_{i+1}, \cdots, \boldsymbol{M}_p)$$

where $\boldsymbol{M}_j$ denotes the j-th column of the matrix $\boldsymbol{M}$.

Step 2. We prove that for the optimal solution $\boldsymbol{U}^*$, we have

$$\forall 1 \leq i, j \leq p, \quad \frac{|U_{S_i}^*|}{|U_{S_j}^*|} = \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[C_{S_i}(\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[C_{S_i}(\boldsymbol{x}+\boldsymbol{\epsilon})]|}{|\mathbb{E}_{\boldsymbol{\epsilon}}[C_{S_j}(\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[C_{S_j}(\boldsymbol{x}+\boldsymbol{\epsilon})]|} \tag{52}$$

Since $\boldsymbol{M} = \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]^\top + \mathrm{Var}_{\boldsymbol{\epsilon}}[\boldsymbol{C}]$, we can obtain the j-th column of $\boldsymbol{M}$ as

$$\boldsymbol{M}_j = \alpha_j \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}] + \boldsymbol{V}_j \tag{53}$$

where $\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}] = [\alpha_1, \cdots, \alpha_p]^\top$, and $\boldsymbol{V}_j = [0, \cdots, \beta_j^2, \cdots, 0]^\top$.

According to the conclusion in Step 1, we have

$$|U_{S_i}^*| = |\frac{1}{\det \boldsymbol{M}}| \cdot |\det(\boldsymbol{M}_1, \cdots, \boldsymbol{M}_{i-1}, \boldsymbol{\rho}, \boldsymbol{M}_{i+1}, \cdots, \boldsymbol{M}_{j-1}, \boldsymbol{M}_j, \boldsymbol{M}_{j+1}, \cdots, \boldsymbol{M}_p)| \tag{54}$$

$$|U_{S_j}^*| = |\frac{1}{\det \boldsymbol{M}}| \cdot |\det(\boldsymbol{M}_1, \cdots, \boldsymbol{M}_{i-1}, \boldsymbol{M}_i, \boldsymbol{M}_{i+1}, \cdots, \boldsymbol{M}_{j-1}, \boldsymbol{\rho}, \boldsymbol{M}_{j+1}, \cdots, \boldsymbol{M}_p)| \tag{55}$$

We know that exchanging the rows or columns of a matrix only changes the sign of the determinant of the matrix, but does not change the absolute value of the determinant. Therefore, we have

$$\begin{aligned}
|U_{S_i}^*| &= |\frac{1}{\det \boldsymbol{M}}| \cdot |\det(\boldsymbol{M}_j, \boldsymbol{\rho}, \boldsymbol{M}_1, \cdots, \boldsymbol{M}_{i-1}, \boldsymbol{M}_{i+1}, \cdots, \boldsymbol{M}_{j-1}, \boldsymbol{M}_{j+1}, \cdots, \boldsymbol{M}_p)| \\
&= |\frac{1}{\det \boldsymbol{M}}| \cdot |\det(\boldsymbol{M}_j, \boldsymbol{\rho}, \boldsymbol{M}_{\text{others}})| \quad //\text{Let } \boldsymbol{M}_{\text{others}} \text{ denote the third to the last column} \\
&= |\frac{1}{\det \boldsymbol{M}}| \cdot |\det(\alpha_j \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}] + \boldsymbol{V}_j, y^* \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}], \boldsymbol{M}_{\text{others}})| \quad //\text{Eq. (53)} \\
&= |\frac{1}{\det \boldsymbol{M}}| \cdot |\underbrace{\det(\alpha_j \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}], y^* \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}], \boldsymbol{M}_{\text{others}})}_{=0} + \det(\boldsymbol{V}_j, y^* \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}], \boldsymbol{M}_{\text{others}})| \\
&\quad //\text{The determinant is 0 if two columns are linearly dependent} \\
&= |\frac{1}{\det \boldsymbol{M}}| \cdot |\det(\boldsymbol{V}_j, y^* \mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{C}], \boldsymbol{M}_{\text{others}})| \\
&= |\frac{1}{\det \boldsymbol{M}}| \cdot |\det \begin{bmatrix} 0 & y^*\alpha_1 & \alpha_1\alpha_1 + \beta_1^2 & \cdots & \alpha_1\alpha_p \\ \vdots & & & & \\ 0 & y^*\alpha_i & \alpha_i\alpha_1 & \cdots & \alpha_i\alpha_p \\ \vdots & & & & \\ \beta_j^2 & y^*\alpha_j & \alpha_j\alpha_1 & \cdots & \alpha_j\alpha_p \\ \vdots & & & & \\ 0 & y^*\alpha_p & \alpha_p\alpha_1 & \cdots & \alpha_p\alpha_p + \beta_p^2 \end{bmatrix}| \\
&= |\frac{1}{\det \boldsymbol{M}}| \cdot |det \begin{bmatrix} \beta_j^2 & y^*\alpha_j & \alpha_j\alpha_1 & \cdots & \alpha_j\alpha_p \\ 0 & y^*\alpha_i & \alpha_i\alpha_1 & \cdots & \alpha_i\alpha_p \\ 0 & y^*\alpha_1 & \alpha_1\alpha_1 + \beta_1^2 & \cdots & \alpha_1\alpha_p \\ \vdots & & & & \\ 0 & y^*\alpha_p & \alpha_p\alpha_1 & \cdots & \alpha_p\alpha_p + \beta_p^2 \end{bmatrix}| \quad //\text{Exchange rows} \\
&= |\frac{\alpha_i}{\det \boldsymbol{M}}| \cdot |\det \begin{bmatrix} \beta_j^2 & y^*\alpha_j & \alpha_j\alpha_1 & \cdots & \alpha_j\alpha_p \\ 0 & y^* & \alpha_1 & \cdots & \alpha_p \\ 0 & y^*\alpha_1 & \alpha_1\alpha_1 + \beta_1^2 & \cdots & \alpha_1\alpha_p \\ \vdots & & & & \\ 0 & y^*\alpha_p & \alpha_p\alpha_1 & \cdots & \alpha_p\alpha_p + \beta_p^2 \end{bmatrix}| \quad //\text{Extract out } \alpha_i \\
&= |\frac{\alpha_i \beta_j^2}{\det \boldsymbol{M}}| \cdot |\det \boldsymbol{M}'|,
\end{aligned} \tag{56}$$

where

$$\boldsymbol{M}' = \begin{bmatrix} y^* & \alpha_1 & \cdots & \alpha_p \\ y^*\alpha_1 & \alpha_1\alpha_1 + \beta_1^2 & \cdots & \alpha_1\alpha_p \\ \cdots & & & \\ y^*\alpha_p & \alpha_p\alpha_1 & \cdots & \alpha_p\alpha_p + \beta_p^2 \end{bmatrix}. \tag{57}$$

Similarly, we can prove that

$$|U_{S_j}^*| = |\frac{\alpha_j \beta_i^2}{\det \boldsymbol{M}}| \cdot |\det \boldsymbol{M}'|. \tag{58}$$

Therefore, we have

$$\forall 1 \leq i, j \leq p, \quad \frac{|U_{S_i}^*|}{|U_{S_j}^*|} = \frac{|\alpha_i/\beta_i^2|}{|\alpha_j/\beta_j^2|} = \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[C_{S_i}(\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[C_{S_i}(\boldsymbol{x}+\boldsymbol{\epsilon})]|}{|\mathbb{E}_{\boldsymbol{\epsilon}}[C_{S_j}(\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[C_{S_j}(\boldsymbol{x}+\boldsymbol{\epsilon})]|}.$$

Step 3. Based on Step 2, we can directly prove that for the optimal solution U^*, we have

$$\forall S \in \Omega, \quad |U_S^*| \propto |\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]/\mathrm{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]| \tag{59}$$

□

G.6. Proof of Theorem 2.6 in the main paper

Theorem 2.6. *Let $A^{min} = \min_S |U_S|$ and $A^{max} = \max_S |U_S|$ denote the lower bound and the upper bound of $|U_S|$ over all interactive concepts S. Then, for any $S \subseteq N$, we have*

$$A^{min} \cdot \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]} \leq \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\mathrm{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]} \leq A^{max} \cdot \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]} \tag{60}$$

Proof. According to Eq. (12) in the main paper, we have $I(S|\boldsymbol{x}') = U_S \cdot C_S(\boldsymbol{x}')$. Hence, we have

$$|\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]| = |U_S| \cdot |\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]|, \quad \mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})] = U_S^2 \cdot \mathrm{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})],$$

Therefore,

$$\frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\mathrm{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]} = |U_S| \cdot \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]}$$

Then, let $A_S^{\min} = \min_S |U_S|$ and $A_S^{\max} = \max_S |U_S|$ denote the lower bound and the upper bound of the absolute value $|U_S|$ over all interactive concepts S, we have

$$A_S^{\min} \cdot \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]} \leq \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\mathrm{Var}_{\boldsymbol{\epsilon}}[C_S(\boldsymbol{x}+\boldsymbol{\epsilon})]} \leq A_S^{\max} \cdot \frac{|\mathbb{E}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]|}{\mathrm{Var}_{\boldsymbol{\epsilon}}[I(S|\boldsymbol{x}+\boldsymbol{\epsilon})]}$$

□

H. Experimental details

Training settings. We trained standard DNNs and BNNs with the same architectures on two image datasets and two tabular datasets. For image datasets, we trained standard DNNs and BNNs with two architectures. On the MNIST dataset, we trained a standard DNN and a BNN with the 5-layer MLP architecture. On the CIFAR-10 dataset, we trained a standard DNN and a BNN with the LeNet architecture. On the two tabular datasets, including the UCI TV news dataset (termed *TV news*) and the UCI census income dataset (termed *census*), we trained standard DNNs and BNNs with the 8-layer MLP architecture. All MLPs contained 100 neurons in each hidden layer. For the training of BNNs, the prior distribution of network weights was set to $\mathcal{N}(\boldsymbol{W}; \mathbf{0}, \boldsymbol{I})$, and the number of Monte Carlo sampling of network weights was set to 1. All standard DNNs and BNNs were trained using the Adam optimizer (Kingma & Ba, 2015) with learning rate 0.001. The 5-layer MLPs (standard DNN and BNN) on the MNIST dataset was trained for 50 epochs. The LeNet (standard DNN and BNN) on the CIFAR-10 dataset was trained for 300 epochs. The 8-layer MLPs (standard DNN and BNN) on tabular datasets were trained for 200 epochs.

Implementation details for the calculation of $I(S)$. Since the computational cost of $I(S)$ was intolerable for image datasets, we applied a sampling-based approximation method to calculate U_S. For the CIFAR-10 dataset (32×32 pixels on each image), we uniformly split each input image into 8×8 patches. Furthermore, we random sampled 12 patches from the central 6×6 region (*i.e.*, we did not sample patches that were on the edges of an image), and considered these patches as input variables for each image. The remaining 52 patches were set to the reference value. Similarly, for the MNIST dataset (28×28 pixels on each image), we uniformly split each input image into 7×7 patches, and randomly sampled 12 patches from the central 5×5 region.

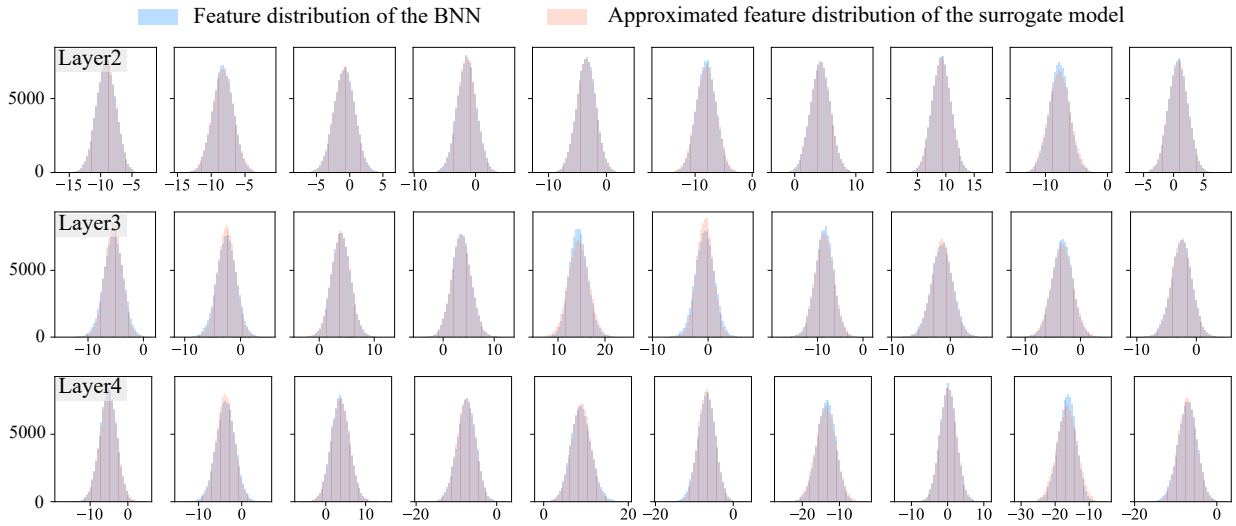

Figure 6. More visualization results of MLP-5 on the MNIST dataset.

Implementation details of the reference value. Let $\mathbb{E}_{\boldsymbol{x}}[x_i]$ denote the mean value of the i-th input dimension over all input samples in the dataset. Then, given an input sample $\boldsymbol{x}$, the reference value is set as follows.

$$r_i = \begin{cases} x_i - \tau, & x_i > \mathbb{E}_{\boldsymbol{x}}[x_i] \\ x_i + \tau, & x_i < \mathbb{E}_{\boldsymbol{x}}[x_i] \end{cases}$$

where $\tau \in \mathbb{R}$ is a constant. We set $\tau = 0.5$ on all datasets (including the TV news dataset, the Census dataset, the MNIST dataset, and the CIFAR-10 dataset). In our experiments, we assume that input samples have been normalized as follows. First, we subtract the mean value of each input dimension over the whole dataset from the input sample. Second, we divide each dimension of the input sample by the standard deviation of this input dimension over the whole dataset. In this way, input samples have zero mean and unit variance on each dimension over the whole dataset, *i.e.*, $\forall i \in N, \mathbb{E}_{\boldsymbol{x}}[x_i] = 0, \text{Var}_{\boldsymbol{x}}[x_i] = 1$.

Implementation details of the experiment in Section 2.2 of the main paper. In Section 2.2 of the main paper, we minimized the KL divergence between the feature distribution in the surrogate DNN model and the feature distribution in the BNN. The feature distributions in the surrogate DNN model and in the BNN were not Gaussian distributions. Therefore, the KL divergence between the feature distributions did not have a close-form formula. To facilitate the optimization, we simply used two Gaussian distributions to approximate the feature distributions in the surrogate DNN model and in the BNN, and optimized the KL divergence between the two Gaussian distributions. Besides, we did not consider the dependency between different feature dimensions to simplify the computation.

I. More visualization results for experiments in Section 2.2 of the main paper

In this subsection, we provided more visualization results to show that the feature distribution of the surrogate DNN model could well approximate the feature distribution of the BNN.

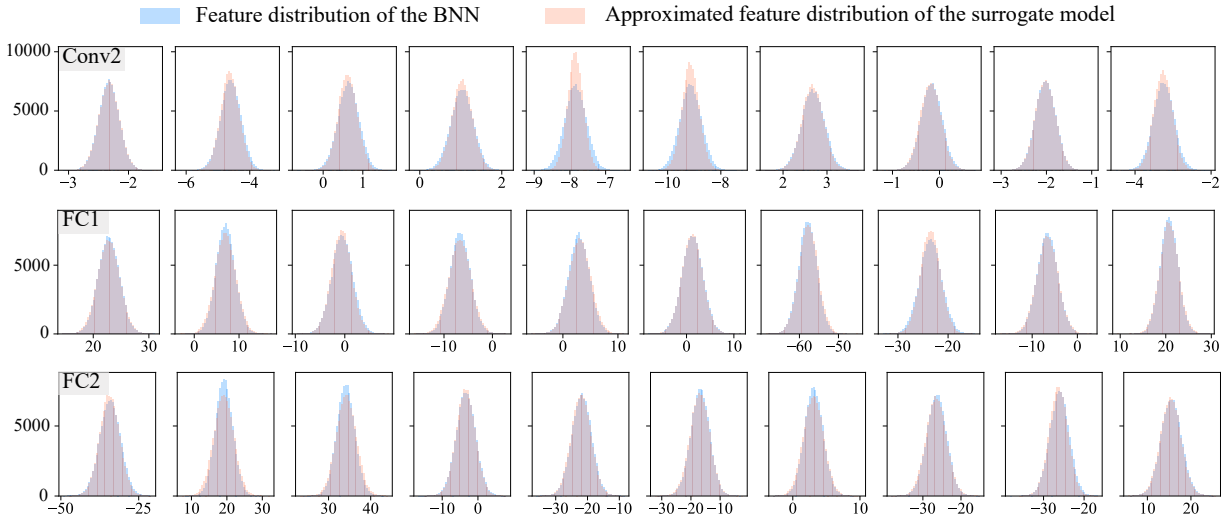

Figure 7. More visualization results of LeNet on the CIFAR-10 dataset.

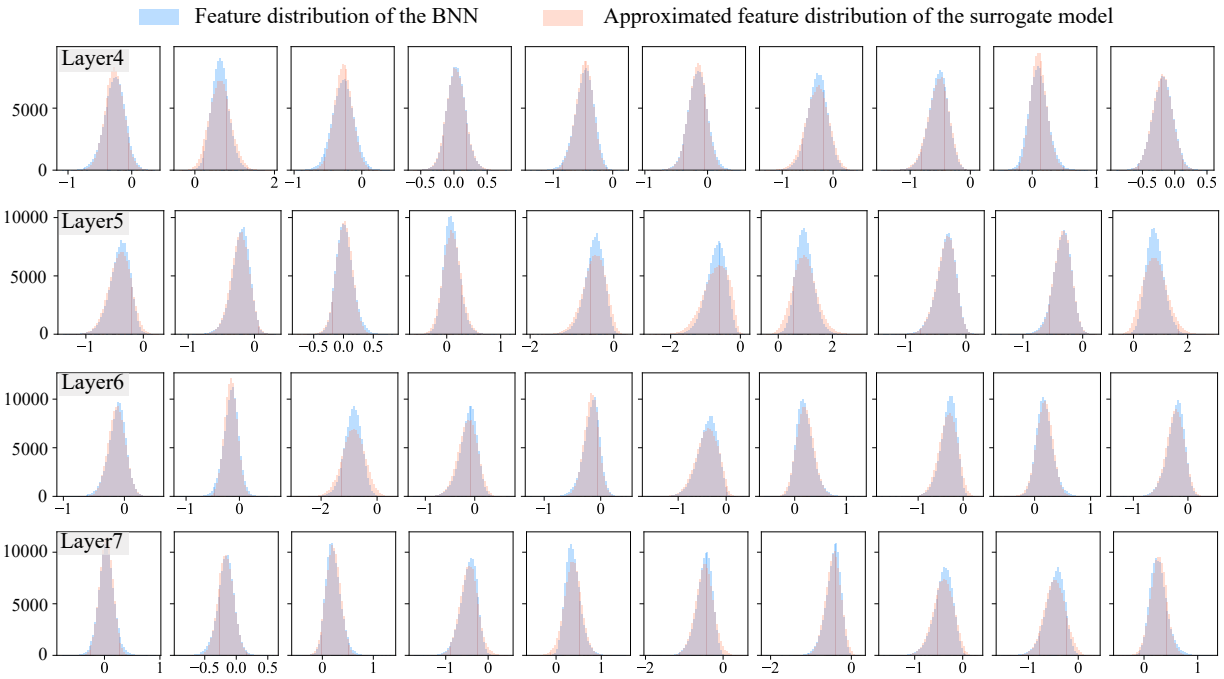

Figure 8. More visualization results of MLP-8 on the Census dataset.

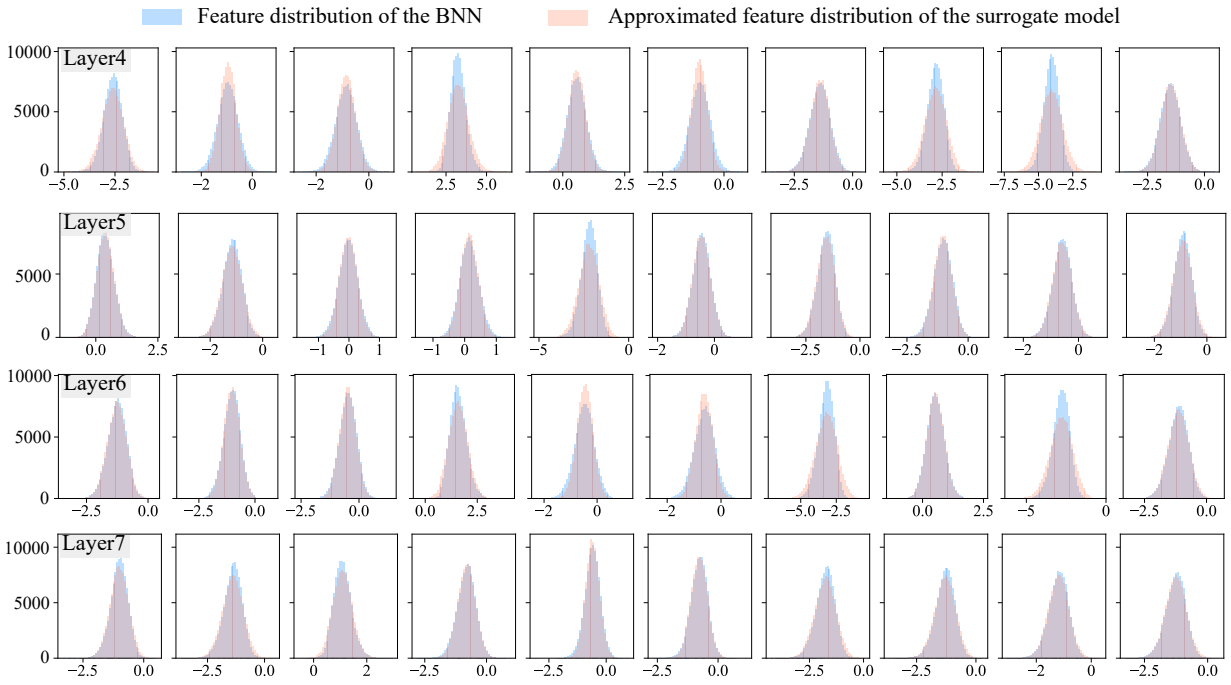

Figure 9. More visualization results of MLP-8 on the TV news dataset.