DISSECTING DNN OVERFITTING: THE INTERPLAY BETWEEN INFERENCE PATTERNS AND OVERFITTING

Anonymous authors
Paper under double-blind review

ABSTRACT

The overfitting of deep neural networks (DNNs) has rarely been studied from the perspective of their detailed inference patterns. In this paper, we explore whether DNN overfitting can be explained through the generalizability of interactions between input variables encoded by the network. Specifically, we explore this question from four complementary perspectives: (1) We find that overfitted DNNs tend to encode non-generalizable interactions. (2) We discover that removing non-transferable interactions from DNNs effectively mitigates overfitting. (3) This improvement arises because enhancing interaction transferability between two DNNs substantially improves the generalizability of these interactions. (4) We find that among various types of feature consistency across DNNs, improving interaction transferability is the most effective in mitigating overfitting.

1 INTRODUCTION

Understanding the generalization power of deep neural networks (DNNs) remains a crucial problem in machine learning. Previous studies have explored the generalization power of DNNs from different perspectives, including generalization bounds (Bartlett et al., 2017), the flatness of the loss landscape (Foret et al., 2020), and neural tangent kernel theory (Jacot et al., 2018), *etc*.

Unlike these existing studies, this paper aims to investigate whether a new paradigm exists for understanding DNN overfitting: *can the inference patterns encoded by DNNs be rigorously quantified, and can their generalizability explain the model's generalization power?* This perspective is analogous to assessing human expertise not only through performance, but also through the underlying decision-making logic (inference patterns).

Background. This line of research builds upon a series of empirical findings and theoretical results in interaction-based explanation (Li & Zhang, 2023b; Ren et al., 2024a; 2023a), which show that the complex inference patterns encoded by a DNN can be faithfully explained as interaction patterns between input variables.

> As shown in Figure 1, although the DNN appears highly complex, it usually encodes only a few interaction patterns, each representing a nonlinear relationship between a set of input variables. For example, a large language model (LLM) equivalently encodes a phrase composed of four words $T = \{ice, melted, due, to\}$, which is represented as an interaction between the four words. Interactions encoded by the LLM contribute numerical effects to the confidence score of generating a target token. *E.g.,* the above phrase (interaction) increases the confidence of generating the target token "*temperature*" by 1.34. Similarly, for vision models, appearance patterns learned by the model can also be represented as interactions between image regions.

The faithfulness of interactions is theoretically guaranteed and empirically verifiable, and has been widely adopted (see Section 2.1 and Appendix A for details).

Our work. Motivated by these findings, this paper explores, from a new perspective, *whether the complex generalization power of a DNN can be explained through interaction patterns.* We conduct four experiments to examine the connection between interactions and the model's overall generalization power[1] , and derive the following key findings.

[1]While DNN overfitting can be analyzed from various perspectives, we provisionally follow the widely adopted perspective (Zhang et al., 2019; Hoffer et al., 2017) to use the gap between training and testing losses to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

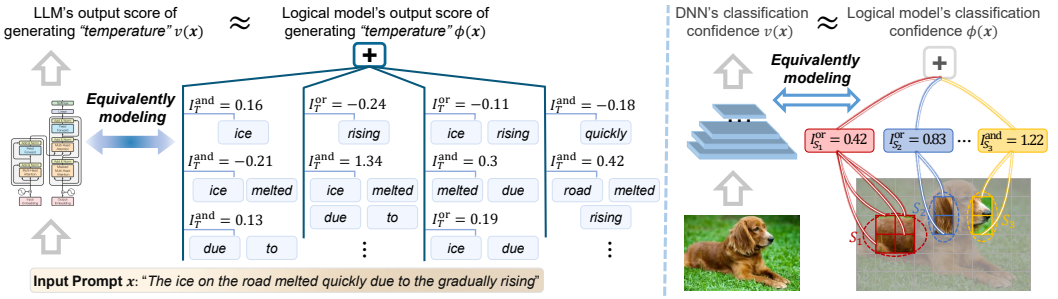

Figure 1: (Left) It has been proven by (Ren et al., 2024a) that the complex inference patterns of a DNN can be faithfully represented by a logical model $\phi(\boldsymbol{x})$ composed of a small number of interactions. Please see Figure 16 in the appendix for AND-OR logical models that explain LLMs. (Right) Illustration of using interactions to explain the inference patterns of visual models.

(1) The first finding is that **overfitted DNNs tend to encode non-generalizable interactions.** *An interaction is considered generalizable (Chen et al., 2024) if it is consistently encoded by different DNNs trained on different datasets; otherwise, it is considered non-generalizable.* Such non-generalizable interactions usually represent unreliable patterns.

(2) As a complement to the first perspective, we find that the transferability of interactions across DNNs trained on the same dataset is highly related to the generalizability of interactions (transferability to DNNs trained on unseen data). The second experiment shows that, when two DNNs are jointly trained on the same training dataset, **removing interactions that are not transferable between the two DNNs substantially alleviates overfitting of the entire DNNs.**

(3) To further substantiate our findings, the third experiment further finds that **enhancing interaction transferability between a specific pair of DNNs makes the learned interactions more likely to generalize/transfer to other DNNs trained on unseen data.** In this way, we can consider these interactions as the *knowledge points* encoded by the DNN. The enhanced generalizability of such knowledge points explains the mitigation of DNN overfitting.

(4) Beyond the above findings, the fourth experiment is conducted to examine whether enhancing interaction transferability (*i.e.,* interaction consistency between different DNNs) can more efficiently mitigate DNN overfitting than boosting other types of feature consistency. To this end, we find that, **among various types of feature consistency across DNNs, only enhancing interaction transferability effectively mitigates overfitting.**

2 Explaining DNN overfitting through interactions

2.1 Preliminaries: interactions

To formalize our ideas, we first revisit the emerging interaction-based explanation theory (Ren et al., 2023a; Li & Zhang, 2023b; Ren et al., 2024a), which uses interactions to represent a DNN's inference patterns. Let us consider a DNN v and an input sample $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$ with n input variables[2], which are indexed by $N = \{1, 2, \ldots, n\}$. We can generate up to 2^n masked states[3] of $\boldsymbol{x}$, given as $\Psi = \{\boldsymbol{x}_S | S \subseteq N\}$, where $\boldsymbol{x}_S$ denotes a masked state that retains only the input variables in S, with all variables in $N \setminus S$ being masked[3]. Let $v(\boldsymbol{x}) \in \mathbb{R}$ denote a scalar output of the DNN. People may define different outputs $v(\boldsymbol{x})$ for different tasks, *e.g.,* the widely-used classification confidence score (Deng et al., 2021; Ren et al., 2024b), as follows.

$$v(\boldsymbol{x}) \overset{\text{def}}{=} \log \frac{p(y = y^*|\boldsymbol{x})}{1 - p(y = y^*|\boldsymbol{x})} \in \mathbb{R}, \tag{1}$$

measure a model's overfitting level, which simplifies the analysis. Section 3 discusses other perspectives on analyzing the generalization power of DNNs.

[2]For image classification tasks, input variables are often considered as patches in an image. In language generation tasks, each input variable typically embodies the embedding vector of a token.

[3]Masking an input variable in S is achieved by substituting it with a baseline value, commonly defined as the average value of the variable across multiple samples (Dabkowski & Gal, 2017).

where $p(y = y^*|\boldsymbol{x})$ denotes the predicted probability of classifying $\boldsymbol{x}$ to the ground-truth label y^*.

In this way, Chen et al. (2024) have learned a logical model and proven that **the shape of the DNN function $v(\cdot)$ can be well matched by the following logical model function $\phi(\cdot)$ over all 2^n different masked states in Ψ.**

$$\forall \boldsymbol{x}' \in \Psi,\ |\phi(\boldsymbol{x}') - v(\boldsymbol{x}')| < \epsilon,$$
$$\text{s.t. } \phi(\boldsymbol{x}') \overset{\text{def}}{=} \sum_{T\in\Omega^{\text{and}}} \underbrace{I_T^{\text{and}} \cdot \mathbb{1}(_{\text{between input variables in } T}^{\boldsymbol{x}' \text{ triggers AND relation}})}_{\text{an AND interaction between input variables in } T} + \sum_{T\in\Omega^{\text{or}}} \underbrace{I_T^{\text{or}} \cdot \mathbb{1}(_{\text{between input variables in } T}^{\boldsymbol{x}' \text{ triggers OR relation}})}_{\text{an OR interaction between input variables in } T} + b, \quad (2)$$

where b is a scalar bias. ϵ is a small positive scalar that ensures the fidelity of using the function $\phi(\cdot)$ to mimic the function $v(\cdot)$. Please see Appendix A for the setting of ϵ and the theoretical and empirical validations of the low approximation error (*i.e.,* high fidelity) of the explanation. Ω^{and} and Ω^{or} represent the set of AND interactions and the set of OR interactions, respectively.

Taking AND-OR interactions in Equation (2) as primitive inference patterns. AND-OR interactions in Ω^{and} and Ω^{or} are powerful enough to accurately mimic $v(\cdot)$ on an exponential number of samples. As a result, these interactions are widely regarded as inference patterns in the DNN (Ren et al., 2023b; Zhou et al., 2024).

The trigger function $\mathbb{1}(_{\text{between input variables in } T}^{\boldsymbol{x}' \text{ triggers AND relation}}) \in \{0, 1\}$ represents an AND relation between a subset $T \subseteq N$ of input variables. It returns 1 if all variables in T are present (not masked[3] in $\boldsymbol{x}'$). The trigger function $\mathbb{1}(_{\text{between input variables in } T}^{\boldsymbol{x}' \text{ triggers OR relation}}) \in \{0, 1\}$ represents an OR relation between a subset $T \subseteq N$ of input variables. It returns 1 whenever any variable in T is present (not masked[3]) in $\boldsymbol{x}'$. $I_T^{\text{and}} \in \mathbb{R}$ and $I_T^{\text{or}} \in \mathbb{R}$ are scalar interaction weights.

Sparsity of interactions. Parameters I_T^{and} and I_T^{or} in the logical model $\phi(\cdot)$ are learned by following (Chen et al., 2024). Please see Appendix G for details. Ren et al. (2024a) have further proved that the number of interactions extracted by a DNN from an input sample is relatively small and theoretically bounded (empirically, around 50-150 interactions).

*Detailed proofs and empirical validation of the sparsity of interactions, and the high fidelity of using $\phi(\cdot)$ to mimic $v(\cdot)$ are introduced in Appendix C and Appendix A, respectively. The attached **video demo** in the supplementary material illustrates how to use interactions to explain DNNs.*

Discussion on computational complexity. Just like the computation of Shapley values (Shapley et al., 1953), extracting interaction effects entails exponential computational cost. Nevertheless, a line of prior studies has developed approximation algorithms (Kang et al., 2024; 2025; Butler et al., 2025) and practical engineering techniques (Ren et al., 2024b; Li & Zhang, 2023b) to speed up interaction extraction. Please refer to Appendix H for details.

2.2 Overfitted DNNs usually tend to encode non-generalizable interactions

The high fidelity of using the $\phi(\cdot)$ function to mimic the shape of the DNN function $v(\cdot)$ on 2^n samples in Equation (2) enables us to consider interactions as primitive inference patterns.

This naturally leads to the following question: *can the representation quality of interactions explain the overfitting of DNNs?* Therefore, in this study, **we quantify the generalizability of interactions, and explore the connection between interaction generalizability and DNN overfitting.**

Metrics for quantifying the generalizability of interactions. The generalizability of a feature is typically characterized by its transferability from training samples to held-out unseen samples (*e.g.,* testing samples). However, for each interaction, exhaustively identifying a certain interaction over all these unseen samples is computationally infeasible. Thus, given a target DNN v, we follow (Chen et al., 2024; He et al., 2025) to define the generalizability of an interaction encoded by v as its transferability to a *baseline DNN* v' trained on unseen samples. *An interaction encoded by v is considered generalizable if it is also encoded by v'*, because all interactions encoded by v' are learned from unseen samples.

Definition 1 (Interaction generalizability)**.** *For each AND interaction encoded by v subject to $S \in \Omega^{and}$, its generalizability is quantified by the following binary metric.*

$$\forall type \in \{and, or\},\ \forall S \in \Omega^{type}, \mathcal{G}_S^{type} \overset{def}{=} \mathbb{1}(S \in \Omega_{v'}^{type}) \cdot \mathbb{1}(I_S^{type} \cdot I_{S,v'}^{type} > 0). \quad (3)$$

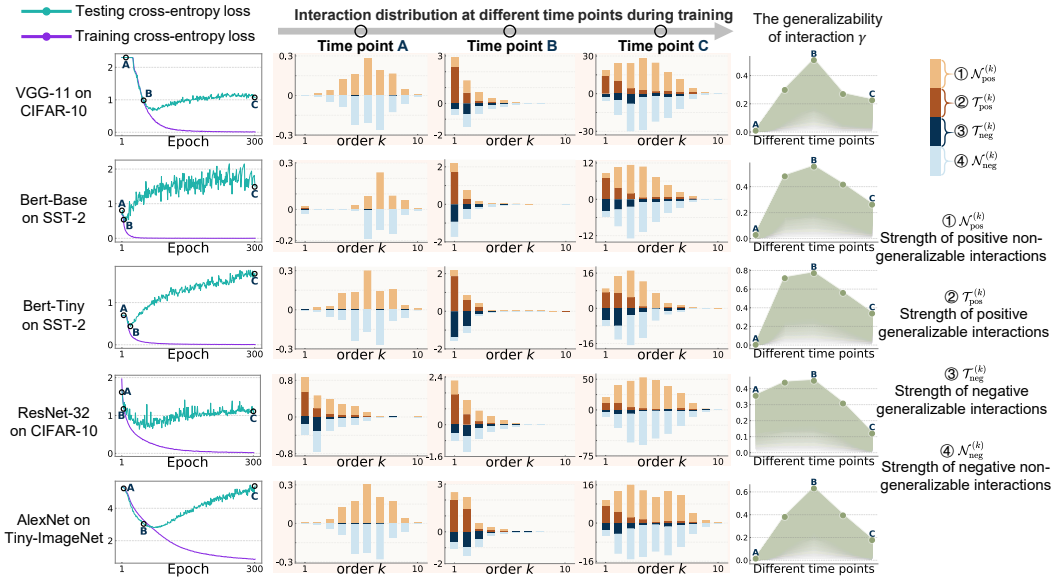

Figure 2: Evolution of the interaction distribution[4] during the normal training phase (from time point A to time point B) and the overfitting phase (from time point B to time point C). The onset of overfitting is identified as time point B, at which the training-testing loss gap begins to widen. We find that, during the overfitting phase, the DNN encodes an increasing number of non-generalizable interactions (*i.e.,* $\mathcal{N}_{\text{pos}}^{(k)}$ and $\mathcal{N}_{\text{neg}}^{(k)}$), so the generalizability of interactions (*i.e.,* the value of γ) gradually decreases. Please see Appendix E.7 for more fine-grained results at additional time points.

Similarly, for each OR interaction subject to $S \in \Omega^{or}$, its generalizability is given as $\mathcal{G}_S^{or}$ above. $\mathbb{1}(\cdot) \in \{0, 1\}$ is a trigger function that returns 1 if the given condition is satisfied.

Here, $\Omega_{v'}^{\text{and}}$ and $\Omega_{v'}^{\text{or}}$ denote the sets of AND and OR interactions encoded by the baseline DNN v'. The notations $I_{S,v'}^{\text{and}}$ and $I_{S,v'}^{\text{or}}$ denote the scalar interaction weights encoded by the baseline DNN v'. $I_S^{\text{and}} \cdot I_{S,v'}^{\text{and}} > 0$ indicates that the interaction S exerts similar effects on classifying the input sample $\boldsymbol{x}$ for both the target DNN and the baseline DNN.

Definition 2 (Order/complexity of interactions)**.** *The order (or complexity) of an AND/OR interaction S is defined as the number of input variables in the set S,* i.e., *order*$(S) = |S|$.

Metrics for the strength of generalizable interactions over different orders. The following four metrics measure (1) the strength of generalizable positive interactions of the k-th order $\mathcal{T}_{\text{pos}}^{(k)}$, (2) the strength of generalizable negative interactions of the k-th order $\mathcal{T}_{\text{neg}}^{(k)}$, (3) the strength of non-generalizable positive interactions $\mathcal{N}_{\text{pos}}^{(k)}$, and (4) the strength of non-generalizable negative interactions $\mathcal{N}_{\text{neg}}^{(k)}$.

$$\begin{aligned}
\mathcal{T}_{\text{pos}}^{(k)} &= \sum_{\text{type}\in\{\text{and, or}\}} \sum_{S\in\Omega^{\text{type}}:|S|=k} \max(I_S^{\text{type}}\cdot\mathcal{G}_S^{\text{type}}, 0),\ \mathcal{N}_{\text{pos}}^{(k)} = \sum_{\text{type}\in\{\text{and, or}\}} \sum_{S\in\Omega^{\text{type}}:|S|=k} \max(I_S^{\text{type}}\cdot(1-\mathcal{G}_S^{\text{type}}), 0),\\
\mathcal{T}_{\text{neg}}^{(k)} &= \sum_{\text{type}\in\{\text{and, or}\}} \sum_{S\in\Omega^{\text{type}}:|S|=k} \min(I_S^{\text{type}}\cdot\mathcal{G}_S^{\text{type}}, 0),\ \mathcal{N}_{\text{neg}}^{(k)} = \sum_{\text{type}\in\{\text{and, or}\}} \sum_{S\in\Omega^{\text{type}}:|S|=k} \min(I_S^{\text{type}}\cdot(1-\mathcal{G}_S^{\text{type}}), 0).
\end{aligned} \tag{4}$$

Furthermore, we use the following metric to quantify the average generalizability of interactions.

$$\gamma = \frac{\sum_k |\mathcal{T}_{\text{pos}}^{(k)}| + |\mathcal{T}_{\text{neg}}^{(k)}|}{\sum_k |\mathcal{T}_{\text{pos}}^{(k)}| + |\mathcal{T}_{\text{neg}}^{(k)}| + |\mathcal{N}_{\text{pos}}^{(k)}| + |\mathcal{N}_{\text{neg}}^{(k)}|}. \tag{5}$$

Discovering that overfitted DNNs tend to encode non-generalizable interactions. The training process of a DNN can be naturally divided into two phases. As shown in Figure 2, during the first phase (from time point A to time point B), the gap between the training and testing cross-entropy loss

[4]The distribution of interactions is computed by averaging the distribution across different samples.

remains small; we refer to this phase as *the normal training phase*. In the second phase (from time point B to time point C), the loss gap begins to widen, which we term *the overfitting phase.*

*Figure 2 shows that when the loss gap was relatively small, the DNN increasingly encoded generalizable interactions (*i.e., $\mathcal{T}_{pos}^{(k)}$ *and* $\mathcal{T}_{neg}^{(k)}$*). Once the loss gap began to widen, the generalizability of interactions gradually declined (*i.e., *gradually decreasing values of* γ*), as the network increasingly encoded non-generalizable interactions (*i.e., $\mathcal{N}_{pos}^{(k)}$ *and* $\mathcal{N}_{neg}^{(k)}$*).* ***These results indicated that the overfitted DNNs usually tended to encode non-generalizable interactions.***

Experimental details are as follows. For image data, we followed (Ren et al., 2024b) to select ten patches from each image as input variables; for natural language data, we treated the embedding of each word as an input variable. Please see Appendix F.5 for details. We trained AlexNet (Krizhevsky et al., 2012) on the Tiny-ImageNet dataset (mnmoustafa, 2017), VGG-11 (Simonyan & Zisserman, 2014) and ResNet-32 (He et al., 2016) on the CIFAR-10 dataset (Krizhevsky et al., 2009), and both Bert-Tiny and Bert-Base (Devlin et al., 2019) models on the SST-2 dataset (Socher et al., 2013).

2.3 Removing non-transferable interactions mitigates the DNN's overfitting

Despite the findings in the previous subsection, it remains difficult to confidently claim a clear connection between the DNN's overfitting and non-generalizable interactions. Therefore, in this subsection, we aim to validate this connection by (1) training a DNN without encoding such non-generalizable interactions, and (2) examining whether removing non-generalizable interactions from a DNN mitigates its overall overfitting.

However, because the baseline DNN is trained on unseen (testing) samples, directly penalizing non-generalizable interactions identified by the baseline DNN may introduce circularity. To avoid this issue, we instead co-train two target DNNs on the same training set, and penalize non-transferable interactions across the two target DNNs. We find that this strategy can also improve the generalizability of interactions, as discussed in detail in Section 2.4.

Training a DNN with non-transferable interactions removed. We jointly train two target DNNs, v_1 and v_2, on the same training set. Then, given a training sample $\boldsymbol{x}$, we introduce two loss terms, $\mathcal{L}_{\text{trans}}(v_1)$ and $\mathcal{L}_{\text{trans}}(v_2)$, to discourage the two DNNs from encoding non-transferable interactions through them.

$$\mathcal{L}_{\text{trans}}(v_1) = \mathbb{E}_{\boldsymbol{x}}\Big[\sum_{\text{type}\in\{\text{and,or}\}}\sum_{S\in\Omega^{\text{type}}}\big(1-\mathcal{G}_{\text{trans},S}^{\text{type}}(\boldsymbol{x})\big)\cdot|I_S^{\text{type}}(\boldsymbol{x})|\Big], \tag{6}$$

where $\big(1-\mathcal{G}_{\text{trans},S}^{\text{type}}(\boldsymbol{x})\big)\cdot|I_S^{\text{type}}(\boldsymbol{x})|$ denotes the strength of interactions in v_1 that cannot transfer to v_2. $\mathcal{G}_{\text{trans},S}^{\text{type}}$ is calculated by applying Equation (3) to the other target DNN v_2. The loss $\mathcal{L}_{\text{trans}}(v_2)$ for v_2 is computed analogously by swapping the roles of v_1 and v_2 in the computation of $\mathcal{G}_{\text{trans},S}^{\text{type}}$ and I_S^{type}.

We jointly optimize standard cross-entropy losses $\mathcal{L}_{\text{CE}}$ and the above losses $\mathcal{L}_{\text{trans}}$ for enhancing interaction transferability through the two DNNs, as follows.

$$\mathcal{L} = \lambda\cdot(\mathcal{L}_{\text{trans}}(v_1)+\mathcal{L}_{\text{trans}}(v_2)) + (1-\lambda)\cdot(\mathcal{L}_{\text{CE}}(v_1)+\mathcal{L}_{\text{CE}}(v_2)), \tag{7}$$

where $\lambda\in(0,1)$ is a scalar weight that controls the relative importance of the two loss terms. Directly minimizing the loss $\mathcal{L}_{\text{trans}}$ involves optimization challenges, so we propose an approximate yet differentiable version of this loss for implementation. Please refer to Appendix B for details.

Experiments: eliminating non-transferable interactions. We conducted experiments to verify whether the proposed loss $\mathcal{L}$ effectively eliminated non-transferable interactions. Specifically, we trained DNNs until overfitting, and then used the loss function $\mathcal{L}$ to remove non-transferable interactions from the overfitted DNNs. Please see Appendix F for detailed experimental settings.

We compared the originally overfitted DNN with its transferability-boosted counterpart. Figure 3 visualizes the distributions of interactions extracted from the two DNNs. *We found that the proposed loss* $\mathcal{L}$ *effectively eliminated non-transferable interactions and increased the average transferability of interactions* γ. This trend also holds for LLMs. Please see Appendix E.1 for experimental results.

The elimination of non-transferable interactions effectively mitigated DNN overfitting. We further compared the training-testing cross-entropy loss gaps between the overfitted DNN and the

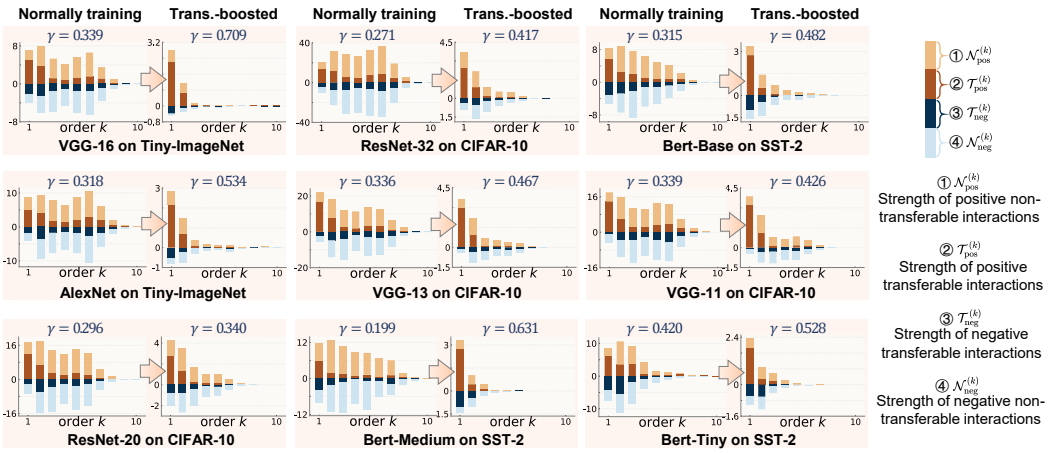

Figure 3: Comparing the interaction distribution[4] between the originally overfitted DNN (left in each pair) and its transferability-boosted counterpart, which is obtained by subsequently fine-tuning the overfitted DNN with $\mathcal{L}$ (right in each pair). We use the metric γ in Equation (5) to measure the transferability of interactions between v_1 and v_2, and γ is enhanced by eliminating non-transferable interactions while preserving transferable interactions. We set $\lambda = 0.3$ in this figure; please see Appendix E.2 for additional results under other choices of λ.

DNN whose non-transferable interactions were removed by the loss $\mathcal{L}$. Figure 4(a) shows that, when DNNs were trained using only the cross-entropy loss, the gap between the training and testing cross-entropy losses gradually increased. In contrast, *when the loss $\mathcal{L}$ was used to remove non-transferable interactions during the training process, DNN overfitting was rapidly alleviated. Specifically, the testing loss decreased while the training loss increased. These results indicated that eliminating non-transferable interactions effectively mitigated DNN overfitting.*

Furthermore, we varied the parameter λ in Equation (7) to remove different ratios of non-transferable interactions, so as to check the change in the training-testing cross-entropy loss gaps. Figure 4(b) shows that *larger values of λ, which eliminated more non-transferable interactions, led to a significantly smaller loss gap.*

Further evidence for the connection between non-transferable interactions and the DNN's overfitting. If DNN overfitting was indeed related to non-transferable interactions, then the degree of overfitting during training should respond immediately to whether the transferability-boosting loss term $\mathcal{L}_{\text{trans}}$ was applied. We found that whenever non-transferable interactions were removed, the degree of overfitting decreased immediately; conversely, when this removal was halted, the DNN gradually exhibited overfitting again. Please see Appendix E.4 for detailed results.

2.4 Deep probe: transferability enhances interaction generalizability

The previous subsection shows that penalizing non-transferable interactions between two DNNs trained on the same training set can effectively mitigate overfitting. In this subsection, we delve into the mechanisms through which eliminating non-transferable interactions helps mitigate overfitting.

Specifically, we aim to show that **promoting interaction transferability is, to some extent, equivalent to enhancing interaction generalizability.**

Quantifying interaction generalizability over multiple DNNs. For a more stable evaluation of interaction generalizability, we use multiple baseline DNNs to construct a set of *ground-truth generalizable AND interactions* and a set of *ground-truth OR interactions*, denoted by $\Omega^{\text{and}}_{\text{GT}}$ and $\Omega^{\text{or}}_{\text{GT}}$, respectively. Because generalizable AND interactions in $\Omega^{\text{and}}_{\text{GT}}$ are supposed to universally appear in different DNNs trained on different data, we additionally train m baseline DNNs with diverse architectures on separate training data[5] (we set $m = 4$ in this paper). An AND interaction S is

[5]To train different DNNs on different training data, we evenly partition the original dataset into $m+1$ disjoint subsets, *i.e.,* using one to train v_1 and v_2 based on Equation (7) and using m datasets to train m baseline DNNs. Please refer to Appendix F.3 for further details.

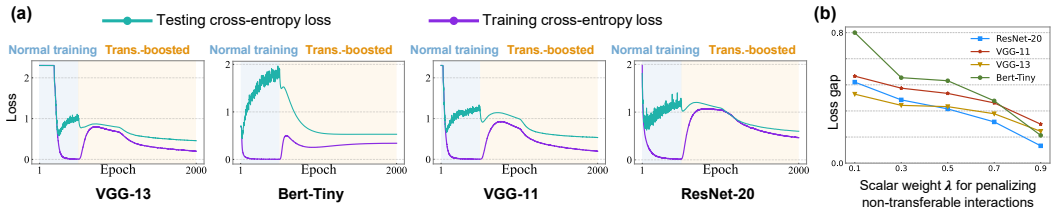

Figure 4: (a) Comparing the training-testing cross-entropy loss gap between an overfitted DNN and its transferability-boosted counterpart, which is obtained by further fine-tuning the overfitted DNN using the loss $\mathcal{L}$. Eliminating non-transferable interactions significantly reduces the loss gap. In this figure, we set $\lambda = 0.8$; additional results under other choices of λ are provided in Appendix E.3. (b) The loss gap progressively decreases as more non-transferable interactions are removed by increasing the value of λ.

collected into $\Omega^{\text{and}}_{\text{GT}}$ as a *ground-truth generalizable AND interaction* if it appears in more than $m/2$ of these DNNs and exerts consistent effects on classifying the input sample $\boldsymbol{x}$ across them. Ground-truth OR interactions are identified in the same manner and collected into $\Omega^{\text{or}}_{\text{GT}}$.

In this way, the obtained ground-truth generalizable interactions in $\Omega^{\text{and}}_{\text{GT}}$ and $\Omega^{\text{or}}_{\text{GT}}$ are believed to be more convincing. Thus, given sets of target interactions in Ω^{and} and Ω^{or}, we measure the ratio of interaction effects that belong to the ground-truth generalizable interactions, as follows.

$$\rho(\Omega^{\text{and}}, \Omega^{\text{or}}) \overset{\text{def}}{=} \frac{\sum_{\text{type}\in\{\text{and,or}\}} \sum_{S\in\Omega^{\text{type}}\cap\Omega^{\text{type}}_{\text{GT}}} |I^{\text{type}}_S| \cdot \mathbb{1}(I^{\text{type}}_S \cdot \sigma^{\text{type}}_{S,\text{GT}} > 0)}{\sum_{\text{type}\in\{\text{and,or}\}} \sum_{S\in\Omega^{\text{type}}} |I^{\text{type}}_S|}, \tag{8}$$

where $\sigma^{\text{type}}_{S,\text{GT}} \in \{-1, +1\}$ denotes the common sign of interaction S among the baseline DNNs.

Experimental validation of the high interaction generalizability of transferability-boosted DNNs. In this subsection, we conducted experiments to show that when the loss $\mathcal{L}$ in Equation (7) was used to encourage two DNNs to encode transferable interactions, these transferable interactions were usually highly generalizable to (universally transferable to) other DNNs trained on different datasets.

Notably, the above two target DNNs v_1 and v_2 were trained on the same dataset and were termed *transferability-boosted DNNs*. Given an input sample $\boldsymbol{x}$, a set of transferable AND interactions $\Omega^{\text{and}}_{\text{trans}}$, and a set of transferable OR interactions $\Omega^{\text{or}}_{\text{trans}}$ were obtained as follows.

$$\forall \text{type} \in \{\text{and}, \text{or}\},\ \Omega^{\text{type}}_{\text{trans}} \overset{\text{def}}{=} \{S \subseteq N | S \in \Omega^{\text{type}}_{v_1}, S \in \Omega^{\text{type}}_{v_2}, I^{\text{type}}_{S,v_1} \cdot I^{\text{type}}_{S,v_2} > 0\}, \tag{9}$$

where $\Omega^{\text{and}}_{v_1}$ and $\Omega^{\text{or}}_{v_1}$ denote the sets of AND interactions and OR interactions extracted from the DNN v_1, respectively. The conditions $I^{\text{and}}_{S,v_1} \cdot I^{\text{and}}_{S,v_2} > 0$ and $I^{\text{or}}_{S,v_1} \cdot I^{\text{or}}_{S,v_2} > 0$ indicate that the interaction S contributed similar effects to both v_1's classification and v_2's classification.

Therefore, we compared interaction generalizability between the *transferability-boosted DNNs* and the *normally trained DNNs*. Specifically, Figure 5 compares: (1) the ratio of generalizable interactions among all interactions extracted from a *normally trained DNN*, denoted by $\rho_1 = \rho(\Omega^{\text{and}}_{\text{normal}}, \Omega^{\text{or}}_{\text{normal}})$; (2) the ratio of generalizable interactions among all interactions extracted from a *transferability-boosted DNN*, denoted by $\rho_2 = \rho(\Omega^{\text{and}}_{v_1}, \Omega^{\text{or}}_{v_1})$; and (3) the ratio of generalizable interactions among all transferable interactions shared by two *transferability-boosted DNNs*, denoted by $\rho_3 = \rho(\Omega^{\text{and}}_{\text{trans}}, \Omega^{\text{or}}_{\text{trans}})$. The *normally trained DNNs* were trained on the same dataset as the *transferability-boosted DNNs*, using only the standard cross-entropy loss. Here, $\Omega^{\text{and}}_{\text{normal}}$ and $\Omega^{\text{or}}_{\text{normal}}$ denote the sets of interactions extracted from the normal DNN.

We found that interactions extracted from the *transferability-boosted DNNs* exhibited much higher generalizability ρ_2 than those from the *normally trained DNN* ρ_1. In particular, transferable interactions in $\Omega^{\text{and}}_{\text{trans}}$ and $\Omega^{\text{or}}_{\text{trans}}$ that were shared by two *transferability-boosted DNNs* showed the highest generalizability ρ_3. *These results indicated that enhancing interaction transferability between two DNNs trained on the same dataset enabled the learned interactions to achieve universal generalizability across diverse DNNs trained on different datasets.*

Experimental details are as follows. For the CIFAR-10 dataset, two VGG-11 models were trained until overfitting was observed and served as the *normally trained DNNs*. We then continued fine-

[6] As in Equation (4), the strength of generalizable interactions *w.r.t.* $\Omega^{\text{and}}_{\text{GT}}$ and $\Omega^{\text{or}}_{\text{GT}}$ is defined in Appendix F.2.

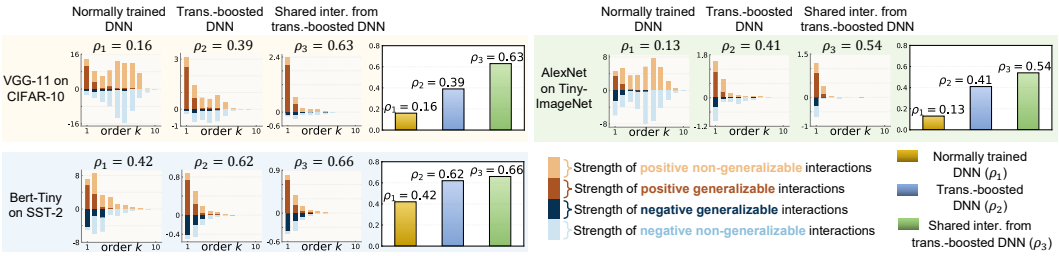

Figure 5: Comparing the generalizability of interactions extracted from the normally trained DNN with that extracted from the transferability-boosted DNN. The overall strength of all generalizable interactions and non-generalizable interactions *w.r.t.* $\Omega^{\text{and}}_{\text{GT}}$ and $\Omega^{\text{or}}_{\text{GT}}$ of each order is computed[6]. Interactions extracted from transferability-boosted DNNs show consistently higher generalizability.

tuning the two DNNs using the proposed loss $\mathcal{L}$ in Equation (7) to obtain two *transferability-boosted DNNs*. For the SST-2 dataset, we selected the Bert-Tiny model as the target architecture, so as to compare the generalizability of interactions between the *normally trained DNN* and the *transferability-boosted DNN*. Similarly, for the Tiny-ImageNet dataset, we selected the AlexNet model as the target architecture. Appendix F.3 introduced how we trained m different DNNs to construct ground-truth generalizable interactions for each target DNN.

2.5 What matters more for overfitting: interaction generalizability or feature consistency?

Although the findings in the previous three subsections have indicated a close connection between non-generalizable (*i.e.,* non-transferable) interactions and the DNN's overfitting, it remains unclear whether interactions provide a more suitable explanation than other related approaches.

In fact, the loss $\mathcal{L}$ proposed in Equation (7) can be considered to enhance the consistency of interactions across two different DNNs. This motivates the following question: **compared with other forms of feature consistency approaches, is enhancing interaction consistency better suited to explain the DNN's overfitting?**

Therefore, to answer the above question, we compare the effectiveness in mitigating overfitting between the proposed loss $\mathcal{L}$ and two widely used loss functions designed to promote feature consistency (Hinton et al., 2015; Romero et al., 2014). Given a training sample $\boldsymbol{x}$, let $\mathbf{z}(\boldsymbol{x})$ and $\mathbf{z}'(\boldsymbol{x})$ denote the outputs before the softmax layer of two DNNs, v and v', respectively. The first competing loss function penalizes the squared error between $\mathbf{z}(\boldsymbol{x})$ and $\mathbf{z}'(\boldsymbol{x})$, as follows.

$$\mathcal{L}_{\text{z}} = \mathbb{E}_{\boldsymbol{x}}\left[\lambda \cdot \left\|\mathbf{z}(\boldsymbol{x}) - \mathbf{z}'(\boldsymbol{x})\right\|_2^2 + (1-\lambda)\cdot\left(\mathcal{L}_{\text{CE}}(v) + \mathcal{L}_{\text{CE}}(v')\right)\right], \tag{10}$$

where $\|\cdot\|_2^2$ denotes the squared ℓ_2-norm. $\lambda \in (0,1)$ is a scalar weight. The second competing loss penalizes the squared error between intermediate-layer features of two DNNs v and v', as follows.

$$\mathcal{L}_{\text{f}} = \mathbb{E}_{\boldsymbol{x}}\left[\lambda \cdot \left\|\mathbf{f}(\boldsymbol{x}) - \mathbf{f}'(\boldsymbol{x})\right\|_2^2 + (1-\lambda)\cdot\left(\mathcal{L}_{\text{CE}}(v) + \mathcal{L}_{\text{CE}}(v')\right)\right], \tag{11}$$

where $\mathbf{f}(\boldsymbol{x})$ and $\mathbf{f}'(\boldsymbol{x})$ denote intermediate-layer features of two DNNs, v and v', respectively.

We conducted experiments to compare DNNs trained using our proposed loss $\mathcal{L}$ with DNNs trained on the two competing losses above (*i.e.,* $\mathcal{L}_{\text{z}}$ and $\mathcal{L}_{\text{f}}$). Please see Appendix F for detailed settings.

Figure 6 shows that only our loss $\mathcal{L}$ effectively eliminated non-transferable interactions, whereas the competing methods (*i.e.,* $\mathcal{L}_{\text{z}}$ and $\mathcal{L}_{\text{f}}$) failed to do so. Among all methods, the DNN trained with $\mathcal{L}$ exhibited the highest ratio γ of transferable interactions. Furthermore, Figure 14 in the Appendix shows that only our loss $\mathcal{L}$ successfully reduced the training-testing cross-entropy loss gap. These results indicated that *the elimination of non-transferable interactions was the most effective way to mitigate overfitting, while other forms of feature consistency did not have this effect.*

In addition, we further compared our method with standard regularization methods, including weight decay and dropout. As reported in Appendix E.6, compared with these standard regularization

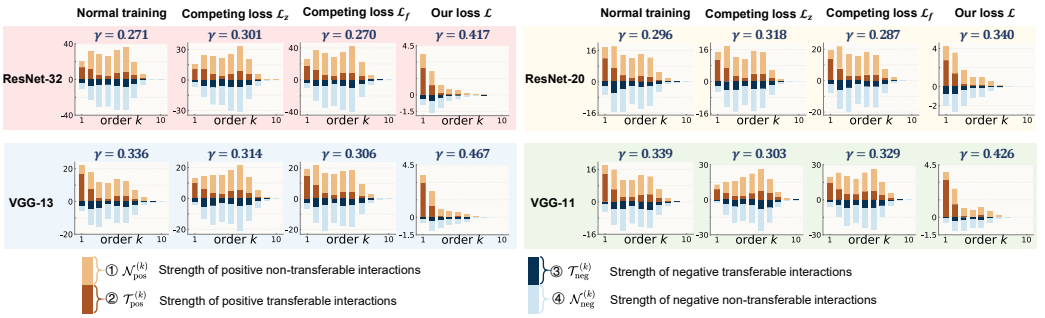

Figure 6: Comparing distributions[4] of transferable and non-transferable interactions learned by different methods. Only the proposed loss $\mathcal{L}$ effectively eliminates non-transferable interactions, and enhances the average transferability of interactions (*i.e.,* the value of γ). In contrast, the other methods exhibit negligible improvements compared to normal training.

methods, our method consistently achieved a substantially smaller training-testing loss gap, while maintaining comparable or slightly improved testing accuracy.

3 Related work and discussions

The generalizability of interactions does not exactly coincide with the overall generalization power of a DNN, but it offers a verifiable metric for explaining the quality of the DNN's inference patterns. Conventional studies, including analyses based on the train-test loss gap (Hoffer et al., 2017; Jiang et al., 2018; Bousquet et al., 2020), neural tangent kernel theory (Jacot et al., 2018), loss-landscape smoothness (Kwon et al., 2021; Keskar et al., 2016; Frumkin et al., 2023; Foret et al., 2020), and generalization bounds on high-dimensional feature spaces (Dziugaite & Roy, 2017; Neyshabur et al., 2015; Bartlett et al., 2017), largely regard model generalization as an overall property.

Interaction-based explanation. A series of studies (Ren et al., 2023a; Li & Zhang, 2023b; Ren et al., 2024a;b; Chen et al., 2024) have empirically observed and further proven that the complex inference logic of a DNN can be faithfully explained as interaction patterns. Specifically, Ren et al. (2023a; 2024a) have proven that a DNN's output scores can be accurately mimicked by a few AND-OR interactions. Furthermore, Zhou et al. (2024); Ren et al. (2024b) have empirically observed that complex interactions tend to exhibit poor generalizability. We go beyond the scope of existing studies and explore the connection between DNN overfitting and interaction generalizability, providing more fine-grained insights into the generalization power of DNNs.

4 Conclusions

In this paper, we have conducted four experiments to validate the connection between DNN overfitting and the generalizability of interactions from four perspectives. First, we find that overfitted DNNs tend to learn more non-generalizable interactions than normally trained DNNs before overfitting. Second, we discover that explicitly removing non-transferable interactions effectively mitigates DNN overfitting. Third, we further explore the underlying causes of this phenomenon, *i.e.,* we find that removing non-transferable interactions between two DNNs also enhances the generalizability of interactions to other DNNs trained on different datasets for the same task. The improved interaction generalizability explains the mitigation of the DNN's overfitting. Finally, by comparing different feature-consistency approaches, we find that enhancing interaction generalizability is more effective in mitigating DNN overfitting than other forms of feature consistency.

Practical value. This paper provides a fine-grained lens for analyzing DNN overfitting. For example, early stopping is a long-standing issue in training AI models. Interactions may provide a detailed signal for identifying when a DNN begins to learn overfitted interactions, and pinpoint the specific set of training samples on which such overfitted interactions emerge. This enables us to identify the time point at which the model's representations achieve peak generalizability. Such insights offer a perspective for guiding the DNN training process. Moreover, subsequent experiments in Appendix D show that removing non-transferable interactions can also improve model performance.

AI USE STATEMENT

Generative AI tools were used solely to improve the clarity, grammar, and readability of the manuscript. They were not used to develop research ideas, theoretical analyses, methods, experiments, data analysis, result interpretation, or scientific conclusions. All AI-assisted edits were reviewed by the authors, who take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Detailed experimental settings are provided in Appendix F. The complete procedure and pseudocode for extracting AND-OR interactions are presented in Appendix G. We will publicly release the source code and configuration files upon acceptance.

REFERENCES

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. *Advances in neural information processing systems*, 30, 2017.

Olivier Bousquet, Yegor Klochkov, and Nikita Zhivotovskiy. Sharper bounds for uniformly stable algorithms. In *Conference on Learning Theory*, pp. 610–626. PMLR, 2020.

Landon Butler, Abhineet Agarwal, Justin Singh Kang, Yigit Efe Erginbas, Bin Yu, and Kannan Ramchandran. Proxyspex: Inference-efficient interpretability via sparse feature interactions in llms. *arXiv preprint arXiv:2505.17495*, 2025.

Lu Chen, Siyu Lou, Benhao Huang, and Quanshi Zhang. Defining and extracting generalizable interaction primitives from DNNs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=OCqyFVFNeF.

Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. *Advances in neural information processing systems*, 30, 2017.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. Goemotions: A dataset of fine-grained emotions. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 4040–4054, 2020.

Huiqi Deng, Qihan Ren, Hao Zhang, and Quanshi Zhang. Discovering and explaining the representation bottleneck of dnns. *arXiv preprint arXiv:2111.06236*, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Natalia Frumkin, Dibakar Gope, and Diana Marculescu. Jumping through local minima: Quantization in the loss landscape of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16978–16988, 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Yuxuan He, Junpeng Zhang, Lei Cheng, Hongyuan Zhang, and Quanshi Zhang. Technical report: Quantifying and analyzing the generalization power of a dnn. *arXiv preprint arXiv:2505.06993*, 2025.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *Advances in neural information processing systems*, 30, 2017.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. *arXiv preprint arXiv:1810.00113*, 2018.

Justin S Kang, Yigit E Erginbas, Landon Butler, Ramtin Pedarsani, and Kannan Ramchandran. Learning to understand: Identifying interactions via the möbius transform. *Advances in Neural Information Processing Systems*, 37:46160–46202, 2024.

Justin Singh Kang, Landon Butler, Abhineet Agarwal, Yigit Efe Erginbas, Ramtin Pedarsani, Kannan Ramchandran, and Bin Yu. Spex: Scaling feature interaction explanations for llms. *arXiv preprint arXiv:2502.13870*, 2025.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International conference on machine learning*, pp. 5905–5914. PMLR, 2021.

Mingjie Li and Quanshi Zhang. Defining and quantifying and-or interactions for faithful and concise explanation of dnns. *arXiv preprint arXiv:2304.13312*, 2023a.

Mingjie Li and Quanshi Zhang. Does a neural network really encode symbolic concepts? In *International conference on machine learning*, pp. 20452–20469. PMLR, 2023b.

Mohammed Ali mnmoustafa. Tiny imagenet, 2017. URL `https://kaggle.com/competitions/tiny-imagenet`.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. Norm-based capacity control in neural networks. In *Conference on learning theory*, pp. 1376–1401. PMLR, 2015.

Jie Ren, Mingjie Li, Qirui Chen, Huiqi Deng, and Quanshi Zhang. Defining and quantifying the emergence of sparse concepts in dnns. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20280–20289, 2023a.

Qihan Ren, Huiqi Deng, Yunuo Chen, Siyu Lou, and Quanshi Zhang. Bayesian neural networks avoid encoding complex and perturbation-sensitive concepts. In *International Conference on Machine Learning*, pp. 28889–28913. PMLR, 2023b.

Qihan Ren, Jiayang Gao, Wen Shen, and Quanshi Zhang. Where we have arrived in proving the emergence of sparse interaction primitives in dnns. In *The Twelfth International Conference on Learning Representations*, 2024a.

Qihan Ren, Junpeng Zhang, Yang Xu, Yue Xin, Dongrui Liu, and Quanshi Zhang. Towards the dynamics of a dnn learning symbolic interactions. *Advances in Neural Information Processing Systems*, 37:50653–50688, 2024b.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

Lloyd S Shapley et al. A value for n-person games. 1953.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

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

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL `https://qwenlm.github.io/blog/qwen2.5/`.

Haotian Zhang, Lin Zhang, and Yuan Jiang. Overfitting and underfitting analysis for deep learning based end-to-end communication systems. In *2019 11th international conference on wireless communications and signal processing (WCSP)*, pp. 1–6. IEEE, 2019.

Huilin Zhou, Hao Zhang, Huiqi Deng, Dongrui Liu, Wen Shen, Shih-Han Chan, and Quanshi Zhang. Explaining generalization power of a dnn using interactive concepts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17105–17113, 2024.

APPENDIX

This appendix provides detailed information that supports the main paper. For clarity, the appendix is organized as follows.

- Appendix A validates the high fidelity of the interaction-based explanations.
- Appendix B describes the implementation details for the transferability-boosted loss $\mathcal{L}_{\text{trans}}$.
- Appendix C examines the sparsity properties of interactions.
- Appendix D discusses the practical value of this work.
- Appendix E presents more experimental results that complement those reported in the main paper.
- Appendix F provides detailed experimental settings.
- Appendix G introduces the procedure for extracting interactions from DNNs.
- Appendix H discusses the computational cost and practical scalability of interactions.

A VALIDATION OF HIGH-FIDELITY INTERACTION-BASED EXPLANATION

In this section, we validate the high fidelity of the interaction-based explanation from both theoretical and empirical perspectives. We first provide a theoretical analysis to establish its high fidelity, and then further corroborate it through empirical validation.

A.1 THEORETICAL VALIDATION

Proof. **(1) Universal matching theorem of AND interactions.**

We will prove that output component u_S^{and} on all 2^n masked samples $\{\boldsymbol{x}_S : S \subseteq N\}$ could be universally explained by the all interactions in $S \subseteq N$, *i.e.*, $\forall \emptyset \neq S \subseteq N, u_S^{\text{and}} = \sum_{\emptyset \neq T \subseteq S} I_T^{\text{and}} + v(\boldsymbol{x}_\emptyset)$. In particular, we define $v_\emptyset^{\text{and}} = v(\boldsymbol{x}_\emptyset)$ (*i.e.*, we attribute output on an empty sample to AND interactions).

Specifically, the AND interaction is defined as $I_T^{\text{and}} = \sum_{L \subseteq T} (-1)^{|T|-|L|} u_L^{\text{and}}$. To compute the sum of AND interactions $\sum_{\emptyset \neq T \subseteq S} I_T^{\text{and}} = \sum_{\emptyset \neq T \subseteq S} \sum_{L \subseteq T} (-1)^{|T|-|L|} u_L^{\text{and}}$, we first exchange the order of summation of the set $L \subseteq T \subseteq S$ and the set $T \supseteq L$. That is, we compute all linear combinations of all sets T containing L with respect to the model outputs u_L^{and} given a set of input variables L, *i.e.*, $\sum_{T: L \subseteq T \subseteq S} (-1)^{|T|-|L|} u_L^{\text{and}}$. Then, we compute all summations over the set $L \subseteq S$.

In this way, we can compute them separately for different cases of $L \subseteq T \subseteq S$. In the following, we consider the cases (1) $L = S = T$, and (2) $L \subseteq T \subseteq S, L \neq S$, respectively.

(1) When $L = S = T$, the linear combination of all subsets T containing L with respect to the model output u_L^{and} is $(-1)^{|S|-|S|} u_L^{\text{and}} = u_L^{\text{and}}$.

(2) When $L \subseteq T \subseteq S, L \neq S$, the linear combination of all subsets T containing L with respect to the model output u_L^{and} is $\sum_{T: L \subseteq T \subseteq S} (-1)^{|T|-|L|} u_L^{\text{and}}$. For all sets $T : S \supseteq T \supseteq L$, let us consider the linear combinations of all sets T with number $|T|$ for the model output u_L^{and}, respectively. Let $m := |T| - |L|$, $(0 \leq m \leq |S| - |L|)$, then there are a total of $C_{|S|-|L|}^m$ combinations of all sets T of order $|T|$. Thus, given L, accumulating the model outputs u_L^{and} corresponding to all $T \supseteq L$, then $\sum_{T: L \subseteq T \subseteq S} (-1)^{|T|-|L|} u_L^{\text{and}} = u_L^{\text{and}} \cdot \underbrace{\sum_{m=0}^{|S|-|L|} C_{|S|-|L|}^m (-1)^m}_{=0} = 0$. Please see the complete derivation of the following formula.

$$
\begin{aligned}
\sum\nolimits_{\emptyset \neq T \subseteq S} I_T^{\text{and}} &= \sum\nolimits_{\emptyset \neq T \subseteq S} \sum\nolimits_{L \subseteq T} (-1)^{|T|-|L|} u_L^{\text{and}} \\
&= \sum\nolimits_{L \subseteq S} \sum\nolimits_{T: L \subseteq T \subseteq S} (-1)^{|T|-|L|} u_L^{\text{and}} - v_\emptyset^{\text{and}} \\
&= \underbrace{u_S^{\text{and}}}_{L=S} + \sum\nolimits_{L \subseteq S, L \neq S} u_L^{\text{and}} \cdot \underbrace{\sum\nolimits_{m=0}^{|S|-|L|} C_{|S|-|L|}^m (-1)^m}_{=0} - v_\emptyset^{\text{and}} \\
&= u_S^{\text{and}} - v_\emptyset^{\text{and}} = u_S^{\text{and}} - v(\boldsymbol{x}_\emptyset)
\end{aligned}
\tag{12}
$$

Thus, we have $\forall \emptyset \neq S \subseteq N, u_S^{\text{and}} = \sum_{\emptyset \neq T \subseteq S} I_T^{\text{and}} + v(\boldsymbol{x}_\emptyset)$.

(2) Universal matching theorem of OR interactions.

According to the definition of OR interactions, we will derive that $\forall S \subseteq N, u_S^{\text{or}} = \sum_{T: T \cap S \neq \emptyset} I_T^{\text{or}}$, where we define $v_\emptyset^{\text{or}} = 0$ (recall that in Step (1), we attribute the output on empty input to AND interactions).

Specifically, the OR interaction is defined as $I_T^{\text{or}} = -\sum_{L \subseteq T} (-1)^{|T|-|L|} v_{N \setminus L}^{\text{or}}$. Similar to the above derivation of the universal matching theorem of AND interactions, to compute the sum of OR interactions $\sum_{T: T \cap S \neq \emptyset} I_T^{\text{or}} = \sum_{T: T \cap S \neq \emptyset} \left[-\sum_{L \subseteq T} (-1)^{|T|-|L|} v_{N \setminus L}^{\text{or}} \right]$, we first exchange the order of summation of the set $L \subseteq T \subseteq N$ and the set $T : T \cap S \neq \emptyset$. That is, we compute all linear combinations of all sets T containing L with respect to the model outputs $v_{N \setminus L}^{\text{or}}$ given a set of input variables L, *i.e.*, $\sum_{T: T \cap S \neq \emptyset, T \supseteq L} (-1)^{|T|-|L|} v_{N \setminus L}^{\text{or}}$. Then, we compute all summations over the set $L \subseteq N$.

In this way, we can compute them separately for different cases of $L \subseteq T \subseteq N, T \cap S \neq \emptyset$. In the following, we consider the cases (1) $L = N \setminus S$, (2) $L = N$, (3) $L \cap S \neq \emptyset, L \neq N$, and (4) $L \cap S = \emptyset, L \neq N \setminus S$, respectively.

(1) When $L = N \setminus S$, the linear combination of all subsets T containing L with respect to the model output $v^{\text{or}}_{N\setminus L}$ is $\sum_{T:T\cap S\neq\emptyset,T\supseteq L}(-1)^{|T|-|L|}v^{\text{or}}_{N\setminus L} = \sum_{T:T\cap S\neq\emptyset,T\supseteq L}(-1)^{|T|-|L|}u^{\text{or}}_{S}$. For all sets $T : T \supseteq L, T \cap S \neq \emptyset$ (then $T \neq N \setminus S, T \neq L$), let us consider the linear combinations of all sets T with number $|T|$ for the model output u^{or}_S, respectively. Let $|T'| := |T| - |L|$, $(1 \leq |T'| \leq |S|)$, then there are a total of $C^{|T'|}_{|S|}$ combinations of all sets T' of order $|T'|$. Thus, given L, accumulating the model outputs u^{or}_S corresponding to all $T \supseteq L$, then $\sum_{T:T\cap S\neq\emptyset,T\supseteq L}(-1)^{|T|-|L|}v^{\text{or}}_{N\setminus L} = u^{\text{or}}_S \cdot \underbrace{\sum_{|T'|=1}^{|S|} C^{|T'|}_{|S|}(-1)^{|T'|}}_{=-1} = -u^{\text{or}}_S$.

(2) When $L = N$ (then $T = N$), the linear combination of all subsets T containing L with respect to the model output $v^{\text{or}}_{N\setminus L}$ is $\sum_{T:T\cap S\neq\emptyset,T\supseteq L}(-1)^{|T|-|L|}v^{\text{or}}_{N\setminus L} = (-1)^{|N|-|N|}v^{\text{or}}_{\emptyset} = v^{\text{or}}_{\emptyset}$.

(3) When $L \cap S \neq \emptyset, L \neq N$, the linear combination of all subsets T containing L with respect to the model output $v^{\text{or}}_{N\setminus L}$ is $\sum_{T:T\cap S\neq\emptyset,T\supseteq L}(-1)^{|T|-|L|}v^{\text{or}}_{N\setminus L}$. For all sets $T : T \supseteq L, T \cap S \neq \emptyset$, let us consider the linear combinations of all sets T with number $|T|$ for the model output u^{or}_S, respectively. Let us split $|T| - |L|$ into $|T'|$ and $|T''|$, *i.e.*, $|T| - |L| = |T'| + |T''|$, where $T' = \{i|i \in T, i \notin L, i \in N \setminus S\}$, $T'' = \{i|i \in T, i \notin L, i \in S\}$ (then $0 \leq |T''| \leq |S| - |S \cap L|$) and $|T'| + |T''| + |L| = |T|$. In this way, there are a total of $C^{|T''|}_{|S|-|S\cap L|}$ combinations of all sets T'' of order $|T''|$. Thus, given L, accumulating the model outputs $v^{\text{or}}_{N\setminus L}$ corresponding to all $T \supseteq L$, then $\sum_{T:T\cap S\neq\emptyset,T\supseteq L}(-1)^{|T|-|L|}v^{\text{or}}_{N\setminus L} = v^{\text{or}}_{N\setminus L} \cdot \sum_{T'\subseteq N\setminus S\setminus L} \underbrace{\sum_{|T''|=0}^{|S|-|S\cap L|} C^{|T''|}_{|S|-|S\cap L|}(-1)^{|T'|+|T''|}}_{=0} = 0$.

(4) When $L \cap S = \emptyset, L \neq N \setminus S$, the linear combination of all subsets T containing L with respect to the model output $v^{\text{or}}_{N\setminus L}$ is $\sum_{T:T\cap S\neq\emptyset,T\supseteq L}(-1)^{|T|-|L|}v^{\text{or}}_{N\setminus L}$. Similarly, let us split $|T| - |L|$ into $|T'|$ and $|T''|$, *i.e.*, $|T| - |L| = |T'| + |T''|$, where $T' = \{i|i \in T, i \notin L, i \in N \setminus S\}$, $T'' = \{i|i \in T, i \in S\}$ (then $0 \leq |T''| \leq |S|$) and $|T'| + |T''| + |L| = |T|$. In this way, there are a total of $C^{|T''|}_{|S|}$ combinations of all sets T'' of order $|T''|$. Thus, given L, accumulating the model outputs $v^{\text{or}}_{N\setminus L}$ corresponding to all $T \supseteq L$, then $\sum_{T:T\cap S\neq\emptyset,T\supseteq L}(-1)^{|T|-|L|}v^{\text{or}}_{N\setminus L} = v^{\text{or}}_{N\setminus L} \cdot \sum_{T'\subseteq N\setminus S\setminus L} \underbrace{\sum_{|T''|=0}^{|S|} C^{|T''|}_{|S|}(-1)^{|T'|+|T''|}}_{=0} = 0$.

Please see the complete derivation of the following formula.

$$
\begin{aligned}
\sum\nolimits_{T:T\cap S\neq\emptyset} I_T^{\text{or}} &= \sum\nolimits_{T:T\cap S\neq\emptyset}\left[-\sum\nolimits_{L\subseteq T}(-1)^{|T|-|L|}v_{N\setminus L}^{\text{or}}\right] \\
&= -\sum\nolimits_{L\subseteq N}\sum\nolimits_{T:T\cap S\neq\emptyset, T\supseteq L}(-1)^{|T|-|L|}v_{N\setminus L}^{\text{or}} \\
&= -\left[\sum_{|T'|=1}^{|S|} C_{|S|}^{|T'|}(-1)^{|T'|}\right]\cdot \underbrace{u_S^{\text{or}}}_{L=N\setminus S} - \underbrace{v_\emptyset^{\text{or}}}_{L=N} \\
&\quad - \sum_{L\cap S\neq\emptyset, L\neq N}\left[\sum_{T'\subseteq N\setminus S\setminus L}\left(\sum_{|T''|=0}^{|S|-|S\cap L|} C_{|S|-|S\cap L|}^{|T''|}(-1)^{|T'|+|T''|}\right)\right]\cdot v_{N\setminus L}^{\text{or}} \\
&\quad - \sum_{L\cap S=\emptyset, L\neq N\setminus S}\left[\sum_{T'\subseteq N\setminus S\setminus L}\left(\sum_{|T''|=0}^{|S|} C_{|S|}^{|T''|}(-1)^{|T'|+|T''|}\right)\right]\cdot v_{N\setminus L}^{\text{or}} \\
&= -(-1)\cdot u_S^{\text{or}} - v_\emptyset^{\text{or}} - \sum_{L\cap S\neq\emptyset, L\neq N}\left[\sum_{T'\subseteq N\setminus S\setminus L} 0\right]\cdot v_{N\setminus L}^{\text{or}} \\
&\quad - \sum_{L\cap S=\emptyset, L\neq N\setminus S}\left[\sum_{T'\subseteq N\setminus S\setminus L} 0\right]\cdot v_{N\setminus L}^{\text{or}} \\
&= u_S^{\text{or}} - v_\emptyset^{\text{or}} \\
&= u_S^{\text{or}}
\end{aligned}
\tag{13}
$$

(3) Universal matching theorem of AND-OR interactions. With the universal matching theorem of AND interactions and the universal matching theorem of OR interactions, we can easily get $v(\boldsymbol{x}_S) = u_S^{\text{and}} + u_S^{\text{or}} = v(\boldsymbol{x}_\emptyset) + \sum_{\emptyset\neq T\subseteq S} I_T^{\text{and}} + \sum_{T:T\cap S\neq\emptyset} I_T^{\text{or}}$, thus, we obtain the universal matching theorem of AND-OR interactions.

□

A.2 Empirical validation

Given an input sample, the above subsection provides a theoretical proof that the shape of the DNN function $v(\cdot)$ is well matched by the corresponding logical model function $\phi(\cdot)$ over all 2^n different masked states in Ψ. In this subsection, we further provide empirical evidence demonstrating that the logical model can faithfully approximate DNN outputs in practice.

Specifically, we randomly select a subset of samples and compute the outputs of both the logical model and the corresponding DNN on all 2^n masked states of these samples.

We conduct experiments by training VGG-11 and ResNet-20 on the CIFAR-10 dataset, AlexNet and VGG-16 on the Tiny-ImageNet dataset, and the Bert-Base and Bert-Tiny models on the SST-2 dataset. For each dataset, we randomly select five samples and compare the outputs of the logical models with those of the corresponding DNNs. Figure 7 shows that the logical models closely match the DNN outputs across all settings, thereby providing strong empirical evidence for the high fidelity of the interaction-based explanations. Please see Appendix G for the setting of ϵ in Equation (2).

B Implementation and optimization of the transferability-boosted loss

Directly minimizing the loss $\mathcal{L}_{\text{trans}}$ defined in Equation (6) involves optimization challenges, so we propose an approximate yet differentiable version of this loss for implementation. This appendix first provides implementation and optimization details for the loss $\mathcal{L}_{\text{trans}}$. We then present a theoretical

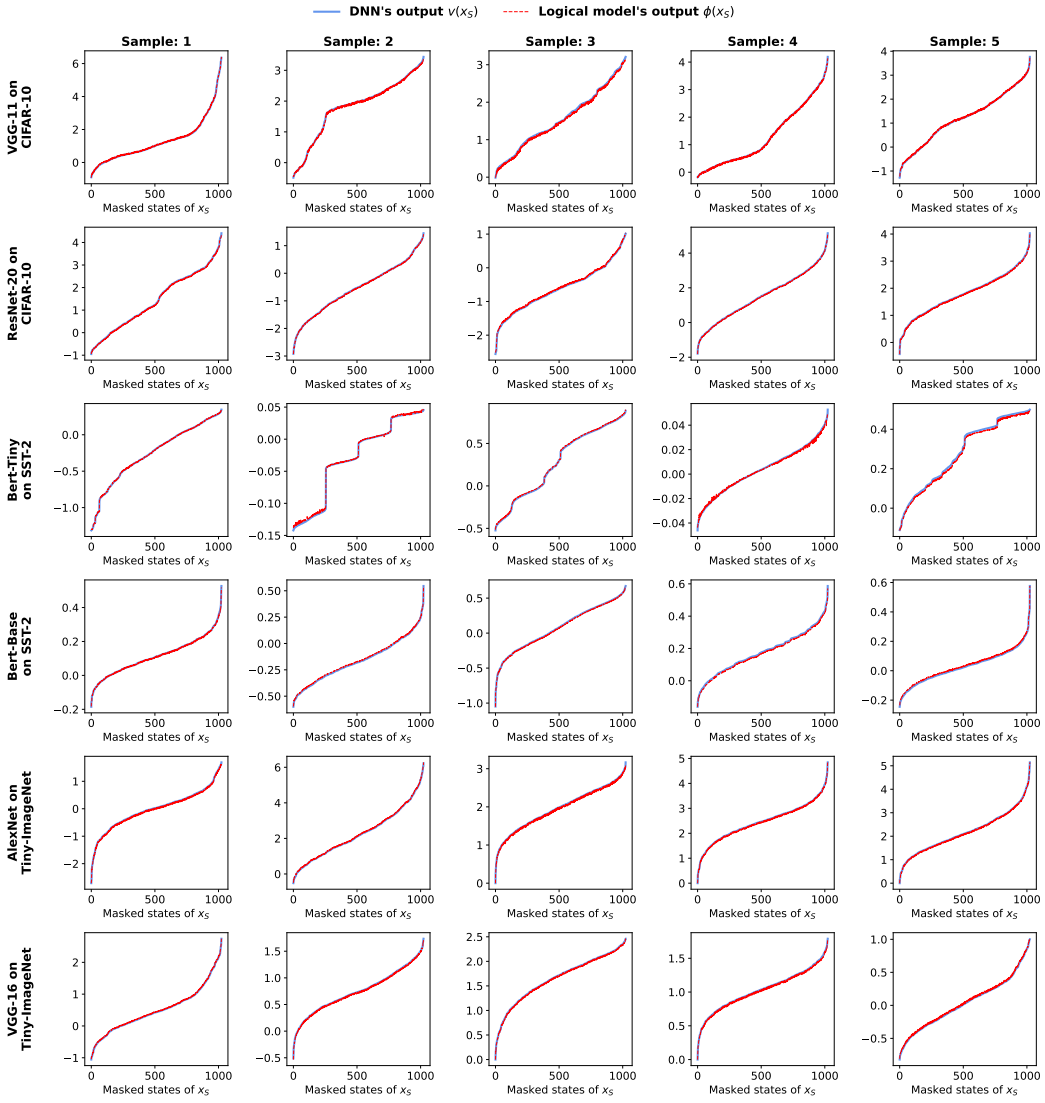

Figure 7: We compare the outputs of the logical model (red curves in each figure) with those of the DNN (blue curves in each figure) for each randomly selected input sample. The shape of the DNN function $v(\cdot)$ is well matched by the corresponding logical model function $\phi(\cdot)$ over all 2^n different masked states in Ψ.

analysis explaining why the approximate yet differentiable version of $\mathcal{L}_{\text{trans}}$ encourages different DNNs to encode transferable (similar) interaction patterns.

B.1 Implementation and optimization details

The objective in Equation (6) directly eliminates non-transferable interactions from v_1, measured *w.r.t.* the DNN v_2 (see Equation (3)), but it is difficult to optimize. To address this, we jointly train two DNNs, v_1 and v_2, and encourage them to encode similar (transferable) interactions, *i.e.,* $\forall S \subseteq N, \forall \text{type} \in \{\text{and}, \text{or}\}, I^{\text{type}}_{S,v_1}(\boldsymbol{x}) \sim I^{\text{type}}_{S,v_2}(\boldsymbol{x})$, where $I^{\text{type}}_{S,v_1}(\boldsymbol{x})$ and $I^{\text{type}}_{S,v_2}(\boldsymbol{x})$ denote the AND/OR interactions extracted by the DNN v_1 and the DNN v_2, respectively. Besides, instead of directly penalizing $\mathbb{E}_{S\subseteq N}\sum_{\text{type}\in\{\text{and,or}\}}[I^{\text{type}}_{S,v_1}(\boldsymbol{x}) - I^{\text{type}}_{S,v_2}(\boldsymbol{x})]^2$, which incurs computational cost of $\mathcal{O}(2^{|N|})$, we propose to train both v_1 and v_2 with the following loss function $\hat{\mathcal{L}}_{\text{trans}}$. **Theoretical analysis in Appendix B.2 demonstrates that minimizing $\hat{\mathcal{L}}_{\text{trans}}$ is an approximate-yet-efficient version of the loss $\mathcal{L}_{\text{trans}}$ to encourage the DNN v_1 and the DNN v_2 to encode similar interactions.**

$$\hat{\mathcal{L}}_{\text{trans}}(v_1, v_2) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\in D}\mathbb{E}_{S_1,S_2,S_3}\left[\left(\mathcal{I}_{v_1}(S_1, S_2|S_3, \boldsymbol{x}) - \mathcal{I}_{v_2}(S_1, S_2|S_3, \boldsymbol{x})\right)^2\right],$$
$$\text{s.t.} \quad \begin{cases} \mathcal{I}_{v_1}(S_1, S_2|S_3, \boldsymbol{x}) = v_1(\boldsymbol{x}_{S_1\cup S_2\cup S_3}) - v_1(\boldsymbol{x}_{S_1\cup S_3}) - v_1(\boldsymbol{x}_{S_2\cup S_3}) + v_1(\boldsymbol{x}_{S_3}), \\ \mathcal{I}_{v_2}(S_1, S_2|S_3, \boldsymbol{x}) = v_2(\boldsymbol{x}_{S_1\cup S_2\cup S_3}) - v_2(\boldsymbol{x}_{S_1\cup S_3}) - v_2(\boldsymbol{x}_{S_2\cup S_3}) + v_2(\boldsymbol{x}_{S_3}). \end{cases} \tag{14}$$

where $S_1, S_2, S_3 \subseteq N$ are three disjoint subsets of the input variables, subject to $S_1\cap S_2 = S_1\cap S_3 = S_2\cap S_3 = \emptyset$. $\mathcal{I}_{v_1}(S_1, S_2|S_3, \boldsymbol{x})$ represents the simplified interaction (*i.e.,* non-linear relationship) between input variables in S_1 and those in S_2, extracted by the target DNN v_1 from the given input sample $\boldsymbol{x}$, conditioned on the variables in S_3. Similarly, $\mathcal{I}_{v_2}(S_1, S_2|S_3, \boldsymbol{x})$ denotes the simplified interaction extracted by the DNN v_2. $\mathcal{L}_{\text{CE}}$ denotes the cross-entropy loss. $(\boldsymbol{x}, \boldsymbol{y})$ denotes a training sample and its corresponding label from the dataset $\mathcal{D}$.

The loss based on the simplified interaction can be computed with significantly lower computational cost. This is because in real-world applications, it is not necessary to exhaustively enumerate all possible divisions of the three subsets (S_1, S_2, S_3), subject to $S_1, S_2, S_3 \subseteq N, S_1\cap S_2 = S_1\cap S_3 = S_2\cap S_3 = \emptyset$. Instead, we can simply sample a random triplet of (S_1, S_2, S_3) to compute the loss $\hat{\mathcal{L}}_{\text{trans}}$ on the input sample $\boldsymbol{x}$ in each training batch. This significantly improves computational efficiency. More importantly, Figure 3 shows that optimizing the approximate loss substantially suppresses non-transferable interactions and increases the interaction transferability ratio γ.

B.2 Theoretical justification: why the approximate transferability-boosted loss encourages two DNNs to encode similar interactions

To justify why the approximate transferability-boosted loss $\hat{\mathcal{L}}_{\text{trans}}$ encourages two DNNs v_1 and v_2 to encode similar interactions, we analyze the structure of the simplified interaction term $\mathcal{I}_{v_1}(S_1, S_2|S_3, \boldsymbol{x})$, and how the approximate transferability-boosted loss term,

$$\mathbb{E}_{S_1,S_2,S_3}\left[\left(\mathcal{I}_{v_1}(S_1, S_2|S_3, \boldsymbol{x}) - \mathcal{I}_{v_2}(S_1, S_2|S_3, \boldsymbol{x})\right)^2\right], \tag{15}$$

encourages the two DNNs v_1 and v_2 to encode similar interactions.

B.2.1 Decomposition of simplified AND interactions

For a fixed input $\boldsymbol{x}$, we consider the simplified interaction between variable sets S_1 and S_2 given S_3, extracted by the DNN v_1, defined as:

$$\mathcal{I}_{v_1}(S_1, S_2 \mid S_3, \boldsymbol{x}) = v_1(\boldsymbol{x}_{S_1\cup S_2\cup S_3}) - v_1(\boldsymbol{x}_{S_1\cup S_3}) - v_1(\boldsymbol{x}_{S_2\cup S_3}) + v_1(\boldsymbol{x}_{S_3}). \tag{16}$$

In this subsection, we focus only on the AND component of v_1 (the process for the OR interaction follows the same logic). According to the decomposition of AND interactions, we define:

$$v_{1,\text{and}}(\boldsymbol{x}_S) := \sum_{\emptyset\neq T\subseteq S} I^{\text{and}}_{T,v_1}. \tag{17}$$

Then, the simplified interaction contributed by the AND component is:

$$\begin{aligned}\mathcal{I}^{\text{and}}_{v_1}(S_1, S_2 \mid S_3, \boldsymbol{x}) &= v_{1,\text{and}}(\boldsymbol{x}_{S_1\cup S_2\cup S_3}) - v_{1,\text{and}}(\boldsymbol{x}_{S_1\cup S_3}) \\ &\quad - v_{1,\text{and}}(\boldsymbol{x}_{S_2\cup S_3}) + v_{1,\text{and}}(\boldsymbol{x}_{S_3}) \\ &= \sum_{\emptyset\neq T\subseteq S_1\cup S_2\cup S_3} I^{\text{and}}_{T,v_1} - \sum_{\emptyset\neq T\subseteq S_1\cup S_3} I^{\text{and}}_{T,v_1} \\ &\quad - \sum_{\emptyset\neq T\subseteq S_2\cup S_3} I^{\text{and}}_{T,v_1} + \sum_{\emptyset\neq T\subseteq S_3} I^{\text{and}}_{T,v_1}.\end{aligned} \tag{18}$$

Equivalently, we can collect the coefficient of each interaction term I^{and}_{T,v_1}:

$$\begin{aligned}\mathcal{I}^{\text{and}}_{v_1}(S_1, S_2 \mid S_3, \boldsymbol{x}) = \sum_{\emptyset\neq T\subseteq S_1\cup S_2\cup S_3} \Big[1 - \mathbf{1}(T\subseteq S_1\cup S_3) - \mathbf{1}(T\subseteq S_2\cup S_3) \\ + \mathbf{1}(T\subseteq S_3)\Big] I^{\text{and}}_{T,v_1}.\end{aligned} \tag{19}$$

For any non-empty subset $T \subseteq S_1 \cup S_2 \cup S_3$, the coefficient

$$1 - \mathbf{1}(T\subseteq S_1\cup S_3) - \mathbf{1}(T\subseteq S_2\cup S_3) + \mathbf{1}(T\subseteq S_3) \tag{20}$$

equals 1 if and only if T intersects both S_1 and S_2, and equals 0 otherwise. Therefore, we obtain:

$$\mathcal{I}^{\text{and}}_{v_1}(S_1, S_2 \mid S_3, \boldsymbol{x}) = \sum_{\substack{\emptyset\neq T\subseteq S_1\cup S_2\cup S_3 \\ T\cap S_1\neq\emptyset,\, T\cap S_2\neq\emptyset}} I^{\text{and}}_{T,v_1}. \tag{21}$$

This result shows that the AND part of the simplified interaction aggregates all AND-interaction terms whose variable set intersects both S_1 and S_2. It represents the simplified interaction (*i.e.,* non-linear relationship) between input variables in S_1 and those in S_2, extracted by v from the given input sample $\boldsymbol{x}$, conditioned on the variables in S_3. Similarly, $\mathcal{I}^{\text{and}}_{v_2}(S_1, S_2|S_3, \boldsymbol{x})$ denotes the simplified interaction extracted by the DNN v_2.

B.2.2 Effect of the approximate transferability-boosted loss

The approximate transferability-boosted loss $\hat{\mathcal{L}}_{\text{trans}}$ penalizes the squared difference between the simplified interactions extracted by v and v' across many randomly sampled disjoint subsets (S_1, S_2, S_3). Concretely,

$$\mathbb{E}_{S_1,S_2,S_3}\left[(\mathcal{I}_v(S_1, S_2|S_3, \boldsymbol{x}) - \mathcal{I}_{v'}(S_1, S_2|S_3, \boldsymbol{x}))^2\right]$$

acts as a regularizer that encourages the two networks to encode similar interactions.

Therefore, by minimizing the approximate transferability-boosted loss over numerous tuples (S_1, S_2, S_3), the two DNNs are encouraged to encode similar interactions between input variables in S_1 and those in S_2, conditioned on the variables in S_3. More importantly, Figure 3 shows that optimizing the approximate loss substantially suppresses non-transferable interactions and increases the interaction transferability ratio γ.

C Validation of the sparsity of interactions

Ren et al. (2024a) have proved three sufficient conditions for the sparsity of AND interactions. Accordingly, in this section, we first revisit the three conditions and then empirically validate the sparsity of interactions.

C.1 Common conditions for sparse interactions

Condition 1. *The DNN does not encode extremely high-order interactions:* $\forall\, T \in \{T \subseteq N||T| \geq M+1\},\ I^{\text{and}}_T = 0$.

Condition 1 is common because extremely high-order interactions usually represent very complex and over-fitted patterns, which are unlikely to be learned by a well-trained DNN in real scenarios.

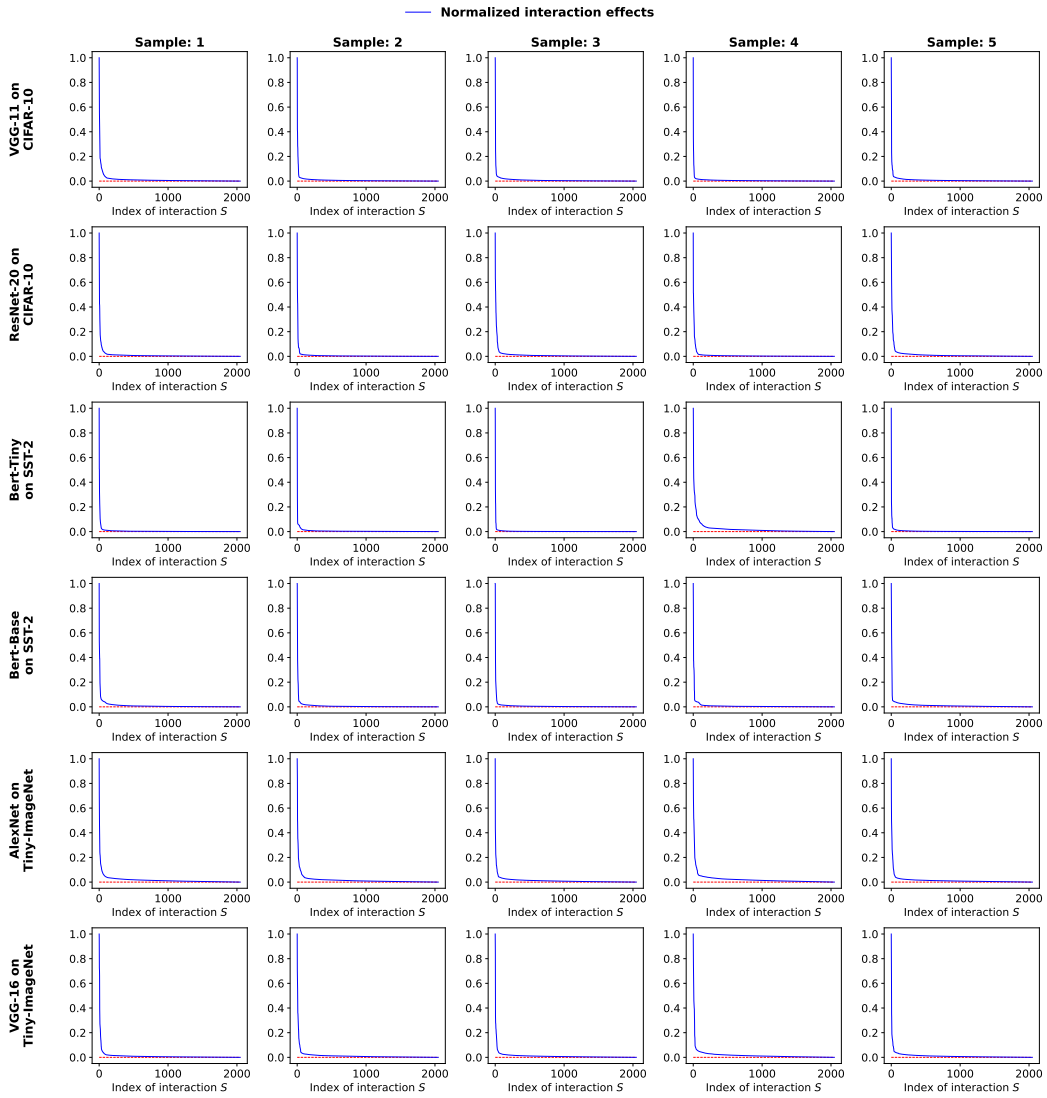

Figure 8: Normalized strength of different interactions, shown in descending order. The logical models encode only a small number of salient interactions for each input sample.

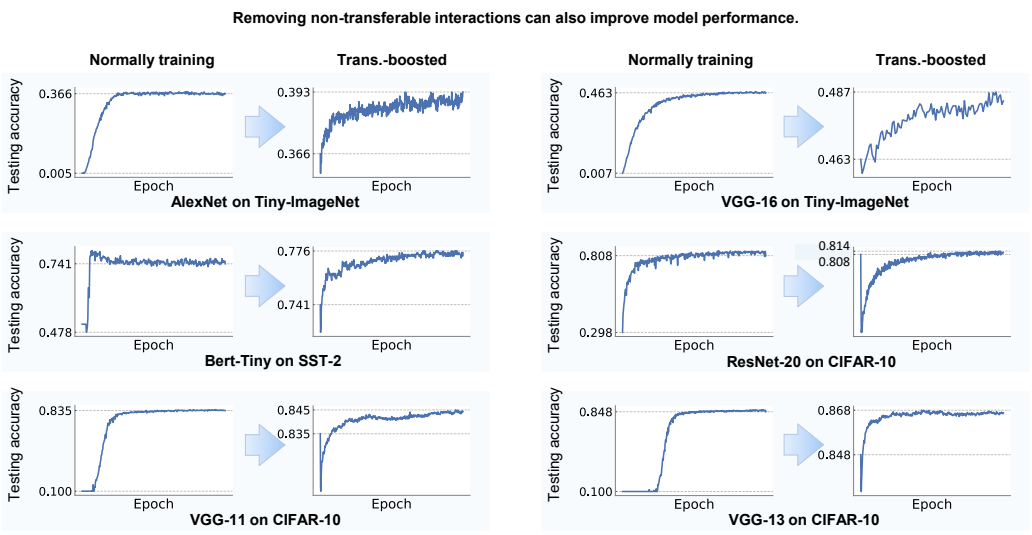

Figure 9: Comparing the testing accuracy between the originally overfitted DNN (left in each pair) and its transferability-boosted counterpart, which is obtained by subsequent fine-tuning the overfitted DNN with $\mathcal{L}$ (right in each pair). Removing non-transferable interactions can improve model performance.

Condition 2. *Let $\bar{u}^{(k)} \overset{\text{def}}{=} \mathbb{E}_{|S|=k}[v(\boldsymbol{x}_S) - v(\boldsymbol{x}_\emptyset)]$ denote the average classification confidence of the DNN over all masked samples $\boldsymbol{x}_S$ with k unmasked input variables. This average classification confidence monotonically increases when k increases: $\forall\, k' \le k$, $\bar{u}^{(k')} \le \bar{u}^{(k)}$.*

Condition 2 implies that a well-trained DNN is likely to have higher average classification confidence for less masked input samples.

Condition 3. *Given the average classification confidence $\bar{u}^{(k)}$ of samples with k unmasked input variables, there is a polynomial lower bound for the average classification confidence with $k'(k' \le k)$ unmasked input variables: $\forall\, k' \le k$, $\bar{u}^{(k')} \ge (\frac{k'}{k})^p\, \bar{u}^{(k)}$, where $p > 0$ is a constant.*

Condition 3 suggests that the classification confidence of the DNN remains relatively stable even when presented with masked input samples. In real-world applications, the classification or detection of masked or occluded samples frequently occurs. As a result, a well-trained DNN typically develops the ability to classify such masked inputs by leveraging local information, which can be derived from the visible portions of the input. Consequently, the model should not produce a substantially reduced confidence score for masked samples.

C.2 Empirical validation

Given an input sample, the preceding subsection presents sufficient conditions for interaction sparsity. Detailed proofs are provided in Ren et al. (2024a). In this subsection, we further present empirical evidence demonstrating that interactions are sparse in practice.

Specifically, we randomly select a subset of input samples and extract the interactions encoded by the DNNs for these samples. Following Ren et al. (2024a), we normalize the interaction strengths and sort them in descending order. We conduct experiments by training VGG-11 and ResNet-20 on the CIFAR-10 dataset, AlexNet and VGG-16 on the Tiny-ImageNet dataset, and Bert-Base and Bert-Tiny on the SST-2 dataset. For each dataset, we randomly select five samples and analyze the normalized interaction distributions for each sample.

As shown in Figure 8, the logical models encode only a small number of salient interactions for each input sample; that is, only a few interactions exhibit large effects, while the vast majority have effects close to zero. These results provide strong empirical evidence for the sparsity of interactions.

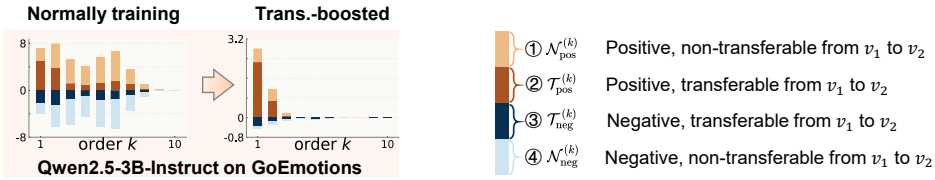

Figure 10: Interaction distributions extracted from Qwen2.5-3B-Instruct on GoEmotions before (left) and after (right) additional fine-tuning with $\mathcal{L}$.

D PRACTICAL VALUE

Improving the performance of models in practice. In fact, we find that our proposed loss function $\mathcal{L}$, designed to eliminate non-transferable interactions, not only mitigates overfitting in DNNs but also improves model performance. Specifically, Figure 9 shows both the testing accuracy achieved during normal training with cross-entropy loss and the testing accuracy after further training with our loss function $\mathcal{L}$. *The results show that our loss function improves model performance.*

Providing significant insights into identifying the precise point of early stopping in DNNs. This paper establishes a clear connection between a DNN's overfitting and the generalizability of interactions. Consequently, interaction-based explanation may provide a principled way to determine the optimal early stopping point in a DNN's training process by objectively explaining how the generalizability of encoded interactions evolves. Specifically, preliminary experiments in Figure 2 show that the dynamics of generalizable interactions (*i.e.*, $\mathcal{T}_{\text{pos}}^{(k)}$ and $\mathcal{T}_{\text{neg}}^{(k)}$) during both the normal training phase (when the training-testing loss gap is relatively small) and the overfitting phase (when the training-testing loss gap begins to widen). During the normal training phase, the generalizability of interactions gradually increases; however, once the DNN enters the overfitting phase, it begins to encode non-generalizable interactions, which serve as a clear signal of overfitting. Therefore, *by tracking the evolution of interaction generalizability throughout the entire training process, one can stop training at the point where interaction generalizability peaks, offering guidance for early stopping from the perspective of the detailed inference patterns encoded by the DNN.*

E MORE EXPERIMENTAL RESULTS

E.1 ADDITIONAL RESULTS ON A LARGER LANGUAGE MODEL

We further examine the relationship between interaction generalizability and overfitting on Qwen2.5-3B-Instruct (Team, 2024). We fine-tune the model on the GoEmotions dataset (Demszky et al., 2020) using LoRA (Hu et al., 2021).

Table 1: Qwen2.5-3B-Instruct on GoEmotions before and after additional fine-tuning with $\mathcal{L}$. The loss gap is test CE minus training CE.

Checkpoint	Interaction transferability ratio (%)	Training CE	Testing CE	Loss gap
Before applying $\mathcal{L}$	22.44	0.1851	0.5393	+0.3542
After applying $\mathcal{L}$	73.98	0.3843	0.4186	+0.0343

We first fine-tune the model until it becomes overfitted. At this checkpoint, the training and testing cross-entropy losses are 0.1851 and 0.5393, respectively, giving a loss gap of 0.3542. Starting from the same checkpoint, we continue fine-tuning the model with our proposed loss $\mathcal{L}$ to penalize non-transferable interactions. We then compare the model before and after this additional fine-tuning.

Figure 10 shows the interaction distributions at the two checkpoints. It shows that the proposed loss $\mathcal{L}$ effectively eliminated non-transferable interactions. As shown in Table 1, the ratio of transferable interactions increases from 22.44% to 73.98%. At the same time, the testing loss decreases from 0.5393 to 0.4186, while the training loss increases from 0.1851 to 0.3843. Consequently, the testing-

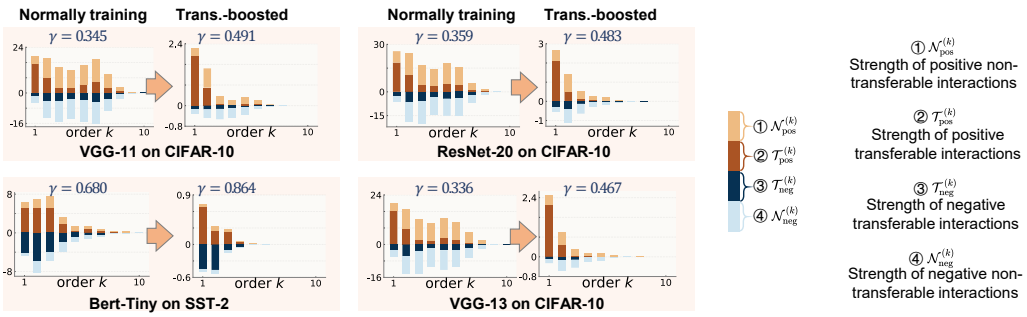

Figure 11: Comparing the interaction distribution[4] between the originally overfitted DNN (left in each pair) and its transferability-boosted counterpart, which is obtained by subsequent fine-tuning the overfitted DNN with $\mathcal{L}$ (right in each pair). We use the metric γ in Equation (5) to measure the transferability of interactions between v_1 and v_2, and γ is enhanced by eliminating non-transferable interactions while preserving transferable interactions. We set $\lambda = 0.8$ in this figure; please see Appendix E.2 for additional results under other choices of λ.

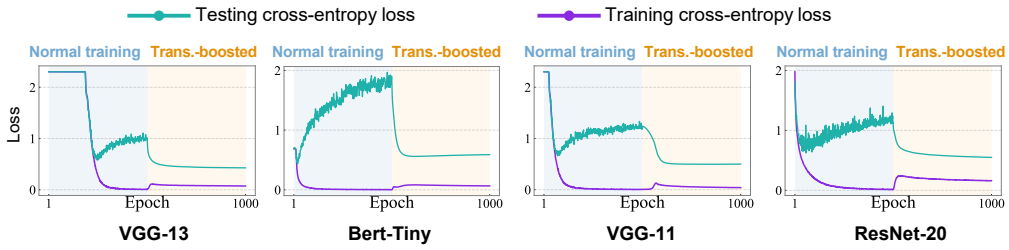

Figure 12: Comparing the training-testing cross-entropy loss gap between an overfitted DNN and its transferability-boosted counterpart, which is obtained by further fine-tuning the overfitted DNN using the loss $\mathcal{L}$. Eliminating non-transferable interactions significantly reduces the loss gap. In this figure, we set $\lambda = 0.3$.

training loss gap decreases from 0.3542 to 0.0343. These results show that the reduction in the loss gap is accompanied by both improved testing loss and a higher ratio of transferable interactions.

E.2 More results on the effectiveness of the proposed loss in eliminating non-transferable interactions

In the main paper, we present results with a fixed parameter $\lambda = 0.3$ to illustrate how the proposed loss $\mathcal{L}$ improves interaction transferability by suppressing non-transferable interactions while preserving transferable ones. In this appendix, we provide additional experimental results under different choices of λ to further demonstrate the robustness and consistency of this effect.

As Figure 11 shows, we set $\lambda = 0.8$ and analyze the resulting interaction distributions after fine-tuning overfitted DNNs with the proposed loss. Across all settings, we observe a consistent reduction in non-transferable interactions, including both positively and negatively contributing ones (*i.e.*, $\mathcal{N}_{\text{pos}}^{(k)}$ and $\mathcal{N}_{\text{neg}}^{(k)}$), accompanied by the preservation of transferable interactions (*i.e.*, $\mathcal{T}_{\text{pos}}^{(k)}$ and $\mathcal{T}_{\text{neg}}^{(k)}$).

As a result, the average interaction transferability γ consistently increases. These results suggest that the effectiveness of $\mathcal{L}$ is not sensitive to a particular hyperparameter choice.

Overall, the additional results in this appendix further support the conclusion that the proposed loss efficiently reshapes the interaction distribution of DNNs toward more transferable structures.

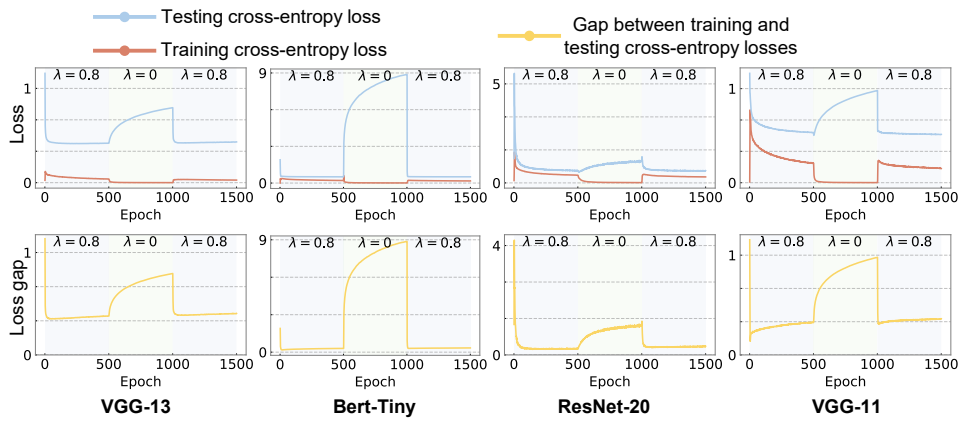

Figure 13: The cross-entropy loss curves on the training and testing sets. We manipulated the training process by temporarily disabling the elimination of non-transferable interactions between the 501st and 1000th epochs by setting $\lambda = 0$. We found that eliminating non-transferable interactions effectively controlled the testing-training cross-entropy loss gap, preventing it from continuously widening. However, whenever the elimination was suspended, the loss gap gradually widened.



Figure 14: Only our method $\mathcal{L}$ successfully reduced the training-testing cross-entropy loss gap, whereas the competing methods (*i.e.,* $\mathcal{L}_z$ and $\mathcal{L}_f$) failed to achieve such an effect.

E.3 Additional results on reducing overfitting by eliminating non-transferable interactions

Figure 12 compares the training-testing cross-entropy loss gap between an originally overfitted DNN and its transferability-boosted counterpart, which is obtained by further fine-tuning the overfitted model using the proposed loss $\mathcal{L}$. Consistent with our analysis, eliminating non-transferable interactions leads to a substantial reduction in the generalization gap, indicating improved robustness and reduced overfitting.

In these experiments, we set $\lambda = 0.3$, and the reduction in the loss gap is clear and consistent across models. These results reinforce the conclusion that the proposed loss effectively mitigates overfitting by systematically eliminating non-transferable interactions.

E.4 Further evidence for the connection between non-transferable interactions and the DNN's overfitting

In this subsection, we provide additional empirical evidence supporting the connection between non-transferable interactions and DNN overfitting by examining the temporal response of overfitting to the application of our proposed loss $\mathcal{L}$.

Specifically, we conducted an experiment to examine the effect of intermittently eliminating non-transferable interactions in the following three phases. (1) Phase 1: non-transferable interactions were eliminated by setting $\lambda = 0.8$ in the 1st-500th epochs. (2) Phase 2: elimination was suspended by

setting $\lambda = 0$ in the 501st-1000th epochs. (3) Phase 3: elimination was resumed by resetting $\lambda = 0.8$ in the 1001st-1500th epochs. We conducted experiments to examine whether the loss gap increased when elimination was suspended and continued to decrease when elimination resumed. Specifically, we trained ResNet-20, VGG-11, and VGG-13 on the CIFAR-10 dataset, as well as Bert-Tiny on the SST-2 dataset.

As shown in Figure 13, whenever the transferability-boosted loss was applied (*i.e.,* setting $\lambda = 0.8$), the training-testing cross-entropy loss gap continuously decreased until convergence. Conversely, whenever the transferability-boosted loss was removed (*i.e.,* setting $\lambda = 0$), the testing cross-entropy loss increased while the training cross-entropy loss approached zero. *These results further indicated that eliminating non-transferable interactions effectively mitigated the degree of a DNN's overfitting.*

E.5 Comparison of different consistency losses on reducing the training-testing loss gap

In this subsection, we empirically compared the effects of different consistency-based loss functions on reducing the training-testing cross-entropy loss gap. Specifically, we evaluated three losses: the proposed loss $\mathcal{L}$, and two feature-consistency losses $\mathcal{L}_{\text{z}}$ and $\mathcal{L}_{\text{f}}$.

For each method, we started from the same overfitted DNN and performed additional fine-tuning using the corresponding loss. We then measured the resulting training-testing cross-entropy loss gap.

The results, shown in Figure 14, indicated that only the proposed loss ($\mathcal{L}$) led to a reduction in the loss gap. In contrast, models fine-tuned with $\mathcal{L}_{\text{z}}$ or $\mathcal{L}_{\text{f}}$ exhibited loss gaps comparable to that of the original overfitted model. These results indicated that *the elimination of non-transferable interactions was the most effective way to mitigate overfitting, while other forms of feature consistency did not have this effect.*

Table 2: Comparison with standard regularization baselines on CIFAR-10 over four random seeds. Results are reported as mean $\pm$ standard deviation. The generalization gap is the testing cross-entropy loss minus the training cross-entropy loss.

Model	**Method**	**Test accuracy (%, ↑)**	**Generalization gap (↓)**
VGG-11	No dropout & no weight decay	84.23 ± 0.28	1.2131 ± 0.0646
	Weight decay only	84.28 ± 0.22	1.1437 ± 0.1245
	Dropout only	84.55 ± 0.17	1.2202 ± 0.0358
	Our method	$\mathbf{84.86 \pm 0.10}$	$\mathbf{0.2974 \pm 0.0027}$
VGG-13	No dropout & no weight decay	86.21 ± 0.26	1.1176 ± 0.0432
	Weight decay only	86.35 ± 0.28	1.1155 ± 0.0793
	Dropout only	86.53 ± 0.17	1.0000 ± 0.0267
	Our method	$\mathbf{87.17 \pm 0.20}$	$\mathbf{0.2489 \pm 0.0043}$

E.6 Comparison with standard regularization baselines

We provide a direct quantitative comparison with standard regularization baselines on CIFAR-10 using VGG-11 and VGG-13, including: (1) no dropout and no weight decay, (2) dropout only, and (3) weight decay only. All methods are evaluated using the same model architectures, training settings, and datasets as in our main experiments, ensuring a fair comparison. Each setting is evaluated over four random seeds.

Table 2 reports the mean and standard deviation of testing accuracy and the generalization gap, where the latter is measured as the difference between the testing loss and the training loss. Our method achieves the highest mean testing accuracy for both architectures, exceeding the strongest regularization baseline by 0.31 and 0.64 percentage points on VGG-11 and VGG-13, respectively. It also substantially reduces the mean generalization gap for both architectures. These results show that the reduction in the generalization gap is accompanied by improved mean testing accuracy. This suggests that our method is effective in mitigating overfitting without sacrificing predictive performance. These results further support the effectiveness of our method.

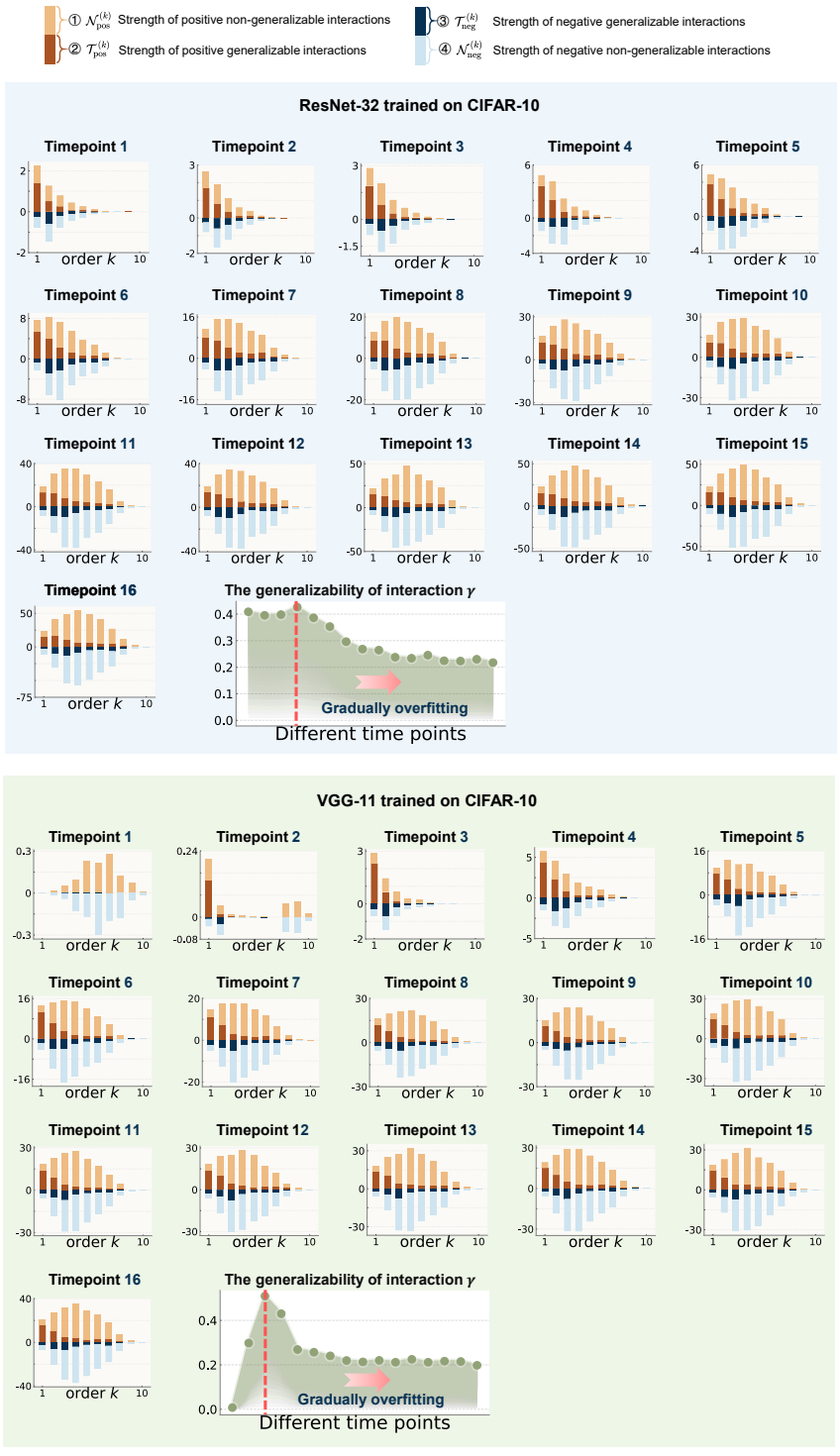

Figure 15: *Fine-grained evolution of interaction distributions.* We visualize interaction distributions at additional training time points. During the overfitting phase, the DNN gradually learns a large number of non-generalizable interactions, leading to a decrease in interaction generalizability.

E.7 FINE-GRAINED EVOLUTION OF INTERACTION DISTRIBUTIONS

We further provide more fine-grained visualizations of interaction distributions at additional training time points. These results complement the analysis in the main paper by showing how interaction patterns evolve during both the normal training phase and the overfitting phase.

Figure 15 shows that during the overfitting phase, the DNN gradually encodes an increasing number of non-generalizable interactions, including both positive and negative non-generalizable interactions, denoted by $\mathcal{N}_{\text{pos}}^{(k)}$ and $\mathcal{N}_{\text{neg}}^{(k)}$, respectively. As a result, the overall generalizability of interactions, measured by γ, consistently decreases. These fine-grained results further support our conclusion that overfitting is accompanied by the accumulation of non-generalizable interaction patterns.

E.8 STABILITY ACROSS RANDOM SEEDS

To evaluate the robustness of our observations, we repeat the experiments with four different random seeds and report the mean and standard deviation of the results. As shown in Tables 3, 4, and 5, the improvements brought by our method remain consistent across different random initializations.

Table 3 evaluates whether our method consistently improves the transferability of learned interactions. Across all evaluated architectures, the transferability-boosted DNNs achieve higher interaction transferability ratios than normally trained DNNs, demonstrating that the elimination of non-transferable interactions is robust across random seeds.

Table 4 further evaluates whether improving interaction transferability leads to more generalizable interactions across DNNs trained on disjoint data partitions. The transferability-boosted DNNs consistently achieve higher generalizability ratios than normally trained DNNs. Moreover, the shared interactions extracted from transferability-boosted DNNs exhibit even higher generalizability ratios, supporting our observation that transferable interactions are more likely to generalize across different DNNs.

Finally, Table 5 compares our method with output-consistency and feature-consistency competing methods. Across all evaluated models, our loss $\mathcal{L}$ consistently achieves the highest ratio of transferable interactions, indicating that the advantage of interaction transferability remains stable under different random seeds.

Table 3: Transferability ratio of interactions across four random seeds. Values are reported as mean $\pm$ standard deviation.

Model	Normal training	Transferability boosted
AlexNet	0.2912 ± 0.0236	0.4883 ± 0.0390
Bert-Base	0.3941 ± 0.0661	0.5454 ± 0.0534
Bert-Medium	0.2849 ± 0.0671	0.6548 ± 0.0168
ResNet-20	0.3119 ± 0.0145	0.3526 ± 0.0122
ResNet-32	0.2971 ± 0.0177	0.3815 ± 0.0248
Bert-Tiny	0.4582 ± 0.0294	0.5273 ± 0.0035
VGG-11	0.3107 ± 0.0216	0.4109 ± 0.0113
VGG-13	0.3451 ± 0.0060	0.4202 ± 0.0398
VGG-16	0.3310 ± 0.0065	0.6347 ± 0.0601

Table 4: Generalizability ratio of interactions across four random seeds. Values are reported as mean $\pm$ standard deviation.

Setting	Normal training	Transferability boosted	Shared interactions
VGG-11 on CIFAR-10	0.1583 ± 0.0178	0.3314 ± 0.0384	0.6389 ± 0.0109
Bert-Tiny on SST-2	0.4042 ± 0.0105	0.6213 ± 0.0121	0.6678 ± 0.0126
AlexNet on Tiny-ImageNet	0.1307 ± 0.0063	0.4036 ± 0.0157	0.5502 ± 0.0141

Table 5: Transferability ratio of different methods across four random seeds. Values are reported as mean $\pm$ standard deviation.

Model	Normal	$\mathcal{L}_z$	$\mathcal{L}_f$	$\mathcal{L}$ (Ours)
ResNet-20	0.3119 ± 0.0145	0.2930 ± 0.0221	0.2969 ± 0.0105	0.3526 ± 0.0122
ResNet-32	0.2971 ± 0.0177	0.3165 ± 0.0131	0.2778 ± 0.0095	0.3815 ± 0.0248
VGG-11	0.3107 ± 0.0216	0.3025 ± 0.0031	0.3218 ± 0.0052	0.4109 ± 0.0113
VGG-13	0.3451 ± 0.0060	0.3199 ± 0.0043	0.3227 ± 0.0118	0.4202 ± 0.0398

E.9 Examples of AND-OR logical models explaining LLMs

In this section, to demonstrate that AND-OR logical models can faithfully explain large language models, we present examples showing that, given specific input prompts, AND-OR logical models can be constructed to faithfully explain both the DeepSeek-r1-distill-llama-8B (Guo et al., 2025) and Qwen2.5-7B (Team, 2024) models, hereafter referred to as the DeepSeek and Qwen models, respectively. Figure 16 illustrates the AND-OR logical models used to explain DeepSeek and Qwen under different input prompts.

F Experimental detail

F.1 Training settings

• Detailed settings of normal training. In this paper, we trained various DNNs for different tasks. Specifically, for the image classification task, we trained VGG-11/13 on the CIFAR-10 dataset (MIT license) with a learning rate of 0.01. We trained ResNet-20/32 on the CIFAR-10 dataset with a learning rate of 0.01. We trained AlexNet and VGG-16 on the Tiny-ImageNet dataset (license unknown) with a learning rate of 0.01. For the sentiment classification task, we trained the Bert-Tiny model, Bert-Medium model, and Bert-Base model on the SST-2 dataset (license unknown) with a learning rate of 0.01. All DNNs were trained using the SGD optimizer (Robbins & Monro, 1951) with a momentum of 0.9.

For models trained on the CIFAR-10 dataset, we set the batch size to 256. Due to memory constraints, for models trained on the Tiny-ImageNet dataset, we used a batch size of 64. For models trained on the SST-2 dataset, we set the batch size to 64 (except for Bert-Tiny, which also used a batch size of 256).

• Detailed settings for verifying the elimination of non-transferable interactions. To verify the effectiveness of our proposed loss function $\mathcal{L}$ in eliminating non-transferable interactions, we followed the aforementioned normal training settings. Specifically, we first trained DNNs with the standard cross-entropy loss for 300 epochs. Subsequently, we further trained these DNNs with our proposed loss function $\mathcal{L}$ for an additional 500 epochs and examined the resulting changes in the distribution of encoded interactions. Except for VGG-16, Bert-Base, and Bert-Tiny, which were trained for 100, 50, and 100 epochs, respectively, due to computational constraints, all other models were trained following the settings described above. For each network architecture, we trained two DNNs on the same dataset with differently initialized parameters, which served as two target DNNs v_1 and v_2, respectively. We kept the batch size unchanged from the standard training.

• Detailed settings for verifying that eliminating non-transferable interactions effectively reduces the training-testing cross-entropy loss gap. We first trained the DNNs with the standard cross-entropy loss for 500 epochs. We then continued training them with our proposed loss function $\mathcal{L}$, using different values of λ, for an additional 500 epochs, and monitored the changes in the testing-training cross-entropy loss. For each network architecture, we trained two DNNs on the same dataset with differently initialized parameters, which served as two target DNNs v_1 and v_2, respectively.

• Detailed settings for comparing DNNs trained using our proposed loss function $\mathcal{L}$ with DNNs trained on the two competing loss functions ($\mathcal{L}_\mathbf{z}$ and $\mathcal{L}_\mathbf{f}$). We first trained the DNNs with the standard cross-entropy loss for 300 epochs. For training with loss terms that promote consistency of output ($\mathcal{L}_z$) and intermediate features ($\mathcal{L}_f$), we continued training the models for 500 additional

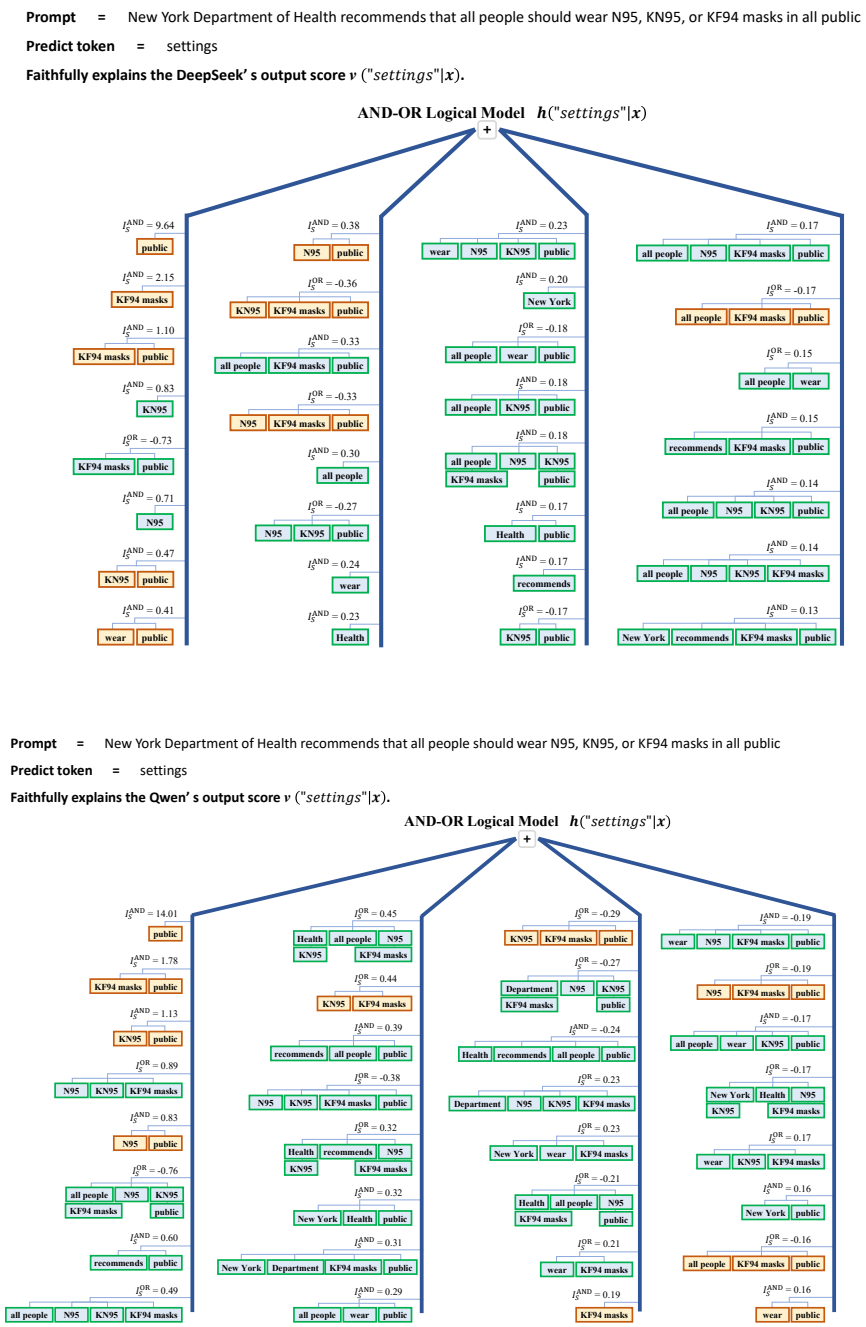

Figure 16: An example of AND-OR logical models constructed to faithfully explain the output scores of the DeepSeek model (top) and the Qwen model (bottom) on a single sample. A further example is presented on the next page.

epochs using a learning rate of 0.001. Again, we set $\lambda = 0.3$ and kept the batch size unchanged. For VGG-11, we empirically encouraged the consistency of intermediate features at the output of the 4th convolutional layer. For VGG-13, we enforced consistency at the output of the 5th convolutional layer. For ResNet-20, we promoted the consistency of intermediate features at the output of the 3rd residual block. For ResNet-32, we did so at the output of the 5th residual block. For each network architecture, we trained two DNNs on the same dataset with differently initialized parameters, which served as two target DNNs v_1 and v_2, respectively.

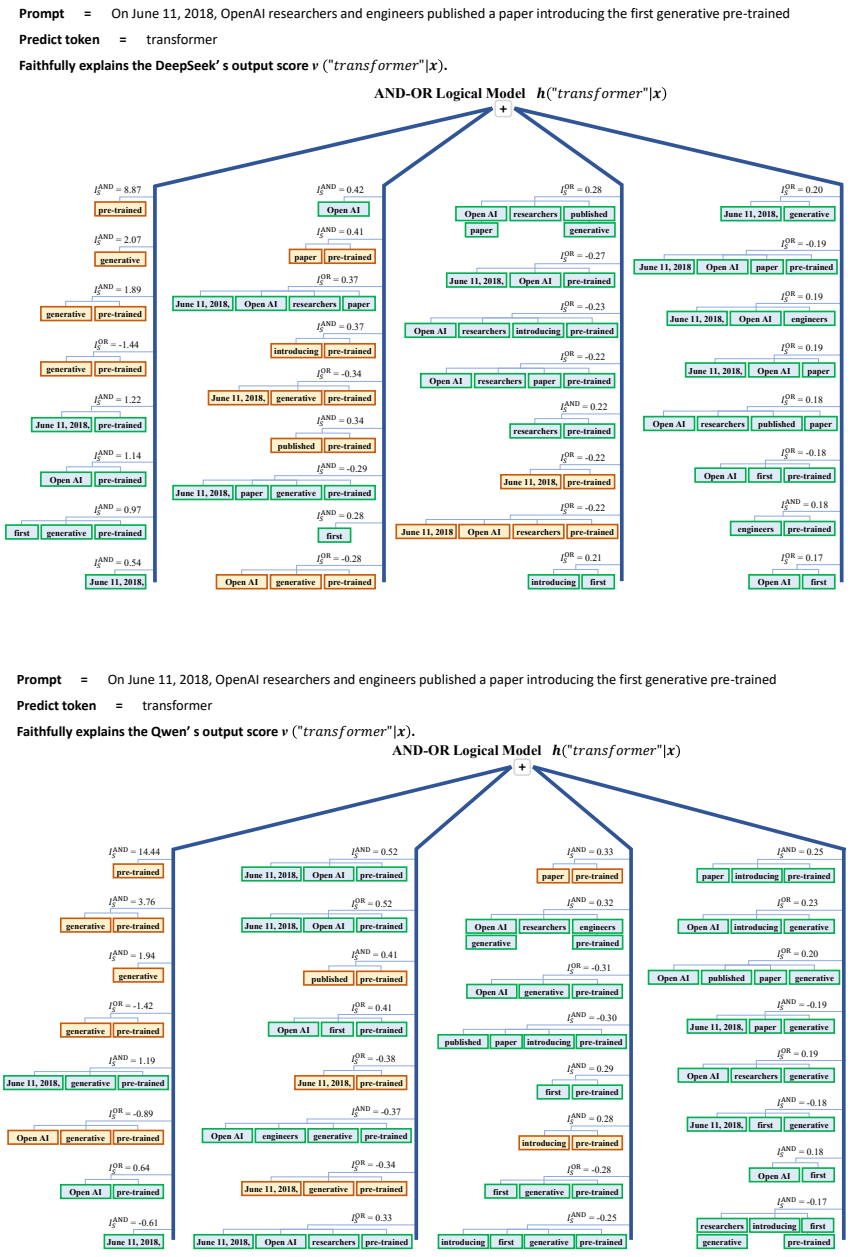

Figure 16: Another example of AND-OR logical models explaining the DeepSeek model (top) and the Qwen model (bottom) on a different sample.

F.2 METRICS FOR THE STRENGTH OF GENERALIZABLE INTERACTIONS OVER DIFFERENT ORDERS

As in Equation (4), the following four metrics measure (1) the strength of generalizable positive interactions of the k-th order $\mathcal{I}_{\text{pos}}^{(k)}$, (2) the strength of generalizable negative interactions of the k-th order $\mathcal{I}_{\text{neg}}^{(k)}$, (3) the strength of non-generalizable positive interactions $\mathcal{J}_{\text{pos}}^{(k)}$, and (4) the strength of non-generalizable negative interactions $\mathcal{J}_{\text{neg}}^{(k)}$.

$$
\begin{aligned}
\mathcal{I}_{\text{pos}}^{(k)} &= \sum_{\text{type}\in\{\text{and},\text{or}\}} \sum_{S\in\widetilde{\Omega}_{\text{GT}}^{\text{type}}:|S|=k} \max(I_S^{\text{type}}, 0), \\
\mathcal{I}_{\text{neg}}^{(k)} &= \sum_{\text{type}\in\{\text{and},\text{or}\}} \sum_{S\in\widetilde{\Omega}_{\text{GT}}^{\text{type}}:|S|=k} \min(I_S^{\text{type}}, 0), \\
\mathcal{J}_{\text{pos}}^{(k)} &= \sum_{\text{type}\in\{\text{and},\text{or}\}} \sum_{S\in\Omega^{\text{type}}\setminus\widetilde{\Omega}_{\text{GT}}^{\text{type}}:|S|=k} \max(I_S^{\text{type}}, 0), \\
\mathcal{J}_{\text{neg}}^{(k)} &= \sum_{\text{type}\in\{\text{and},\text{or}\}} \sum_{S\in\Omega^{\text{type}}\setminus\widetilde{\Omega}_{\text{GT}}^{\text{type}}:|S|=k} \min(I_S^{\text{type}}, 0).
\end{aligned} \tag{22}
$$

where $\widetilde{\Omega}_{\text{GT}}^{\text{type}} = \{S \in \Omega^{\text{type}} \cap \Omega_{\text{GT}}^{\text{type}} : I_S^{\text{type}} \cdot \sigma_{S,\text{GT}}^{\text{type}} > 0\}$. $\Omega_{\text{GT}}^{\text{and}}$ and $\Omega_{\text{GT}}^{\text{or}}$ denote the set of *ground-truth generalizable AND interactions* and the set of *ground-truth OR interactions*, respectively.

F.3 Details of constructing ground-truth generalizable interactions

To construct the ground-truth set of generalizable interactions, we require multiple DNNs trained on different data partitions and with different architectures. To this end, for each dataset (the CIFAR-10 dataset, the Tiny-ImageNet dataset, and the SST-2 dataset), we randomly partition the original training set into five mutually disjoint subsets of equal size.

One subset is used to train the *normally trained DNN* as well as the *transferability-boosted DNNs* v_1 and v_2 based on Equation (7). The remaining four subsets are each used to train an additional DNN with a distinct architecture. Specifically, for the CIFAR-10 dataset, we train ResNet-20, ResNet-32, ResNet-44, and VGG-13; for the Tiny-ImageNet dataset, we train ResNet-34, ResNet-50, VGG-13, and VGG-16; and for the SST-2 dataset, we train Bert-Mini, Bert-Small, Bert-Medium, and Bert-Base. In all cases, we follow the same protocol by training four additional models on the four disjoint subsets.

We then extract interactions from each of these four DNNs. An AND interaction S is collected into $\Omega_{\text{GT}}^{\text{and}}$ as a *ground-truth generalizable AND interaction* if it appears in more than 2 of these DNNs and exerts consistent effects on classifying the input sample $\boldsymbol{x}$ across them. Ground-truth OR interactions are identified in the same manner and collected into $\Omega_{\text{GT}}^{\text{or}}$.

F.4 Computer resources

All DNNs can be trained on a single NVIDIA A800 GPU (with 80G GPU memory), and can be finished within 48 hours. Calculating all interactions for an input sample takes an average of 30-50 seconds.

F.5 Details about how to calculate interactions for different DNNs

Interaction-based explanations have been widely applied to real-world DNNs, largely due to prior efforts (Ren et al., 2024b; Li & Zhang, 2023b) that substantially reduce the computational cost of interaction extraction. Moreover, analyses of DNN generalization typically require computing only interaction distributions rather than exhaustively enumerating all interactions. As a result, selectively computing interactions over a small number of salient input variables is usually sufficient to obtain stable interaction distributions for generalization analysis. Consequently, the computation can be handled efficiently in typical experimental settings.

Extensive experiments on large language models and image-classification models further support this observation. We find that DNNs usually attend to only a small number of salient text segments or image regions when making predictions. As a result, explanations based on fewer than 10 salient text segments or image regions are often sufficient to faithfully capture the model's inference patterns. This empirical property significantly improves the computational efficiency of interaction-based explanations and ensures their practical applicability to large-scale models and datasets.

• **For image data in different image datasets,** for the above reasons, we followed (Ren et al., 2024b; Li & Zhang, 2023b) and adopted a sampling-based approximation method to extract AND-OR

interactions. Specifically, we considered the feature map after the low layer as intermediate-layer features of DNNs. We uniformly split the central region of each intermediate-layer feature (*i.e.,* we did not consider the pixels on the edges of an image) into 5×5 patches and selected the 10 patches with the highest L1 norms (*i.e.,* the brightest ones) to calculate interactions, and considered these patches as input variables for each intermediate-layer feature. We used $\mathbf{0}$ as a baseline value to mask the variables in $N \backslash T$.

• **For natural language data in SST-2 dataset,** we considered the outputs of the low-layer corresponding to input words as input features. We considered the embeddings corresponding to input features as input variables for each input sentence, and we randomly sampled 10 words, which must have a specific meaning and not be stop words, to calculate interactions. We used the average embedding over different input variables to mask the tokens in $N \backslash T$.

To measure the generalizability of interactions extracted from a given input sample $\boldsymbol{x}$, we select ten input variables once and use the same image patch locations or word positions, with identical indexing, across all DNNs compared on $\boldsymbol{x}$.

Algorithm 1 Compute AND and OR Interactions and Select Salient Ones

Require: Deep neural network v, input sample $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$, set of indices $N = \{1, 2, \ldots, n\}$, small noise threshold ζ, significance threshold τ, convergence threshold ϵ.
Ensure: AND interactions I_T^{and}, OR interactions I_T^{or}, and interaction sets Ω^{and} and Ω^{or}.
Initialize learnable parameters $\{\gamma_L\}$ and $\{\delta_L\}$ for all $L \subseteq N$.
Compute baseline output $v(\boldsymbol{x}_\emptyset)$, where $\boldsymbol{x}_\emptyset$ is the masked sample with all variables removed.
Initialize previous loss $\mathcal{L}_{\text{prev}} \leftarrow \infty$.
repeat
  for each subset $L \subseteq N$ **do**
    Compute masked sample $\boldsymbol{x}_L$ by removing variables not in L.
    Compute network output $v(\boldsymbol{x}_L)$.
    Compute noise term δ_L constrained in $[-\zeta, \zeta]$, where $\zeta = 0.01 \cdot |v(\boldsymbol{x}) - v(\boldsymbol{x}_\emptyset)|$.
    Decompose $v(\boldsymbol{x}_L)$ into AND and OR components:
    $u_L^{\text{and}} \leftarrow 0.5 \cdot (v(\boldsymbol{x}_L) - \delta_L) + \gamma_L$
    $u_L^{\text{or}} \leftarrow 0.5 \cdot (v(\boldsymbol{x}_L) - \delta_L) - \gamma_L$
  end for
  for each subset $T \subseteq N$ **do**
    Compute AND interaction I_T^{and}:

$$I_T^{\text{and}} \leftarrow \sum_{L \subseteq T} (-1)^{|T|-|L|} u_L^{\text{and}}$$

    Compute OR interaction I_T^{or}:

$$I_T^{\text{or}} \leftarrow - \sum_{L \subseteq T} (-1)^{|T|-|L|} u_{N \backslash L}^{\text{or}}$$

  end for
  Compute current loss $\mathcal{L} \leftarrow \sum_{T \subseteq N} \left(|I_T^{\text{and}}| + |I_T^{\text{or}}| \right)$.
  Optimize parameters $\{\gamma_L\}$ and $\{\delta_L\}$ to minimize $\mathcal{L}$.
  Check for convergence: $|\mathcal{L} - \mathcal{L}_{\text{prev}}| < \epsilon$.
  Update previous loss: $\mathcal{L}_{\text{prev}} \leftarrow \mathcal{L}$.
until convergence
Select AND interactions:

$$\Omega^{\text{and}} \leftarrow \{T \subseteq N : |I_T^{\text{and}}| > \tau\}$$

Select OR interactions:

$$\Omega^{\text{or}} \leftarrow \{T \subseteq N : |I_T^{\text{or}}| > \tau\}$$

return I_T^{and}, I_T^{or}, Ω^{and}, and Ω^{or}.

G DETAILS TO EXTRACT THE SPARSEST AND-OR INTERACTIONS

A method is proposed (Li & Zhang, 2023a; Chen et al., 2024) to simultaneously extract AND interactions I_T^{and} and OR interactions I_T^{or} from the network output. Given a masked sample $\boldsymbol{x}_L$, Li & Zhang (2023a) proposed to learn a decomposition $v(\boldsymbol{x}_L) = u_L^{\text{and}} + u_L^{\text{or}}$ towards the sparsest interactions. The component u_L^{and} was explained by AND interactions, and the component u_L^{or} was explained by OR interactions. Specifically, they decomposed $v(\boldsymbol{x}_L)$ into $u_L^{\text{and}} = 0.5 \cdot v(\boldsymbol{x}_L) + \gamma_L$ and $u_L^{\text{or}} = 0.5 \cdot v(\boldsymbol{x}_L) - \gamma_L$, where $\{\gamma_L : L \subseteq N\}$ is a set of learnable variables that determine the decomposition. In this way, the AND interactions and OR interactions can be computed, *i.e.*, $I_T^{\text{and}} = \sum_{L \subseteq T} (-1)^{|T|-|L|} u_L^{\text{and}}$, and $I_T^{\text{or}} = -\sum_{L \subseteq T} (-1)^{|T|-|L|} u_{N \setminus L}^{\text{or}}$.

The parameters $\{\gamma_L\}$ were learned by minimizing the following LASSO-like loss to obtain sparse interactions:

$$\min_{\{\gamma_L\}} \sum_{T \subseteq N} |I_T^{\text{and}}| + |I_T^{\text{or}}| \tag{23}$$

Removing small noises. A small noise δ in the network output may significantly affect the extracted interactions, especially for high-order interactions. Thus, (Li & Zhang, 2023a) proposed to learn to remove a small noise term δ_T from the computation of AND-OR interactions. Specifically, the decomposition was rewritten as $u_L^{\text{and}} = 0.5(v(\boldsymbol{x}_L) - \delta_L) + \gamma_L$ and $u_L^{\text{or}} = 0.5(v(\boldsymbol{x}_L) - \delta_L) - \gamma_L$. Thus, the parameters $\{\delta_L\}$ and $\{\gamma_L\}$ are simultaneously learned by minimizing the loss function in Eq. (23). The values of $\{\delta_L\}$ were constrained in $[-\zeta, \zeta]$ where $\zeta = 0.01 \cdot |v(\boldsymbol{x}) - v(\boldsymbol{x}_\emptyset)|$. Therefore, we can also set $\epsilon = \zeta$ to ensure the fidelity of using the logical model function $\phi(\cdot)$ to mimic the DNN function $v(\cdot)$.

Algorithm of extracting AND-OR interactions. The technical details of computing I_T^{and} and I_T^{or} are provided in the following pseudocode in Algorithm 1.

H COMPUTATIONAL COST AND PRACTICAL SCALABILITY

Similar to the computation of Shapley values (Shapley et al., 1953), extracting interaction effects can be computationally expensive in the worst case, since it requires evaluating model outputs under different masked input states. Nevertheless, this theoretical complexity can often be substantially reduced in practice. Prior studies have shown that DNNs usually encode only a sparse set of salient interactions, which makes efficient approximation possible. For example, the Sparse Möbius Transform reduces the computational complexity to $O(nK \log n)$ when only K salient interactions are present (Kang et al., 2024). Spectral Explainer further improves efficiency by identifying important interactions in LLMs with long input sequences (Kang et al., 2025). More recently, ProxySPEX enhances scalability by using lightweight proxy models to approximate the local decision behavior of the original model (Butler et al., 2025). Together with practical engineering techniques (Li & Zhang, 2023a), these methods provide effective tools for scaling interaction-based explanations.

In addition, prior work (Ren et al., 2024b; Li & Zhang, 2023b) has proposed various techniques to reduce the cost of extracting interactions. Moreover, analyses of DNN generalization typically require computing only interaction distributions rather than exhaustively enumerating all interactions. It means that selectively calculating interactions on a few salient input variables is usually enough to obtain a stable interaction distribution for generalization analysis. Consequently, the computation can be handled efficiently in typical experimental settings.

Therefore, although exact interaction extraction remains computationally expensive in theory, existing approximation algorithms, flexible granularity choices, and practical engineering optimizations make interaction-based analysis feasible in typical experimental settings. Indeed, interaction-based explanations have been applied to a wide range of DNNs, including LLMs, quantitative trading models, pedestrian detection models in autonomous driving, and industrial model-evaluation scenarios.