BATCH NORMALIZATION IS BLIND TO THE FIRST AND SECOND DERIVATIVES OF THE LOSS *w.r.t.* FEATURES

Anonymous authors
Paper under double-blind review

ABSTRACT

We prove that when we do the Taylor series expansion of the loss function, the BN operation will block the influence of the first-order term and most influence of the second-order term of the loss. This is a potential defect of the BN operation. We also find that such a problem is caused by the standardization phase of the BN operation. We believe that the proof of the blindness of a deep model is of significant value to avoiding systemic collapses of a deep model, although such a blindness does not always makes significant damages in all applications. Experiments show that the BN operation significantly affects feature representations in specific tasks[1].

1 INTRODUCTION

Batch normalization (BN) (Ioffe & Szegedy, 2015) plays a crucial role in deep learning and has achieved great success. However, in recent years, people gradually found some shortcomings of the BN operation in some specific applications, including the incompatibility with the dropout operation (Li et al., 2019), hurting the classification accuracy in adversarial training (Xie et al., 2020; Galloway et al., 2019), decreasing the quality of images generated by generative models (Salimans et al., 2016), and causing gradient explosion when a stacked network was deep enough (Yang et al., 2019).

In this paper, we aim to prove a potential defect of the BN operation, *i.e.*, the BN operation is not capable of faithfully reflecting all signals of the loss function. Specifically, when we do the Taylor series expansion of the loss function *w.r.t.* the output of the BN operation, **we prove that the BN operation will block the back-propagation of the first and second derivatives of the loss function *w.r.t.* features in the Taylor series expansion.** *I.e.*, the BN operation makes network layers before the BN layer blind to all first derivatives and some second derivatives of the loss function *w.r.t.* features, which hurts the trustworthiness of the BN operation.

More crucially, the proof of the BN's clear trend of learning specific loss terms and ignoring other loss terms is of great value to both theory and practice. The alarm of such a trend may guide the design of deep neural networks (DNNs), although such a trend is serious in some applications, but is not so serious in other applications.

Given the i-th input sample, let us do the second-order Taylor series expansion of the loss, $\text{Loss}(\mathbf{y}^{(i)})$[2], where $\mathbf{y}^{(i)}$ denotes the output feature of the standardization phase (subtracting the mean and dividing the standard deviation) of the BN operation. $\text{Loss}(\mathbf{y}^{(i)}) = \text{Loss}(\tilde{\mathbf{y}}) + (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{g} + \frac{1}{2!}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) + R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})$, where $\mathbf{g} \in \mathbb{R}^D$ and $\mathbf{H} \in \mathbb{R}^{D\times D}$ denote the gradient and the Hessian matrix of the loss at the point $\tilde{\mathbf{y}} \in \mathbb{R}^D$, respectively. $\tilde{\mathbf{y}}$ is an arbitrary vector close to $\mathbf{y}^{(i)}$.

• Let us first focus on **the simplest case** that all samples in the mini-batch share the same analytic formula of the loss function, *e.g., the loss for invertible generative models* (Dinh et al., 2014; 2016; Kingma & Dhariwal, 2018), In this case, all samples in the mini-batch share the same $\mathbf{g}$ and $\mathbf{H}$.

• Then, let us focus on **more general cases**, which may have the following two issues. (1) Loss functions of different samples in a mini-batch have different analytic formulas, owing to the diversity of labels, *i.e.*, $\text{Loss}(\mathbf{y}^{(i)}, \textit{label}^{(i)})$. (2) The gradient $\mathbf{g}$ and the Hessian matrix $\mathbf{H}$ of the loss are unstable,

[1] We will release all the codes and datasets when this paper is accepted.

[2] In this paper, $\text{Loss}(\mathbf{y})$ is a simplification of $\text{Loss}(h(\mathbf{y}))$, where $h(\cdot) : \mathbb{R}^D \to \mathbb{R}$ denotes the function of network layers between the BN operation and the loss function. We omit $h(\cdot)$ to simplify the description.

because the change of gating states in gating layers (*e.g.*, the ReLU layer) is discontinuous and unpredictable. In these cases, we compute the average gradient $\widehat{\mathbf{g}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{g}^{(i)}$ over all samples in the mini-batch. Then, we compute an equivalent matrix $\widehat{\mathbf{H}}$ to approximately represent losses of different samples in a batch into the same Taylor-like formulation as $\text{Loss}(\mathbf{y}^{(i)}) = \text{Loss}(\tilde{\mathbf{y}}) + (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^{\top}\widehat{\mathbf{g}} + \frac{1}{2!}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^{\top}\widehat{\mathbf{H}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) + \Delta^{(i)}$, where $\widehat{\mathbf{H}}$ is computed to minimize the error term $\Delta^{(i)}$. In this way, $\Delta^{(i)}$ sums up all errors caused by the average gradient $\widehat{\mathbf{g}}$ and the equivalent matrix $\widehat{\mathbf{H}}$, and obtains a considerable value, which is not limited to terms of greater-than-two orders in the Taylor expansion. In fact, we do not require an ignorable $\Delta^{(i)}$ value.

Therefore, for both the simplest case and the more general case, we have discovered and proven the following three conclusions.
1. *Theorem 1 shows that the gradient $\boldsymbol{g}$ (or $\widehat{\boldsymbol{g}}$) at the point $\tilde{\boldsymbol{y}}$ cannot affect the learning of parameters before the BN operation.*
2. *Theorem 2 shows that diagonal elements in the Hessian matrix $\boldsymbol{H}$ (or $\widehat{\boldsymbol{H}}$) at the point $\tilde{\boldsymbol{y}}$ cannot affect network parameters before the BN operation.*
3. *Theorem 3 shows that for off-diagonal elements in $\boldsymbol{H}$ (or $\widehat{\boldsymbol{H}}$) at the point $\tilde{\boldsymbol{y}}$, their impacts on the learning of network parameters before the BN operation are significantly reduced.*

We attribute the above blindness problem to the standardization phase in the BN operation. Although the affine transformation phase in the BN operation allows the DNN to learn from the gradient $\mathbf{g}$ to mitigate the blindness problem, to some extent, experiments have shown that such a blindness problem significantly affects the learning of specific neural networks. Furthermore, as an application, we find that the blindness problem can be simply fixed, if we replace the BN operation with the layer normalization operation (Zaremba et al., 2014).

2 Related work

Normalization methods were usually designed to normalize features or parameters of DNNs, so as to accelerate the training process or improve the generalization of a DNN. Typical normalization methods include the BN operation (Ioffe & Szegedy, 2015), the layer normalization (LN) (Zaremba et al., 2014), the group normalization (Wu & He, 2018), the spectral normalization (Miyato et al., 2018), and the weight normalization (WN) (Salimans & Kingma, 2016).

Some studies discussed positive effects of normalization methods on representation learning. Santurkar et al. (2018) showed that the BN operation smoothed the optimization landscape. Luo et al. (2018) proved that in specific applications, the BN operation was equivalent to the WN operation with a specific regularization term, which improved the generalization of a DNN. Morcos et al. (2018) found that the BN operation usually discouraged the reliance on very few feature dimensions in experiments. Lin et al. (2021) proved that the spectral normalization controled the variance of network parameters in a way similar with the LeCun initialization (LeCun et al., 2012).

Other studies discussed negative effects of normalization methods on representation learning. The BN operation in the GAN usually decreased the visual quality of the generated images in experiments (Salimans et al., 2016). The BN operation was not compatible with both the dropout operation (Li et al., 2019) and weight decay (Van Laarhoven, 2017; Li et al., 2020). Xie et al. (2020) and Galloway et al. (2019) discovered that in adversarial training, the BN operation usually reduced classification accuracy on both clean images and adversarial examples. Yang et al. (2019) proved that a large number of BN layers in a stacked network would probably cause gradient explosion, although the BN operation in a shallow network could still smooth the loss landscape (Santurkar et al., 2018). Xu et al. (2019) showed that the layer normalization increased the risk of over-fitting by ablation experiments. Ioffe (2017) pointed out that when samples in a mini-batch were not independent, the BN's effectiveness diminished. To this end, our theory proves such a demerit of the BN operation from a new perspective, *i.e.*, the BN operation will block the back-propagation of the first and second derivatives of the loss function.

Some studies discussed properties of network parameters when the DNN had normalization layers. Ba et al. (2016) found that when the BN operation or the WN operation was employed, backward gradients through a hidden layer was scale-invariant *w.r.t.* network parameters. Wan et al. (2020) found that gradients of the loss *w.r.t.* network parameters before the BN operation were orthogonal on these parameters when these parameters are scale-invariant.

Unlike previous research, this study focuses on a different issue, *i.e.*, exploring the BN's effects on pushing the DNN to learn specific types of features. In fact, explaining feature representations of a DNN is an emerging direction in recent years. The information-bottleneck theory (Tishby & Zaslavsky, 2015; Shwartz-Ziv & Tishby, 2017; Wolchover & Reading, 2017; Amjad & Geiger, 2019) indicated that a DNN tended to expose task-relevant features and discarded task-irrelevant features, so as to learn discriminative features for classification. Deng *et al.* (Deng et al., 2022) proved that a DNN usually encoded simple interactions between very few input variables and complex interactions between almost all input variables, but it was difficult to encode interactions between an intermediate number of input variables. Unlike previous analysis of feature representations, we prove that the BN operation will block the influence of the first-order loss term and a considerable ratio of the influence of the second-order loss term.

3 BLINDNESS OF THE BN OPERATION

In this section, we aim to prove that the BN operation will block the back-propagation of the first and the second derivatives of the loss function. Given n samples in a mini-batch, let $\mathbf{X} = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}] \in \mathbb{R}^{D\times n}$ denote features of these samples in an intermediate layer before a BN operation, where the i-th column $\mathbf{x}^{(i)} \in \mathbb{R}^D$ corresponds to the i-th sample. Then, the BN operation $\mathbf{Z} = \mathit{BN}(\mathbf{X})$ can be written as the following two phases.

$$\mathbf{Z} = \mathit{diag}(\boldsymbol{\gamma})\mathbf{Y} + \boldsymbol{\beta}\mathbf{1}_n^\top \qquad \text{(affine transformation phase)} \tag{1}$$

$$\mathbf{Y} = \mathit{diag}(\boldsymbol{\sigma}\circ\boldsymbol{\sigma} + \varepsilon\mathbf{1}_D)^{-\frac{1}{2}}(\mathbf{X} - \boldsymbol{\mu}\mathbf{1}_n^\top) \qquad \text{(standardization phase)} \tag{2}$$

where $\mathbf{Z} = [\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \ldots, \mathbf{z}^{(n)}] \in \mathbb{R}^{D\times n}$ denotes the output features of the BN operation; $\mathbf{Y} = [\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(n)}] \in \mathbb{R}^{D\times n}$ denotes the standardized features; $\boldsymbol{\gamma}, \boldsymbol{\beta} \in \mathbb{R}^D$ are used to scale and shift the standardized features; $\boldsymbol{\mu} = \frac{1}{n}\mathbf{X}\mathbf{1}_n \in \mathbb{R}^D$; $\boldsymbol{\sigma} = [\sqrt{\boldsymbol{\Sigma}_{1,1}}, \ldots, \sqrt{\boldsymbol{\Sigma}_{D,D}}]^\top \in \mathbb{R}^D$ represents a vector of the standard deviations corresponding to diagonal elements in the covariance matrix $\boldsymbol{\Sigma} = \frac{1}{n}(\mathbf{X} - \boldsymbol{\mu}\mathbf{1}_n^\top)(\mathbf{X} - \boldsymbol{\mu}\mathbf{1}_n^\top)^\top \in \mathbb{R}^{D\times D}$; $\mathbf{1}_n \in \mathbb{R}^n$ is an all-ones vector; $\circ$ denotes the element-wise product; ε is a tiny positive constant to avoid dividing zero. We ignore the ε term to simplify further proofs. $\mathit{diag}(\cdot)$ transforms a vector to a diagonal matrix. In this way, the training loss on the i-th sample can be represented as a function of the standardized feature $\mathbf{y}^{(i)}$. We use the Taylor series expansion at a fixed point $\tilde{\mathbf{y}} \in \mathbb{R}^D$ (which is an arbitrary vector close to $\mathbf{y}^{(i)}$) to decompose the loss function *w.r.t.* $\mathbf{y}^{(i)}$ into terms of multiple orders, as follows.

$$\text{Loss}(\mathbf{y}^{(i)}; \tilde{\mathbf{y}}) = \text{Loss}(\tilde{\mathbf{y}}) + (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top\mathbf{g} + \frac{1}{2!}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top\mathbf{H}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) + R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \tag{3}$$

where $\mathbf{g} \in \mathbb{R}^D$ and $\mathbf{H} \in \mathbb{R}^{D\times D}$ denote the gradient and the Hessian matrix of $\text{Loss}(\mathbf{y}^{(i)})$, respectively, at the fixed point $\tilde{\mathbf{y}}$; $R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})$ denotes the effect of terms of greater-than-two orders.

In this paper, we consider the following two cases to discuss the effects of the BN operation on the first and second derivatives of the loss function over all samples in a mini-batch.

• **To simplify the introduction, let us first take the following simplest case as an example to introduce our theory. Then, we will use all our theorems to explain more general cases.** In the simplest case, all samples in a mini-batch share the same analytic formula of the loss function. In fact, many applications belong to this case. (1) **Example 1**: in the training of some invertible generative models (Dinh et al., 2014; 2016; Kingma & Dhariwal, 2018), all training samples share the same analytic formula of the loss function, $\text{Loss}(\mathbf{y}^{(i)}) = -\log p(\mathit{input}^{(i)})$, which measures the log-likelihood of sample $\mathit{input}^{(i)}$ estimated by the model. (2) **Example 2**: in distributed learning (Dean et al., 2012), if samples of different categories are stored in different data centers, then samples in the same data center may have the same label. In this specific application, cross-entropy losses $\mathit{crossEntropy}(\mathbf{y}^{(i)}, \mathit{label}^*)$ of samples in the same data center can be all represented in the same analytic formula. (3) **Example 3**: a type of adversarial training (Zheng et al., 2020) requires people to generate multiple adversarial examples in different steps of the multi-step attack, given the same input sample. Thus, just like in Example 2, if we put this set of adversarial samples in the same mini-batch to train the DNN, then all adversarial examples in the mini-batch have the same label, thereby sharing the same analytic formula of the loss function.

For any arbitrary loss function (including the cross-entropy loss, the MSE loss, the logistic loss, and etc.), the overall loss of all samples in a mini-batch $\text{Loss}^{\text{batch}} = \sum_{i=1}^n \text{Loss}(\mathbf{y}^{(i)}; \tilde{\mathbf{y}})$ can be re-written

as the sum of four compositional terms, as follows.

$$\text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H}) \xrightarrow{\textbf{decomposed into}} \text{Loss}^{\text{constant}} + \text{Loss}^{\text{grad}}(\mathbf{g}) + \text{Loss}^{\text{Hessian}}(\mathbf{H}) + \sum\nolimits_i R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \tag{4}$$

where $\text{Loss}^{\text{constant}} = n\text{Loss}(\tilde{\mathbf{y}})$ is a constant *w.r.t.* input features $\mathbf{X}$ of the BN operation; $\text{Loss}^{\text{grad}}(\mathbf{g}) = \sum_{i=1}^{n}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top\mathbf{g}$ and $\text{Loss}^{\text{Hessian}}(\mathbf{H}) = \sum_{i=1}^{n}\frac{1}{2!}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top\mathbf{H}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})$ denote the first-order and second-order terms of the loss function in the Taylor series expansion, respectively.

Theorem 1. (proven in Appendix D.1). *If the loss function is computed based on the output of a batch centering operation $\mathbf{X} - \boldsymbol{\mu}\mathbf{1}_n^\top$ on input features $\mathbf{X}$, then $\text{Loss}^{\text{grad}}(\mathbf{g})$ and $\text{Loss}^{\text{constant}}$ have no gradients on $\mathbf{X}$, i.e., $\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{X}} = \mathbf{0}$ and $\frac{\partial \text{Loss}^{\text{constant}}}{\partial \mathbf{X}} = \mathbf{0}$.*

Lemma 1. (proven in Appendix C.1). *Let us set $\varepsilon = 0$. The tiny positive constant ε is only used to avoid dividing zero, so we can ignore ε in all following paragraphs to simplify the proof. Then, $\text{Loss}^{\text{Hessian}}(\mathbf{H})$ can be decomposed into two terms, i.e.,*

$$\text{Loss}^{\text{Hessian}}(\mathbf{H}) = \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) + \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}),$$

where $\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) = \sum_{i=1}^{n}\frac{1}{2!}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top\mathbf{H}^{\text{diag}}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})$, and $\text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) = \sum_{i=1}^{n}\frac{1}{2!}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top\mathbf{H}^{\text{off}}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})$; $\mathbf{H}^{\text{diag}}$ and $\mathbf{H}^{\text{off}} = \mathbf{H} - \mathbf{H}^{\text{diag}}$ denote the matrix only containing diagonal elements and the matrix only containing off-diagonal elements in $\mathbf{H}$, respectively.

Theorem 2. (proven in Appendix D.2). *If the loss function is computed based on the output of the standardization phase in Equation (2) on input features $\mathbf{X}$, then $\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})$ has no gradients on $\mathbf{X}$, i.e., $\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{X}} = \mathbf{0}$.*

Lemma 2. (proven in Appendix C.2). *Let $\mathbf{x}_d = [\mathbf{X}_{d,1}, \mathbf{X}_{d,2}, \ldots, \mathbf{X}_{d,n}]^\top \in \mathbb{R}^n$ denote the d-th feature dimension of all the n samples in a mini-batch. Then, $\text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})$ can be decomposed into the loss term depending on $\mathbf{x}_d$ (i.e., $L_d(\mathbf{H}^{\text{off}}_{d,:}) = \mathbf{H}^{\text{off}}_{d,:}(\mathbf{Y} - \tilde{\mathbf{y}}\mathbf{1}_n^\top)\mathbf{y}_d$) and the loss term independent with $\mathbf{x}_d$, thereby $\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} = \frac{\partial L_d(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$, where $\mathbf{H}^{\text{off}}_{d,:} = [\mathbf{H}^{\text{off}}_{d,1}, \mathbf{H}^{\text{off}}_{d,2}, \ldots, \mathbf{H}^{\text{off}}_{d,D}] \in \mathbb{R}^D$ denotes the d-th row of $\mathbf{H}^{\text{off}}$, $\mathbf{y}_d = [\mathbf{Y}_{d,1}, \mathbf{Y}_{d,2}, \ldots, \mathbf{Y}_{d,n}]^\top \in \mathbb{R}^n$. Furthermore, $L_d(\mathbf{H}^{\text{off}}_{d,:})$ can be decomposed as*

$$L_d(\mathbf{H}^{\text{off}}_{d,:}) = L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:}) + L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:}),$$

where $L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:}) = \mathbf{H}^{\text{off}}_{d,:}\mathbf{Y}^{\text{linear}}\mathbf{y}_d$ and $L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:}) = \mathbf{H}^{\text{off}}_{d,:}(\mathbf{Y}^{\text{non}} - \tilde{\mathbf{y}}\mathbf{1}_n^\top)\mathbf{y}_d$. $\mathbf{Y}^{\text{linear}} = [\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D]^\top\mathbf{o}_d^\top$ and $\mathbf{Y}^{\text{non}} = \mathbf{Y} - \mathbf{Y}^{\text{linear}}$, where $\mathbf{o}_d = \frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}$ denotes the unit vector in the direction of $\mathbf{y}_d$. Then, $L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:}) = \|\mathbf{y}_d\| \cdot \mathbf{H}^{\text{off}}_{d,:}[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D]^\top$. Therefore, $\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} = \frac{\partial L_d(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} = \frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} + \frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$.

Theorem 3. (proven in Appendix D.3). *If the loss function is computed based on the output of the standardization phase in Equation (2) on input features $\mathbf{X}$, then $\forall d$, $L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})$ has no gradients on the d-th feature dimension over all n samples $\mathbf{x}_d \in \mathbb{R}^n$, i.e.,*

$$\forall d,\ \frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} = \mathbf{0},$$

thus $\frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d \partial \mathbf{H}^{\text{off}}_{d,:}} = \frac{\partial^2}{\partial \mathbf{x}_d \partial \mathbf{H}^{\text{off}}_{d,:}}(A[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D]^\top) = \mathbf{0}$, s.t. $A = \|\mathbf{y}_d\| \cdot \mathbf{H}^{\text{off}}_{d,:}$. On the other hand, $\frac{\partial^2 L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d \partial \mathbf{H}^{\text{off}}_{d,:}} = \frac{1}{\boldsymbol{\sigma}_d} \cdot [(\mathbf{y}_1 - \|\mathbf{y}_1\|\cos(\mathbf{y}_1, \mathbf{y}_d)\mathbf{o}_d), \ldots, (\mathbf{y}_D - \|\mathbf{y}_D\|\cos(\mathbf{y}_D, \mathbf{y}_d)\mathbf{o}_d)] \neq \mathbf{0}$.

Lemma 2 shows that the gradient of $L_d(\mathbf{H}^{\text{off}}_{d,:})$ *w.r.t.* $\mathbf{x}_d$ can be decomposed into two terms, *i.e.*, $\forall d, \frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} = \frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} + \frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$, where $\frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$ reflects the interaction utility yielded by feature components in all dimensions that are linear-correlated with $\mathbf{y}_d$, *i.e.*, $[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D]^\top$, subject to $\mathbf{o}_d = \frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}$. $\frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$ reflects the interaction utility between $\mathbf{y}_d$ and feature components that are not linearly-correlated with $\mathbf{y}_d$, *i.e.*, removing linearly-correlated components, $\forall j, \mathbf{y}_j - \|\mathbf{y}_j\|\cos(\mathbf{y}_j, \mathbf{y}_d)\mathbf{o}_d$. Empirically, linearly-correlated feature components usually represent similar concepts, thereby having stronger interaction utilities than non-correlated feature components. We have conducted experiments to verify such an empirical claim in Section 4. According to Theorem 3, the interaction utility between linearly-correlated feature components cannot pass through the BN operation, *i.e.*, $\frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} = \mathbf{0}$. Therefore, for each dimension d, we can consider that a considerable ratio of the influence of $\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d}$ cannot pass through the BN operation.

Corollary 1. (proven in Appendix D.4). *If the loss function is computed based on the output of the standardization phase in Equation (2) on input features* $\mathbf{X}$*, then based on Theorems 1, 2, and 3, we have* $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{g}} = \mathbf{0}$, $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} = \mathbf{0}$*, and* $\forall d$, $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{x}_d \partial \mathbf{H}^{\text{off}}_{d,:}} = \frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d \partial \mathbf{H}^{\text{off}}_{d,:}} + \frac{\partial^2 L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d \partial \mathbf{H}^{\text{off}}_{d,:}}$*, where* $\frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d \partial \mathbf{H}^{\text{off}}_{d,:}} = \mathbf{0}$ *in the training phase of a neural network. In contrast, in the testing phase,* $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{g}} \neq \mathbf{0}$, $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} \neq \mathbf{0}$*, and* $\frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d \partial \mathbf{H}^{\text{off}}_{d,:}} \neq \mathbf{0}$.

Findings: $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{g}^\top} = \mathbf{0}$, $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} = \mathbf{0}$, and $\forall d$, $\frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d \partial \mathbf{H}^{\text{off}}_{d,:}} = \mathbf{0}$ in Corollary 1 show that the BN operation will block the back-propagation of the following three types of influence, *i.e.*, (1) the influence of the first derivatives in $\mathbf{g}$, (2) the influence of the diagonal elements in the Hessian matrix $\mathbf{H}$, and (3) a considerable ratio of the influence of off-diagonal elements in the Hessian matrix $\mathbf{H}$. More crucially, the BN's blocking of the above influence exists in the training phase, but it does not exist in the testing phase, which may bring in potential risks of neural networks with BN operations.

The reason for blindness. According to Equations (1) and (2), the BN operation contains two phases, the affine transformation phase and the standardization phase. We find that derivatives of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ in the standardization phase eliminate the influence of $\text{Loss}^{\text{grad}}(\mathbf{g})$, $\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})$, and $L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})$. Please see Appendix H for the proof.

Furthermore, although $\boldsymbol{\gamma}$ in the affine transformation phase can **alleviate** the blindness to $\text{Loss}^{\text{grad}}(\mathbf{g})$ by encoding the gradient $\mathbf{g}$ of the first-order term of the loss, $\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})$ and $L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})$ still cannot influence parameters in all layers before the BN operation. Beyond this, the distinctive contribution of this study is to warn people for the potential defect of the BN in learning specific loss terms.

• In **more general cases**, which may have two more issues. First, different samples in a mini-batch have different analytic formulas of loss functions. For example, losses of different samples are computed with different ground-truth labels. Second, the gradient $\mathbf{g}$ and the Hessian matrix $\mathbf{H}$ are unstable, because the change of gating states over different samples in gating layers (*e.g.*, the ReLU layer) is discontinuous and unpredictable.

Theorem 4. (proven in Appendix C.3) *The loss function for each* i*-th sample,* $1 \leq i \leq n$*, can be written as* $\text{Loss}(\mathbf{y}^{(i)}; \tilde{\mathbf{y}}) = \text{Loss}(\tilde{\mathbf{y}}) + \text{Loss}^{\text{grad}}(\widehat{\mathbf{g}}) + \text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}}) + \text{Loss}^{\text{off}}(\widehat{\mathbf{H}}^{\text{off}}) + \Delta^{(i)}$*, where* $\widehat{\mathbf{g}} = \frac{1}{n}\sum_{i=1}^n \mathbf{g}^{(i)}$ *denotes the average gradient,* $\widehat{\mathbf{H}}$ *denotes the equivalent Hessian matrix. Both* $\widehat{\mathbf{g}}$ *and* $\widehat{\mathbf{H}}$ *are shared by all samples in the mini-batch.* $\Delta^{(i)}$ *denotes the distinctive term of the* i*-th sample, which sums up all errors caused by* $\widehat{\boldsymbol{g}}$ *and* $\widehat{\boldsymbol{H}}$*, but is not limited to terms of greater-than-two orders in the Taylor expansion.*

We optimize the equivalent $\widehat{\mathbf{H}}$ by minimizing $\sum_{i=1}^n \|\mathbf{g}^{(i)} - \widehat{\mathbf{g}} - \widehat{\mathbf{H}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})\|_2^2$. Just like the simplest case, we prove that Theorems 1, 2, 3, and Corollary 1 also hold for general cases. *I.e.*, $\text{Loss}^{\text{grad}}(\widehat{\mathbf{g}})$ and $\text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}})$ have no gradients on features before the BN operation, and $\text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{off}})$ does not have strong effects on features before the BN operation. Please see Appendix E for proofs.

• **Do we need a very tiny residual term (*i.e*,** $R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})$ **in Equation (3) and** $\Delta^{(i)}$ **in Theorem 4) in the Taylor series expansion?** Because the most crucial issue in this paper is to examine whether the loss function can faithfully pass its ***"all"*** effects through the BN operation, moderately large values for $\text{Loss}^{\text{grad}}(\mathbf{g})$ and $\text{Loss}^{\text{Hessian}}(\mathbf{H})$ are already enough to illustrate the seriousness of the blindness problem. It is not necessary to let $\text{Loss}^{\text{grad}}(\mathbf{g})$ and $\text{Loss}^{\text{Hessian}}(\mathbf{H})$ represent almost all values of the loss (*i.e.*, letting the residual term $R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})$ or $\Delta^{(i)}$ be ignorable). Even though the first two terms in the Taylor series expansion just take a relatively small portion of all loss effects, people still have the right to know the truth of the BN's behavior. In fact, Table 5 shows that **the** $\text{Loss}^{\text{grad}}(\widehat{\mathbf{g}})$ **and** $\text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}})$ **terms have made 37.8%–79.5% influence *w.r.t.* the loss value in a general task of image classification.** Clarifying detailed potential defects of the BN's behavior has distinctive values to the safe and trustworthy AI. Nevertheless, experiments in Section 4 further show that given an ignorable residual term, the blindness problem may fully dominate the learning process, thereby damaging feature representations of the DNN.

Anyway, as a common trick, we can set $\tilde{\mathbf{y}}$ to the mean value $\mathbb{E}_i[\mathbf{y}^{(i)}]$ to boost the fitness of the Taylor series expansion of losses on most samples, thereby reducing the residual term. In particular, we

Table 1: Verifying the BN's effects on the back-propagation of derivatives of different orders.

Δgrad_0	Δgrad_1	Δgrad_2	Δgrad_3
0 ± 0	1.79e-7 $\pm$ 2.07e-7	2.85e-7 $\pm$ 3.94e-8	0.51 $\pm$ 0.33

Table 2: Verifying the BN's effects on the back-propagation of the first and second derivatives.

DNN	$\Delta\text{grad}^{\text{first}}$	$\Delta\text{grad}^{\text{second,diag}}$	$\Delta\text{grad}^{\text{second,off}}$
VGG-11 with a BN layer after the third top FC	1.37e-8 $\pm$ 2.72e-8	2.72e-8 $\pm$ 5.41e-8	1019.49 $\pm$ 116.45
VGG-11 with a BN layer after conv5-2	6.97e-16 $\pm$ 1.18e-16	1.23e-09 $\pm$ 2.47e-09	323.48 $\pm$ 33.79
VGG-11 with a BN layer after conv5-1	6.88e-16 $\pm$ 1.71e-16	1.35e-09 $\pm$ 2.93e-09	387.46 $\pm$ 300.19

have proven in Appendix F that the third and higher-order derivatives in the residual term of the sigmoid function have small strengths when the classification is confident.

4 EXPERIMENTS

In this section, we conducted experiments to verify above theorems, and analyze the impact of the blindness problem on feature representations of DNNs.

• **Experimental verification of Theorems 1 and 2**. We conducted two experiments on a synthetic dataset and the CIFAR-10 dataset, respectively, to verify that $\text{Loss}^{\text{grad}}(\mathbf{g})$ and $\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})$ had no gradients on features before the BN operation.

Verifying the blindness of the BN operation on the synthesized loss functions. In this experiment, we directly designed the loss function as a polynomial with derivatives of different orders, in order to evaluate the effects of the BN operation on derivatives of different orders. To this end, we synthesized a group of five loss functions with derivatives of different orders, $\forall 0 \leq k \leq 4$, $\text{Loss}_k(\mathbf{y}|\boldsymbol{\lambda}) = \sum_{i=1}^{n}\sum_{k'=k}^{4}\lambda_{k'}(y^{(i)})^{k'}$, by sampling parameters $\boldsymbol{\lambda} = [\lambda_0, \lambda_1, \ldots, \lambda_4]^\top \sim N(\boldsymbol{\mu} = \mathbf{0}, \Sigma = I_{5\times 5})$ from a Gaussian distribution. Thus, we constructed a dataset[1] containing 1000 groups of loss functions by sampling $\boldsymbol{\lambda}$ for 1000 times. Then, we put a BN operation on the top of a 5-layer MLP (each layer containing $M = 100$ neurons) for each group of five losses *w.r.t.* a specific $\boldsymbol{\lambda}$, we put each loss, Loss_k, upon a BN operation, to train a neural network. Thus, we used five losses $\{\text{Loss}_k\}$ to train five MLPs. The input of the MLP was high-dimensional noises sampled from a Gaussian distribution $N(\boldsymbol{\mu} = \mathbf{0}, \Sigma = I_{M\times M})$, and the output was a scalar after a linear operation.

We measured $\Delta\text{grad}_q = \mathbb{E}_{\boldsymbol{\lambda}}\left[\|\frac{\partial \text{Loss}_q(\mathbf{y}|\boldsymbol{\lambda})}{\partial \mathbf{x}} - \frac{\partial \text{Loss}_{q+1}(\mathbf{y}|\boldsymbol{\lambda})}{\partial \mathbf{x}}\|/\|\frac{\partial \text{Loss}_q(\mathbf{y}|\boldsymbol{\lambda})}{\partial \mathbf{x}}\|\right], q = 0, 1, 2, 3$, to examine whether the q-th order term of the loss could pass its influence through the BN operation. Table 1 shows that $\Delta\text{grad}_0 = 0, \Delta\text{grad}_1 \approx 0$, and $\Delta\text{grad}_2 \approx 0$, which proved that the zeroth-order, the first-order, and the second-order terms[3] of the loss could not pass their influence through the BN operation. The small deviation was caused by the accumulation of tiny systematic computational errors in a DNN. Besides, $\Delta\text{grad}_3 = 0.51 \pm 0.33$ indicated in the Taylor series expansion that the third-order term successfully passed its influence through the BN operation.

Verifying the blindness of the BN operation in image classification. We constructed three VGG-11 network by adding a BN layer (in Equations (1) and (2)) after the third top FC, conv5-2, and conv5-1, respectively. Each network was trained on the CIFAR-10 dataset with a classification loss ($\text{Loss}^* = \sum_{i=1}^{n}\text{Loss}^{\text{cls}}(\mathbf{y}^{(i)})$). Then, we tested the blindness of the BN operation to the gradient $\mathbf{g}$ and diagonal elements in the Hessian matrix $\mathbf{H}$ of the network. Specifically, in order to evaluate the exact effect of each term in the Taylor series expansion of the loss, we manually added noisy first derivatives and noisy second derivatives to construct three additional losses (Loss_2, Loss_3, and Loss_4). *I.e.*, $\text{Loss}_2 = \sum_{i=1}^{n}(\text{Loss}^{\text{cls}}(\mathbf{y}^{(i)}) + \boldsymbol{\epsilon}^\top\mathbf{y}^{(i)})$, $\text{Loss}_3 = \sum_{i=1}^{n}(\text{Loss}^{\text{cls}}(\mathbf{y}^{(i)}) + (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathit{diag}(\boldsymbol{\epsilon})(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}))$, and $\text{Loss}_4 = \sum_{i=1}^{n}(\text{Loss}^{\text{cls}}(\mathbf{y}^{(i)}) + (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{E}^{\text{off}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}))$. We set $\tilde{\mathbf{y}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{y}^{(i)}$ in this experiment. Each element in $\boldsymbol{\epsilon} \in \mathbb{R}^D$ and $\mathbf{E}^{\text{off}} \in \mathbb{R}^{D\times D}$ was sampled from a Gaussian distribution $N(\mu = 0, \sigma^2 = 0.1^2)$. Diagonal elements in $\mathbf{E}^{\text{off}}$ were set to 0.

In this way, if $\frac{\partial \text{Loss}^*}{\partial \mathbf{X}} = \frac{\partial \text{Loss}_2}{\partial \mathbf{X}} = \frac{\partial \text{Loss}_3}{\partial \mathbf{X}}$, then it proved that the first derivatives in $\mathbf{g}$ and diagonal elements in $\mathbf{H}$ could not pass their influence through the BN operation. Therefore, we used

[3] In this experiment, the Hessian matrix reduced to the scalar second derivative.

Table 3: Gradients of $L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})$ *w.r.t.* the input feature $\mathbf{x}_d$ of the BN operation.

	AlexNet-1	AlexNet-2	AlexNet-3	LeNet-1	LeNet-2	LeNet-3
$\|\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d}\|$	7.12e-12 ± 8.64e-12	2.85e-11 ± 5.11e-11	3.86e-11 ± 7.45e-11	1.01e-10 ± 1.98e-9	5.17e-12 ± 8.75e-12	1.45e-11 ± 2.28e-11

Table 4: Comparing the relative significance between $\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})/\partial \mathbf{y}_d$ and $\partial L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})/\partial \mathbf{y}_d$.

	AlexNet-1	AlexNet-2	AlexNet-3	LeNet-1	LeNet-2	LeNet-3
$\|\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}\|/\|\frac{\partial L_d(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}\|$	**0.84 ± 0.19**	**0.73 ± 0.25**	**0.85 ± 0.22**	**0.67 ± 0.29**	**0.68 ± 0.29**	**0.77 ± 0.27**
$\|\frac{\partial L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}\|/\|\frac{\partial L_d(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}\|$	0.47 ± 0.22	0.56 ± 0.29	0.40 ± 0.28	0.63 ± 0.27	0.61 ± 0.29	0.51 ± 0.27

metrics $\Delta\text{grad}^{\text{first}} = \|\frac{\partial \text{Loss}^*}{\partial \mathbf{X}} - \frac{\partial \text{Loss}_2}{\partial \mathbf{X}}\|_F/\|\frac{\partial \text{Loss}^*}{\partial \mathbf{X}}\|_F$, $\Delta\text{grad}^{\text{second,diag}} = \|\frac{\partial \text{Loss}^*}{\partial \mathbf{X}} - \frac{\partial \text{Loss}_3}{\partial \mathbf{X}}\|_F/\|\frac{\partial \text{Loss}^*}{\partial \mathbf{X}}\|_F$, and $\Delta\text{grad}^{\text{second,off}} = \|\frac{\partial \text{Loss}^*}{\partial \mathbf{X}} - \frac{\partial \text{Loss}_4}{\partial \mathbf{X}}\|_F/\|\frac{\partial \text{Loss}^*}{\partial \mathbf{X}}\|_F$ to measure the influence of the first derivatives in $\mathbf{g}$, diagonal elements in $\mathbf{H}$, and off-diagonal elements in $\mathbf{H}$ on the gradient $\frac{\partial \text{Loss}^*}{\partial \mathbf{X}}$. According to Table 2, $\Delta\text{grad}^{\text{first}} \approx 0$ and $\Delta\text{grad}^{\text{second,diag}} \approx 0$ proved that elements in $\mathbf{g}$ and diagonal elements in $\mathbf{H}$ could not pass their influence through the BN operation. The small deviation was caused by the accumulation of tiny systematic computational errors in a DNN. Besides, the large value of $\Delta\text{grad}^{\text{second,off}}$ indicated that off-diagonal elements in $\mathbf{H}$ could pass their influence through the BN operation. In addition, Appendix I.5 shows more results that each element in $\boldsymbol{\epsilon}$ and $\mathbf{E}^{\text{off}}$ was sampled from Gaussian distributions with large μ and large σ^2, which also yielded similar conclusions.

• **Experimental verification of Theorem 3**. In this experiment, we aimed to verify that $L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})$ had no gradients on features before the BN operation. To this end, we directly computed the norm of the gradient $\|\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})/\partial \mathbf{x}_d\|$, where $\mathbf{H}_{d,:}^{\text{off}}$ was computed by following (Cohen et al., 2020). To comprehensively test the BN on different layers, we revised the AlexNet (Krizhevsky et al., 2012) by adding five additional FC layers before the top FC layer, and revised the LeNet (LeCun et al., 1989) by adding seven additional FC layers before the top FC layer. Please see Appendix I.1 about the revised architectures. For each DNN, we added a BN layer before the 1st, 2nd, and 3rd top FC layers, respectively, to construct *AlexNet-1, AlexNet-2, AlexNet-3* and *LeNet-1, LeNet-2, LeNet-3*. Table 3 reports $\|\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})/\partial \mathbf{x}_d\| \approx 0$, which was averaged over different feature dimensions (d) and different batches. The small deviation was caused by the accumulation of tiny systematic computational errors. It proved that $L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})$ had no gradients on features before the BN operation.

• **Examining that interactions between linearly-correlated feature components took a main part of the loss $L_d(\mathbf{H}_{d,:}^{\text{off}})$.** According to Lemma 2, $L_d(\mathbf{H}_{d,:}^{\text{off}}) = L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}}) + L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})$. Note that we have proven that the $L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})$ term has no effects on $\mathbf{x}_d$, so in this experiment, we aimed to further show that the $L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})$ term had considerable gradients *w.r.t.* $\mathbf{y}_d$. If so, it means that the BN blocked significant influence of $\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}$ in back-propagation. To this end, we computed $\|\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}\|/\|\frac{\partial L_d(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}\|$ to measure the relative significance of the compositional influence of $\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}$ to $\frac{\partial L_d(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}$. Similarly, and we computed $\|\frac{\partial L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}\|/\|\frac{\partial L_d(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}\|$ measured the relative significance of $\frac{\partial L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}$ to $\frac{\partial L_d(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}$. We conducted experiments on *AlexNet-1, AlexNet-2, AlexNet-3, LeNet-1, LeNet-2*, and *LeNet-3*. Table 4 reports the relative significance averaged over different feature dimensions and different batches. $\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}$ had considerable impacts on the overall gradient $\frac{\partial L_d(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d}$, which demonstrated that the BN blocked significant influence of $L_d(\mathbf{H}_{d,:}^{\text{off}})$.

• **Examining that the blocked loss terms $\text{Loss}^{\text{grad}}(\widehat{\mathbf{g}})$ and $\text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}})$ were non-ignorable *w.r.t.* the entire loss in *general* cases**. To this end, we measured the proportion of $\text{Loss}^{\text{grad}}(\widehat{\mathbf{g}})$ and $\text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}})$ in the overall loss in the general case. We conducted experiments on AlexNet, VGG-11 and VGG-16, each network was added a BN layer before the top FC layer. These DNNs were trained on the CIFAR-10 dataset for image classification. We used the metric $p^{\text{blocked}} = (\sum_{i=1}^n |(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \widehat{\mathbf{g}}| + \sum_{i=1}^n \frac{1}{2!}|(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \widehat{\mathbf{H}}^{\text{diag}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})|)/\sum_{i=1}^n \text{Loss}(\mathbf{y}^{(i)}; \tilde{\mathbf{y}})$ to measure the proportion of the blocked above two loss terms. According to Table 5, p^{blocked} values for the VGG-16, VGG-11, and AlexNet were 79.5%, 66.2%, and 37.8%, respectively. It meant that the blocked loss signals were as significant as 79.5% of the loss value. This proved that the blocked loss terms $\text{Loss}^{\text{grad}}(\widehat{\mathbf{g}})$ and $\text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}})$ were still made non-ignorable impacts on model training in general cases.

Table 5: Measuring the proportion of the blocked two loss terms $\text{Loss}^{\text{grad}}(\widehat{\mathbf{g}})$ and $\text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}})$.

	VGG-16	VGG-11	AlexNet
p^{blocked}	0.795 ± 0.397	0.662 ± 0.303	0.378 ± 0.200

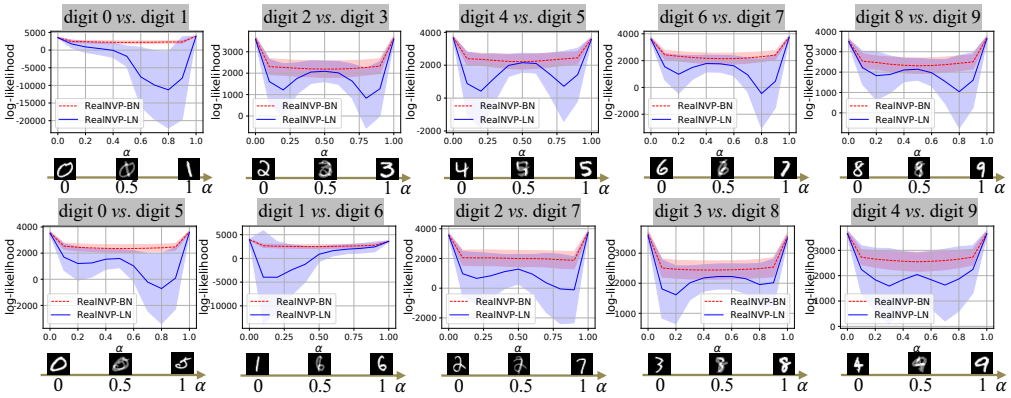

Figure 1: Log-likelihood of real images (at $\alpha = 0$ and $\alpha = 1$) and interpolated images generated by the RealNVP-BN and the RealNVP-LN. The shaded area represents the standard deviation.

• **Analyzing the features learned by neural networks with BN operations.** We conducted experiments to compare DNNs with BN operations and DNNs without BN operations in different applications, in order to analyze the influence of the BN operation on feature representations.

Experiment 1: We measured the BN's effects on feature representations of the invertible generative model RealNVP (Dinh et al., 2016). The vanilla RealNVP had BN operations, thus being termed *RealNVP-BN*. Besides, we constructed another RealNVP by replacing all BN layers with LN layers[4], as a baseline (namely *RealNVP-LN*) for comparison. All RealNVPs were trained on the MNIST dataset (LeCun et al., 1998).

We tested whether a RealNVP model could successfully distinguish real images and fake images. This was the key capacity for a generative model. Specifically, a well-trained RealNVP was supposed to predict high log-likelihood on real images I_1^{real} and I_2^{real}, and yield low log-likelihood on fake images. We generated fake images by linear interpolation, $I_\alpha^{\text{inter}} = \alpha I_1^{\text{real}} + (1-\alpha) I_2^{\text{real}}, \alpha \in (0,1)$. To sharpen the difference caused by the BN operation, we learned different groups of RealNVP-BN and RealNVP-LN, each being trained to generate a specific pair of categories, as Figure 1 shows. Then, for each RealNVP, we computed the average log-likelihood of interpolated images I_α^{inter} *w.r.t.* a specific interpolation rate α, *i.e.*, $\mathbb{E}_{\forall I_1^{\text{real}}, I_2^{\text{real}}}[\log p(I_\alpha^{\text{inter}})]$. Figure 1 shows that RealNVP-LN usually assigned much higher log-likelihood with real images (*i.e.*, images at the points of $\alpha = 0$ and $\alpha = 1$) than interpolated images. In comparison, RealNVP-BN could not significantly distinguish real images and interpolated images. It may be because that the first derivative of the loss usually reflected prototype features for different categories. Thus, the blindness to the first derivative prevented the RealNVP from modeling prototype features of different digits. In addition, Appendix I.3 shows results on more RealNVP models with various revised architectures, which also yielded similar conclusions.

Experiment 2: We measured the BN's effects on feature representations for classification. We compared four groups of DNNs trained on the CIFAR-10 dataset, as follows. The first group of DNNs did not contain any normalization operations, namely *DNN-ori*. The second group of DNNs were obtained by adding BN operations to the DNN-ori, termed *DNN-BN*. The third group of DNNs were obtained by replacing all BN operations in the DNN-BN with LN operations, namely *DNN-LN*. The above three groups of DNNs were trained when each mini-batch only contained samples in a specific category. The fourth group of DNNs had the same architecture as the DNN-ori, but they were trained when all samples in a mini-batch had different labels, termed *DNN-ori-base*. We selected classic network architectures for classification, including VGG-11, VGG-16, ResNet-18, ResNet-34 (He et al., 2016), DenseNet-121, and DenseNet-169 (Huang et al., 2017). For VGG-11 and VGG-16, we constructed and learned all four groups of DNNs. Specifically, we added one single BN (or

[4]Please see Appendix I.2 about how to invert features in RealNVP-LN.

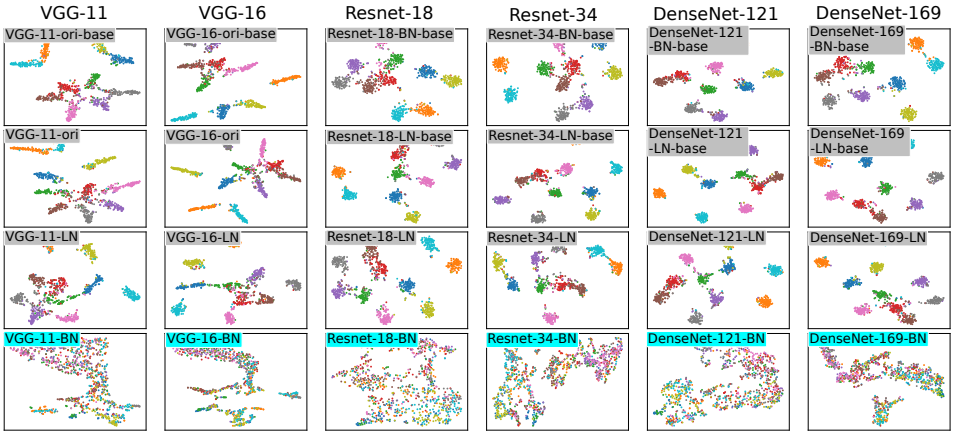

Figure 2: Comparing features of DNNs with BN layers and DNNs without BN layers. Points of different colors indicate samples of different categories. The DNN-BN usually learned much less clustered features than the DNN-LN, when these two DNNs were learned with the same settings.

LN) layer before the second top FC layer of the VGG network. For DNNs already containing BN operations, *i.e.*, ResNet-18/34 and DenseNet-121/169, we directly selected these DNNs as DNN-BN networks, and there were no DNN-ori networks for such DNNs. In addition, for such DNNs, we also learned two baselines, namely *DNN-BN-base* and *DNN-LN-base*, which were trained when all samples in a mini-batch had different labels.

In Figure 2, we used t-SNE (van der Maaten & Hinton, 2008) to visualize the input feature of the top FC layer of each DNN, in order to compare the BN's effects on the feature representation. When losses of all samples in a mini-batch shared the same analytic formula, features of the DNN-BN on different samples were far less clustered than those of the DNN-ori and the DNN-LN. This experiment indicates that the BN operation prevented the DNN from learning discriminative features of samples in each specific category. It was because the first derivatives on the average sample $\tilde{\mathbf{y}}$ in a specific category usually contained information of common discriminative features shared by different samples in this category. In comparison, when losses of all samples in a mini-batch had different analytic formulas, the blindness problem did not significantly damage the feature representation of the DNN. Nevertheless, people had the right to know the BN's potential risk, although sometimes, the remained terms, including both the residual term and effects of non-diagonal elements in $\mathbf{H}$, were already enough for classification. In addition, Appendix I.4 shows results on more network architectures, which also yielded similar conclusions.

5 Conclusions and Discussion

In this paper, we have discovered and theoretically proven the intrinsic blindness problem with the BN operation. Such a problem may bring in an uncommon yet non-ignorable risk in the learning of DNNs. Experiments have verified our theoretical proofs. Besides, experiments have demonstrated that in specific applications, the blindness of the BN operation prevents the DNN from learning discriminative features.

Moreover, we have also proven that it is the standardization phase of the BN operation causes the above effects of the BN operation. However, it is difficult to obtain a simple conclusion that the standardization phase is harmful. In fact, Xu et al. (2019) proved that the standardization phase in the LN operation re-centered gradients of the loss *w.r.t.* features, and reduced the variance of these gradients. Xu *et al.* also found that the standardization phase improved the performance of DNNs in experiment. Besides, Liu et al. (2021) showed that the standardization phase could effectively alleviate the "self-enhancement" phenomenon, and avoided hurting the diversity of features in the DNN. Nevertheless, we have proven that the standardization phase causes the blindness to the first and second derivatives of the loss function in specific applications. To this end, we use the LN operation to replace the BN operation in such applications, which avoids the blindness problem. Appendix A further introduces some limitations of applying our findings to real applications.

ETHICS STATEMENT STATEMENT

As a fundamental research in machine learning, this paper does not introduce any new ethical or societal concerns. The results in this paper do not include misleading claims; their correctness is theoretically verified. Related work is accurately represented. Though in theory any technique can be misused, it is not likely to happen at the current stage.

REPRODUCIBILITY STATEMENT

This research discovered and theoretically explained the potential defects of the BN operation. For our theoretical results, formal statements and the complete proofs of all Lemmas, Theorems and the Corollary in Section 3 are provided in Appendix B, C, D, E, and the disscussion of assumptions are provided in Appendix F, G, H. For our empirical results, we use two public datasets, links to which are included in the main text, and we have discribed additional experimental details in Appendix I, including various model architectures, hyperparameter setting, and how to inverse features in RealNVP-LN, which ensure the reproducibility. We will release all the codes and datasets when this paper is accepted.

REFERENCES

Rana Ali Amjad and Bernhard C Geiger. Learning representations for neural network-based classification using the information bottleneck principle. *IEEE transactions on pattern analysis and machine intelligence*, 42(9):2225–2239, 2019.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Jeremy Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. In *International Conference on Learning Representations*, 2020.

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. *Advances in neural information processing systems*, 25, 2012.

Huiqi Deng, Qihan Ren, Xu Chen, Hao Zhang, Jie Ren, and Quanshi Zhang. Discovering and explaining the representation bottleneck of dnns. In *International Conference on Learning Representations*, 2022.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Angus Galloway, Anna Golubeva, Thomas Tanay, Medhat Moussa, and Graham W Taylor. Batch normalization is a cause of adversarial vulnerability. *arXiv, abs/1905.02161*, 2019.

Gaston H. Gonnet and Ralf Scholl. *Scientific Computation*. Scientific Computation, 2009.

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.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

Sergey Ioffe. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. *Advances in neural information processing systems*, 30, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

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.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

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

Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pp. 9–50. Springer, 2012.

Xiang Li, Shuo Chen, Xiaolin Hu, and Jian Yang. Understanding the disharmony between dropout and batch normalization by variance shift. In *Computer Vision and Pattern Recognition*, 2019.

Xiang Li, Shuo Chen, and Jian Yang. Understanding the disharmony between weight normalization family and weight decay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4715–4722, 2020.

Zinan Lin, Vyas Sekar, and Giulia Fanti. Why spectral normalization stabilizes gans: Analysis and improvements. *Advances in neural information processing systems*, 34, 2021.

Dongrui Liu, Shaobo Wang, Jie Ren, Kangrui Wang, Sheng Yin, and Quanshi Zhang. Trap of feature diversity in the learning of mlps. *arXiv preprint arXiv:2112.00980*, 2021.

Ping Luo, Xinjiang Wang, Wenqi Shao, and Zhanglin Peng. Towards understanding regularization in batch normalization. *arXiv preprint arXiv:1809.00846*, 2018.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Ari S Morcos, David GT Barrett, Neil C Rabinowitz, and Matthew Botvinick. On the importance of single directions for generalization. *arXiv preprint arXiv:1803.06959*, 2018.

Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Advances in neural information processing systems*, 29, 2016.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Neural Information Processing Systems*, 2016.

Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? *Advances in Neural Information Processing Systems*, 31, 2018.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

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

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pp. 1–5. IEEE, 2015.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Twan Van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017.

Ruosi Wan, Zhanxing Zhu, Xiangyu Zhang, and Jian Sun. Spherical motion dynamics of deep neural networks with batch normalization and weight decay. 2020.

Natalie Wolchover and Lucy Reading. New theory cracks open the black box of deep learning. *Quanta Magazine*, 3, 2017.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 819–828, 2020.

Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. Understanding and improving layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

Greg Yang, Jeffrey Pennington, Vinay Rao, Jascha Sohl-Dickstein, and Samuel S. Schoenholz. A mean field theory of batch normalization. In *International Conference on Learning Representations*, 2019.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

Haizhong Zheng, Ziqi Zhang, Juncheng Gu, Honglak Lee, and Atul Prakash. Efficient adversarial training with transferable adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1181–1190, 2020.

A LIMITATIONS OF OUR STUDY

Only in a few cases, the blindness problem of the BN operation brings serious systemic collapses. Since we cannot fully ensure that such special cases will not appear in real applications, the warning of potential defects of the BN's behavior is still of great values to the safe and trustworthy AI. When a DNN suffers from the blindness problem of the BN operation, our theory will give a direct explanation and solution to the failure of the system, instead of letting people blindly enumerate different tricks in an experimental manner to solve the problem.

B PRELIMINARY: BATCH NORMALIZATION

In this section, in order to help readers understand the analysis in the paper, we first revisit the details of Batch Normalization (BN) (Ioffe & Szegedy, 2015).

Notations. Given n samples in a mini-batch, let $\mathbf{X} = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(n)}] = [\mathbf{x}_1^\top; \mathbf{x}_2^\top; \dots; \mathbf{x}_D^\top] \in \mathbb{R}^{D\times n}$ denote features of these samples in an intermediate layer before a BN operation. The i-th column vector corresponds to the i-th sample with D-dimensional feature, which is written as $\mathbf{x}^{(i)} \in \mathbb{R}^D$, $i = 1, \cdots, n$; the d-th row vector corresponds to the d-th feature dimension of all the n samples, the transposition of which is written as $\mathbf{x}_d \in \mathbb{R}^n$, $d = 1, \cdots, D$. For the convenience of readers, we represent $\mathbf{X}$ as following forms for an intuitive understanding of above description.

$$\mathbf{X} = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{x}^{(1)} & \mathbf{x}^{(2)} & \cdots & \mathbf{x}^{(n)} \\ | & | & \cdots & | \end{bmatrix} = \begin{bmatrix} — \ \mathbf{x}_1^\top \ — \\ — \ \mathbf{x}_2^\top \ — \\ \vdots \\ — \ \mathbf{x}_D^\top \ — \end{bmatrix} = \begin{bmatrix} \mathbf{X}_{1,1} & \mathbf{X}_{1,2} & \cdots & \mathbf{X}_{1,n} \\ \mathbf{X}_{2,1} & \mathbf{X}_{2,2} & \cdots & \mathbf{X}_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{X}_{D,1} & \mathbf{X}_{D,2} & \cdots & \mathbf{X}_{D,n} \end{bmatrix}.$$

Then, the BN operation $\mathbf{Z} = \mathit{BN}(\mathbf{X})$ normalizes $\mathbf{X}$ as follows.

$$\mathbf{Z} = \mathit{diag}(\boldsymbol{\gamma})\mathbf{Y} + \boldsymbol{\beta}\mathbf{1}_n^\top \qquad \text{(affine transformation phase)} \tag{1}$$

$$\mathbf{Y} = \mathit{diag}(\boldsymbol{\sigma} \circ \boldsymbol{\sigma} + \varepsilon\mathbf{1}_D)^{-\frac{1}{2}}(\mathbf{X} - \boldsymbol{\mu}\mathbf{1}_n^\top) \qquad \text{(standardization phase)} \tag{2}$$

where $\mathbf{Z} = [\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \dots, \mathbf{z}^{(n)}] \in \mathbb{R}^{D\times n}$ denotes output features of the BN operation; $\mathbf{Y} = [\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \dots, \mathbf{y}^{(n)}] = [\mathbf{y}_1^\top; \mathbf{y}_2^\top; \dots; \mathbf{y}_D^\top] \in \mathbb{R}^{D\times n}$ denotes standardized features; $\boldsymbol{\gamma}, \boldsymbol{\beta} \in \mathbb{R}^D$ are used to scale and shift the standardized features; $\boldsymbol{\mu} = \frac{1}{n}\mathbf{X}\mathbf{1}_n \in \mathbb{R}^D$; $\boldsymbol{\Sigma} = \frac{1}{n}(\mathbf{X} - \boldsymbol{\mu}\mathbf{1}_n^\top)(\mathbf{X} - \boldsymbol{\mu}\mathbf{1}_n^\top)^\top \in \mathbb{R}^{D\times D}$ denotes the covariance matrix of $\mathbf{X}$, and then $\boldsymbol{\sigma} = [\sqrt{\boldsymbol{\Sigma}_{1,1}}, \dots, \sqrt{\boldsymbol{\Sigma}_{D,D}}]^\top \in \mathbb{R}^D$ represents a vector of the standard deviations corresponding to diagonal elements in $\boldsymbol{\Sigma}$; $\mathbf{1}_D \in \mathbb{R}^D$ and $\mathbf{1}_n \in \mathbb{R}^n$ are all-one vectors; $\circ$ denotes the element-wise product; ε is a tiny positive constant to avoid dividing zero; $\mathit{diag}(\cdot)$ transforms a vector to a diagonal matrix. Similarly,

$$\mathbf{Y} = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{y}^{(1)} & \mathbf{y}^{(2)} & \cdots & \mathbf{y}^{(n)} \\ | & | & \cdots & | \end{bmatrix} = \begin{bmatrix} — \ \mathbf{y}_1^\top \ — \\ — \ \mathbf{y}_2^\top \ — \\ \vdots \\ — \ \mathbf{y}_D^\top \ — \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{1,1} & \mathbf{Y}_{1,2} & \cdots & \mathbf{Y}_{1,n} \\ \mathbf{Y}_{2,1} & \mathbf{Y}_{2,2} & \cdots & \mathbf{Y}_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{Y}_{D,1} & \mathbf{Y}_{D,2} & \cdots & \mathbf{Y}_{D,n} \end{bmatrix}.$$

Since the tiny positive constant ε is only used to avoid dividing zero, we ignore this term to simplify further proofs. Therefore,

$$\mathbf{Y} = \mathit{diag}(\boldsymbol{\sigma})^{-1}(\mathbf{X} - \boldsymbol{\mu}\mathbf{1}_n^\top) \qquad \text{(standardization phase)} \tag{3}$$

For the d-th feature dimension of all the n samples, $d = 1, \cdots, D$,

$$\mathbf{y}_d = \frac{\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n}{\boldsymbol{\sigma}_d} \in \mathbb{R}^n. \tag{4}$$

The transposition of $\mathbf{y}_d$, *i.e.*, $\mathbf{y}_d^\top$ corresponds to the d-th row vector of the standardized feature matrix $\mathbf{Y}$.

C PROOFS OF LEMMAS 1, 2, AND THEOREM 4

In this section, we prove Lemmas 1, 2, and Theorem 4 in the main paper, which implement loss decomposition, in order to analyze the BN's effects on different components of each term of the loss function in the Taylor series expansion.

The training loss of the i-th sample can be represented as a function of the standardized feature $\mathbf{y}^{(i)}$. We use the Taylor series expansion to decompose the loss function *w.r.t.* $\mathbf{y}^{(i)}$ into terms of multiple orders, as follows.

$$\text{Loss}(\mathbf{y}^{(i)};\tilde{\mathbf{y}}) = \text{Loss}(\tilde{\mathbf{y}}) + (\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{g} + \frac{1}{2!}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{H}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}) + R_2(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}) \tag{5}$$

where $\mathbf{g} = [\mathbf{g}_1, \cdots, \mathbf{g}_D] \in \mathbb{R}^D$ and $\mathbf{H} = (\mathbf{H}_{j,k}) \in \mathbb{R}^{D\times D}$ denote the gradient and the Hessian matrix of $\text{Loss}(\mathbf{y}^{(i)})$ at a fixed point $\tilde{\mathbf{y}} = [\tilde{\mathrm{y}}_1, \tilde{\mathrm{y}}_2, \cdots, \tilde{\mathrm{y}}_D] \in \mathbb{R}^D$; $R_2(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})$ denotes the sum of high-order terms.

Loss decomposition. Let $\text{Loss}^{\text{batch}} \in \mathbb{R}$ denote the overall loss of samples in a mini-batch.We consider the BN operation and decompose $\text{Loss}^{\text{batch}}$ in the following two cases.

The simplest case: Let us first consider the simplest case where all samples in a mini-batch share the same analytic formula of the loss function. Therefore, if we do the Taylor series expansion at the same fixed point $\tilde{\mathbf{y}}$ for loss function of each i-th sample $i = 1, \cdots, n$, the overall loss of samples in a mini-batch $\text{Loss}^{\text{batch}} = \sum_{i=1}^n \text{Loss}(\mathbf{y}^{(i)};\tilde{\mathbf{y}})$ can be re-written as the sum of four compositional terms, as follows.

$$\text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H}) \xlongequal{\textbf{decomposed into}} \text{Loss}^{\text{constant}} + \text{Loss}^{\text{grad}}(\mathbf{g}) + \text{Loss}^{\text{Hessian}}(\mathbf{H}) + \sum\nolimits_i R_2(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}) \tag{6}$$

where $\text{Loss}^{\text{constant}} = n\text{Loss}(\tilde{\mathbf{y}})$ is a constant *w.r.t.* input features $\mathbf{X}$ of the BN operation; $\text{Loss}^{\text{grad}}(\mathbf{g}) = \sum_{i=1}^n (\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{g}$ and $\text{Loss}^{\text{Hessian}}(\mathbf{H}) = \sum_{i=1}^n \frac{1}{2!}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{H}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})$ denote the first-order and second-order terms of the loss function in the Taylor expansion, respectively.

Proof.

$$\begin{aligned}
\text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H}) &= \sum_{i=1}^n \text{Loss}(\mathbf{y}^{(i)};\tilde{\mathbf{y}}) \\
&= \sum_{i=1}^n \left(\text{Loss}(\tilde{\mathbf{y}}) + (\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{g} + \frac{1}{2!}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{H}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}) + R_2(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}) \right) \\
&= n\text{Loss}(\tilde{\mathbf{y}}) + \sum_{i=1}^n (\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{g} + \sum_{i=1}^n \frac{1}{2!}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{H}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}) + \sum_{i=1}^n R_2(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}).
\end{aligned}$$

□

Then,

$$\text{Loss}^{\text{constant}} = n\text{Loss}(\tilde{\mathbf{y}}) \tag{7}$$

$$\text{Loss}^{\text{grad}}(\mathbf{g}) = \sum_{i=1}^n (\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{g} \tag{8}$$

$$\text{Loss}^{\text{Hessian}}(\mathbf{H}) = \sum_{i=1}^n \frac{1}{2}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top \mathbf{H}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}) \tag{9}$$

C.1 PROOF OF LEMMA 1

Lemma 1 further decomposes $\text{Loss}^{\text{Hessian}}(\mathbf{H})$ into two terms, as follows.

Lemma 1. *Let us set $\varepsilon = 0$ (the tiny positive constant ε is only used to avoid dividing zero, so we can ignore ε in all following paragraphs to simplify the proof). Then,* $\text{Loss}^{\text{Hessian}}(\mathbf{H})$ *can be decomposed into two terms, i.e.,*

$$\text{Loss}^{\text{Hessian}}(\mathbf{H}) = \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) + \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}),$$

where $\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) = \sum_{i=1}^{n} \frac{1}{2!}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}^{\text{diag}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})$*, and* $\text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) = \sum_{i=1}^{n} \frac{1}{2!}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}^{\text{off}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})$*;* $\mathbf{H}^{\text{diag}}$ *and* $\mathbf{H}^{\text{off}} = \mathbf{H} - \mathbf{H}^{\text{diag}}$ *denote the matrix only containing diagonal elements and the matrix only containing off-diagonal elements in* $\mathbf{H}$*, respectively.*

Proof.

$$\begin{aligned}
\text{Loss}^{\text{Hessian}}(\mathbf{H}) &= \sum_{i=1}^{n} \frac{1}{2}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \\
&= \sum_{i=1}^{n} \frac{1}{2}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \left(\mathbf{H}^{\text{diag}} + \mathbf{H}^{\text{off}}\right)(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \\
&= \sum_{i=1}^{n} \frac{1}{2}\left((\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}^{\text{diag}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) + (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}^{\text{off}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})\right) \\
&= \sum_{i=1}^{n} \frac{1}{2}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}^{\text{diag}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) + \sum_{i=1}^{n} \frac{1}{2}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}^{\text{off}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \\
&= \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) + \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})
\end{aligned} \tag{10}$$

□

C.2 Proof of Lemma 2

Lemma 2 further decomposes $\text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})$ into two terms, as follows.

Lemma 2. *Let* $\mathbf{x}_d = [\mathbf{X}_{d,1}, \mathbf{X}_{d,2}, \ldots, \mathbf{X}_{d,n}]^\top \in \mathbb{R}^n$ *denote the d-th feature dimension of all the n samples in a mini-batch. Then,* $\text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})$ *can be decomposed into the loss term depending on* $\mathbf{x}_d$ *(i.e.,* $L_d(\mathbf{H}^{\text{off}}_{d,:}) = \mathbf{H}^{\text{off}}_{d,:}(\mathbf{Y} - \tilde{\mathbf{y}}\mathbf{1}_n^\top)\mathbf{y}_d$*) and the loss term independent with* $\mathbf{x}_d$*, thereby* $\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} = \frac{\partial L_d(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$*, where* $\mathbf{H}^{\text{off}}_{d,:} = [\mathbf{H}^{\text{off}}_{d,1}, \mathbf{H}^{\text{off}}_{d,2}, \ldots, \mathbf{H}^{\text{off}}_{d,D}] \in \mathbb{R}^D$ *denotes the d-th row of* $\mathbf{H}^{\text{off}}$*,* $\mathbf{y}_d = [\mathbf{Y}_{d,1}, \mathbf{Y}_{d,2}, \ldots, \mathbf{Y}_{d,n}]^\top \in \mathbb{R}^n$*. Furthermore,* $L_d(\mathbf{H}^{\text{off}}_{d,:})$ *can be decomposed as*

$$L_d(\mathbf{H}^{\text{off}}_{d,:}) = L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:}) + L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:}),$$

where $L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:}) = \mathbf{H}^{\text{off}}_{d,:}\mathbf{Y}^{\text{linear}}\mathbf{y}_d$ *and* $L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:}) = \mathbf{H}^{\text{off}}_{d,:}(\mathbf{Y}^{\text{non}} - \tilde{\mathbf{y}}\mathbf{1}_n^\top)\mathbf{y}_d$*.* $\mathbf{Y}^{\text{linear}} = [\mathbf{o}_d^\top \mathbf{y}_1, \mathbf{o}_d^\top \mathbf{y}_2, \cdots, \mathbf{o}_d^\top \mathbf{y}_D]^\top \mathbf{o}_d^\top$ *and* $\mathbf{Y}^{\text{non}} = \mathbf{Y} - \mathbf{Y}^{\text{linear}}$*, where* $\mathbf{o}_d = \frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}$ *denotes the unit vector in the direction of* $\mathbf{y}_d$*. Then,* $L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:}) = \|\mathbf{y}_d\| \cdot \mathbf{H}^{\text{off}}_{d,:}[\mathbf{o}_d^\top \mathbf{y}_1, \mathbf{o}_d^\top \mathbf{y}_2, \cdots, \mathbf{o}_d^\top \mathbf{y}_D]^\top$*. Therefore,* $\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} = \frac{\partial L_d(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} = \frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} + \frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$.

Proof. We prove *Lemma 2* by two steps as follows.

Step 1: Firstly, we prove that $\text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})$ can be decomposed into the loss term depending on $\mathbf{x}_d$ (i.e., $L_d(\mathbf{H}^{\text{off}}_{d,:}) = \mathbf{H}^{\text{off}}_{d,:}(\mathbf{Y} - \tilde{\mathbf{y}}\mathbf{1}_n^\top)\mathbf{y}_d$) and the loss term independent with $\mathbf{x}_d$, thereby $\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} = \frac{\partial L_d(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$.

Based on *Lemma* 1,

$$
\begin{aligned}
\text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) &= \sum_{i=1}^{n} \frac{1}{2}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}^{\text{off}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \\
&= \sum_{i=1}^{n} \left(\frac{1}{2} \sum_{j=1}^{D} \sum_{k=1}^{D} \mathbf{H}^{\text{off}}_{j,k}(\mathbf{y}^{(i)}_j - \tilde{\mathbf{y}}_j)(\mathbf{y}^{(i)}_k - \tilde{\mathbf{y}}_k) \right) \\
&= \sum_{i=1}^{n} \frac{1}{2} \left(\sum_{j=1}^{D} \mathbf{H}^{\text{off}}_{d,j}(\mathbf{y}^{(i)}_d - \tilde{\mathbf{y}}_d)(\mathbf{y}^{(i)}_j - \tilde{\mathbf{y}}_j) + \sum_{k=1}^{D} \mathbf{H}^{\text{off}}_{k,d}(\mathbf{y}^{(i)}_k - \tilde{\mathbf{y}}_k)(\mathbf{y}^{(i)}_d - \tilde{\mathbf{y}}_d) \right) \\
&\quad + \sum_{i=1}^{n} \left(\frac{1}{2} \sum_{j=1, j\neq d}^{D} \sum_{k=1, k\neq d}^{D} \mathbf{H}^{\text{off}}_{j,k}(\mathbf{y}^{(i)}_j - \tilde{\mathbf{y}}_j)(\mathbf{y}^{(i)}_k - \tilde{\mathbf{y}}_k) \right) \\
&\qquad //\text{Note that the term } \mathbf{H}^{\text{off}}_{\mathrm{d,d}}(\mathbf{y}^{(\mathrm{i})}_{\mathrm{d}} - \tilde{\mathbf{y}}_{\mathrm{d}})(\mathbf{y}^{(\mathrm{i})}_{\mathrm{d}} - \tilde{\mathbf{y}}_{\mathrm{d}}) \text{ is computed twice for summation,} \\
&\qquad \text{but since that } \mathbf{H}^{\text{off}}_{\mathrm{d,d}} = 0 \text{ according to } \textit{Lemma } 1, \text{ this term is zero.} \\
&= \sum_{i=1}^{n} \frac{1}{2} \left(\sum_{j=1}^{D} \mathbf{H}^{\text{off}}_{d,j}(\mathbf{y}^{(i)}_d - \tilde{\mathbf{y}}_d)(\mathbf{y}^{(i)}_j - \tilde{\mathbf{y}}_j) + \sum_{k=1}^{D} \mathbf{H}^{\text{off}}_{k,d}(\mathbf{y}^{(i)}_k - \tilde{\mathbf{y}}_k)(\mathbf{y}^{(i)}_d - \tilde{\mathbf{y}}_d) \right) + \mathrm{L}^{\text{tmp-independent1}}_d \\
&\qquad //\text{Let } \mathrm{L}^{\text{tmp-independent1}}_{\mathrm{d}} \text{ denote } \sum_{\mathrm{i=1}}^{\mathrm{n}} \left(\frac{1}{2} \sum_{\mathrm{j=1,j\neq d}}^{\mathrm{D}} \sum_{\mathrm{k=1,k\neq d}}^{\mathrm{D}} \mathbf{H}^{\text{off}}_{\mathrm{j,k}}(\mathbf{y}^{(\mathrm{i})}_{\mathrm{j}} - \tilde{\mathbf{y}}_{\mathrm{j}})(\mathbf{y}^{(\mathrm{i})}_{\mathrm{k}} - \tilde{\mathbf{y}}_{\mathrm{k}}) \right) \text{ for simplicity.} \\
&= \sum_{i=1}^{n} \sum_{j=1}^{D} \mathbf{H}^{\text{off}}_{d,j}(\mathbf{y}^{(i)}_d - \tilde{\mathbf{y}}_d)(\mathbf{y}^{(i)}_j - \tilde{\mathbf{y}}_j) + \mathrm{L}^{\text{tmp-independent1}}_d \quad //(\mathbf{H}^{\text{off}} \text{ is symmetric.}) \\
&= \sum_{j=1}^{D} \mathbf{H}^{\text{off}}_{d,j} \left(\sum_{i=1}^{n} (\mathbf{y}^{(i)}_d - \tilde{\mathbf{y}}_d)(\mathbf{y}^{(i)}_j - \tilde{\mathbf{y}}_j) \right) + \mathrm{L}^{\text{tmp-independent1}}_d \\
&= \sum_{j=1}^{D} \mathbf{H}^{\text{off}}_{d,j}(\mathbf{y}_j - \tilde{\mathbf{y}}_j \mathbf{1}_n)^\top (\mathbf{y}_d - \tilde{\mathbf{y}}_d \mathbf{1}_n) + \mathrm{L}^{\text{tmp-independent1}}_d \\
&= \sum_{j=1}^{D} \mathbf{H}^{\text{off}}_{d,j}(\mathbf{y}_j - \tilde{\mathbf{y}}_j \mathbf{1}_n)^\top \mathbf{y}_d - \sum_{j=1}^{D} \mathbf{H}^{\text{off}}_{d,j}(\mathbf{y}_j - \tilde{\mathbf{y}}_j \mathbf{1}_n)^\top \tilde{\mathbf{y}}_d \mathbf{1}_n + \mathrm{L}^{\text{tmp-independent1}}_d \\
&= \mathbf{H}^{\text{off}}_{d,:} \left(\mathbf{Y} - \tilde{\mathbf{y}} \mathbf{1}_n^\top \right) \mathbf{y}_d - \tilde{\mathbf{y}}_d \sum_{j=1}^{D} \mathbf{H}^{\text{off}}_{d,j}(\mathbf{y}_j^\top \mathbf{1}_n - \tilde{\mathbf{y}}_j \mathbf{1}_n^\top \mathbf{1}_n) + \mathrm{L}^{\text{tmp-independent1}}_d \\
&= \mathbf{H}^{\text{off}}_{d,:} \left(\mathbf{Y} - \tilde{\mathbf{y}} \mathbf{1}_n^\top \right) \mathbf{y}_d - \tilde{\mathbf{y}}_d \sum_{j=1, j\neq d}^{D} \mathbf{H}^{\text{off}}_{d,j}(\mathbf{y}_j^\top \mathbf{1}_n - \tilde{\mathbf{y}}_j \mathbf{1}_n^\top \mathbf{1}_n) + \mathrm{L}^{\text{tmp-independent1}}_d \qquad //\mathbf{H}^{\text{off}}_{d,d} = 0 \\
&= \mathbf{H}^{\text{off}}_{d,:} \left(\mathbf{Y} - \tilde{\mathbf{y}} \mathbf{1}_n^\top \right) \mathbf{y}_d - \mathrm{L}^{\text{tmp-independent2}}_d + \mathrm{L}^{\text{tmp-independent1}}_d \\
&\qquad //\text{Let } \mathrm{L}^{\text{tmp-independent2}}_{\mathrm{d}} \text{ denote } \tilde{\mathrm{y}}_{\mathrm{d}} \sum_{\mathrm{j=1,j\neq d}}^{\mathrm{D}} \mathbf{H}^{\text{off}}_{\mathrm{d,j}}(\mathbf{y}_{\mathrm{j}}^\top \mathbf{1}_{\mathrm{n}} - \tilde{\mathbf{y}}_{\mathrm{j}} \mathbf{1}_{\mathrm{n}}^\top \mathbf{1}_{\mathrm{n}}) \text{ for simplicity.} \\
&= \mathrm{L}_d(\mathbf{H}^{\text{off}}_{d,:}) + \mathrm{L}^{\text{independent}}_d. \\
&\qquad //\text{Let } \mathrm{L_d}(\mathbf{H}^{\text{off}}_{\mathrm{d,:}}) \text{denote } \mathbf{H}^{\text{off}}_{\mathrm{d,:}} \left(\mathbf{Y} - \tilde{\mathbf{y}} \mathbf{1}_{\mathrm{n}}^\top \right), \text{ and let } \mathrm{L}^{\text{independent}}_{\mathrm{d}} \text{ denote } (-\mathrm{L}^{\text{tmp-independent2}}_{\mathrm{d}} + \mathrm{L}^{\text{tmp-independent1}}_{\mathrm{d}}).
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} &= \frac{\partial \left(\mathrm{L}_d(\mathbf{H}^{\text{off}}_{d,:}) + \mathrm{L}^{\text{independent}}_d(\mathbf{H}^{\text{off}}_{d,:}) \right)}{\partial \mathbf{x}_d} = \frac{\partial \mathrm{L}_d(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} + \frac{\partial \mathrm{L}^{\text{independent}}_d(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} \\
&= \frac{\partial L_d(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} + \mathbf{0} = \frac{\partial L_d(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}.
\end{aligned}
$$

Step 2: $L_d(\mathbf{H}^{\text{off}}_{d,:})$ can be futher decomposed into two terms, as follows.

$$
\begin{aligned}
L_d(\mathbf{H}^{\text{off}}_{d,:}) &= \mathbf{H}^{\text{off}}_{d,:}\left(\mathbf{Y}-\tilde{\mathbf{y}}\mathbf{1}_n^\top\right)\mathbf{y}_d \\
&= \mathbf{H}^{\text{off}}_{d,:}\left(\mathbf{Y}^{\text{linear}}+\mathbf{Y}^{\text{non}}-\tilde{\mathbf{y}}\mathbf{1}_n^\top\right)\mathbf{y}_d \qquad //\mathbf{Y}=\mathbf{Y}^{\text{linear}}+\mathbf{Y}^{\text{non}} \\
&= \mathbf{H}^{\text{off}}_{d,:}\mathbf{Y}^{\text{linear}}\mathbf{y}_d + \mathbf{H}^{\text{off}}_{d,:}\left(\mathbf{Y}^{\text{non}}-\tilde{\mathbf{y}}\mathbf{1}_n^\top\right)\mathbf{y}_d \\
&= L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:}) + L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:}).
\end{aligned} \tag{11}
$$

where $\mathbf{Y}^{\text{linear}} = \left[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D\right]^\top \mathbf{o}_d^\top \in \mathbb{R}^{D\times n}$, $\mathbf{o}_d = \frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}$ is the unit vector in the direction of $\mathbf{y}_d$, and $\mathbf{Y}^{\text{non}} = [\mathbf{y}_1 - \|\mathbf{y}_1\|\cos(\mathbf{y}_1,\mathbf{y}_d)\mathbf{o}_d, \cdots, \mathbf{y}_D - \|\mathbf{y}_D\|\cos(\mathbf{y}_D,\mathbf{y}_d)\mathbf{o}_d]^\top$.
In particular,

$$
\begin{aligned}
L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:}) &= \mathbf{H}^{\text{off}}_{d,:}\mathbf{Y}^{\text{linear}}\mathbf{y}_d = \mathbf{H}^{\text{off}}_{d,:}\left[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D\right]^\top \mathbf{o}_d^\top\mathbf{y}_d \\
&= \mathbf{H}^{\text{off}}_{d,:}\left[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D\right]^\top \left(\frac{\mathbf{y}_d^\top}{\|\mathbf{y}_d\|}\mathbf{y}_d\right) \\
&= \mathbf{H}^{\text{off}}_{d,:}\left[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D\right]^\top \left(\frac{\|\mathbf{y}_d\|^2}{\|\mathbf{y}_d\|}\right) \\
&= \|\mathbf{y}_d\| \cdot \mathbf{H}^{\text{off}}_{d,:}\left[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D\right]^\top
\end{aligned}
$$

In this way, for all d, the gradient of $L_d(\mathbf{H}^{\text{off}}_{d,:})$ *w.r.t.* $\mathbf{x}_d$ can be decomposed into two terms,

$$
\begin{aligned}
\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} &= \frac{\partial\left(L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:}) + L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})\right)}{\partial \mathbf{x}_d} \\
&= \frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} + \frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d},
\end{aligned} \tag{12}
$$

where $\frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$ reflects the interaction utility yielded by feature components in all dimensions that are linear-correlated with $\mathbf{y}_d$, *i.e.*, $[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D]^\top$, subject to $\mathbf{o}_d = \frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}$. $\frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d}$ reflects the interaction utility between $\mathbf{y}_d$ and feature components that are not linearly-correlated with $\mathbf{y}_d$, *i.e.*, removing linearly-correlated components, $\forall j, \mathbf{y}_j - \|\mathbf{y}_j\|\cos(\mathbf{y}_j,\mathbf{y}_d)\mathbf{o}_d$.

□

C.3 Proof of Theorem 4

Then let us consider **more general cases**, which may have the following two issues. (1) Loss functions of different samples in a mini-batch have different analytic formulas, owing to the diversity of labels, *i.e.*, $\text{Loss}(\mathbf{y}^{(i)}, \textit{label}^{(i)})$.(2) The gradient $\mathbf{g}$ and the Hessian matrix $\mathbf{H}$ of the loss are unstable, because the change of gating states in gating layers (*e.g.*, the ReLU layer) is discontinuous and unpredictable. In these case, we decompose the loss function for each i-th sample as *Theorem 4*.

Theorem 4. *The loss function for each i-th sample, $1 \le i \le n$, can be written as* $\text{Loss}(\mathbf{y}^{(i)};\tilde{\mathbf{y}}) = \text{Loss}(\tilde{\mathbf{y}}) + \text{Loss}^{\text{grad}}(\widehat{\mathbf{g}}) + \text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}}) + \text{Loss}^{\text{off}}(\widehat{\mathbf{H}}^{\text{off}}) + \Delta^{(i)}$, *where $\widehat{\mathbf{g}} = \frac{1}{n}\sum_{i=1}^n \mathbf{g}^{(i)}$ denotes the average gradient, $\widehat{\mathbf{H}}$ denotes the equivalent Hessian matrix. Both $\widehat{\mathbf{g}}$ and $\widehat{\mathbf{H}}$ are shared by all samples in the mini-batch. $\Delta^{(i)}$ denotes the distinctive term of the i-th sample, which sums up all errors caused by $\widehat{\boldsymbol{g}}$ and $\widehat{\boldsymbol{H}}$, but is not limited to terms of greater-than-two orders in the Taylor expansion.*

Proof. The loss function for each i-th sample, $1 \le i \le n$, can be written as

$$
\text{Loss}(\mathbf{y}^{(i)};\tilde{\mathbf{y}}) = \text{Loss}(\tilde{\mathbf{y}}) + (\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top\mathbf{g}^{(i)} + \frac{1}{2!}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}})^\top\mathbf{H}^{(i)}(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}) + R_2(\mathbf{y}^{(i)}-\tilde{\mathbf{y}}). \tag{13}
$$

where $\mathbf{g}^{(i)} \in \mathbb{R}^D$ and $\mathbf{H}^{(i)} = (\mathbf{H}_{j,k}) \in \mathbb{R}^{D\times D}$ denote the gradient and the Hessian matrix of $\text{Loss}(\mathbf{y}^{(i)})$ at the fixed point $\tilde{\mathbf{y}} \in \mathbb{R}^D$; $R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})$ denotes the sum of high-order terms.

Let $\widehat{\mathbf{g}} = \frac{1}{n}\sum_{i=1}^n \mathbf{g}^{(i)}$ and $\widehat{\mathbf{H}}$ denote the average gradient and an equivalent Hessian matrix (to be learned), respectively; then we can approximately represent losses of each i-th sample in a batch into the same Taylor-like formulation as

$$\text{Loss}(\mathbf{y}^{(i)}) = \text{Loss}(\tilde{\mathbf{y}}) + (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \widehat{\mathbf{g}} + \frac{1}{2!}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \widehat{\mathbf{H}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) + \Delta^{(i)}. \tag{14}$$

where $\Delta^{(i)} = (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top(\mathbf{g}^{(i)} - \widehat{\mathbf{g}}) + \frac{1}{2!}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top(\mathbf{H}^{(i)} - \widehat{\mathbf{H}})(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) + R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})$ sums up all errors caused by the average gradient $\widehat{\mathbf{g}}$ and the equivalent matrix $\widehat{\mathbf{H}}$, which is not limited to terms of greater-than-two orders in the Taylor expansion.

Further, we minimize $\sum_{i=1}^n \|\mathbf{g}^{(i)} - \widehat{\mathbf{g}} - \widehat{\mathbf{H}}(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})\|_2^2$ to learn the equivalent matrix $\widehat{\mathbf{H}}$ that minimize the approxiamte error.

□

D PROOFS OF THEOREMS 1, 2, 3, AND COROLLARY 1

In this section, we prove Theorems 1, 2, 3, and Corollary 1 in the main paper, which show the effects of the BN operation on each loss term in Equation (6) during the back-propagation in **Case1**.

Back-propagation through the BN layer. During the back-propagation, for all $d \in \{1, \cdots, D\}$, the gradient of $\text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})$ $w.r.t.$ the d-th feature dimension of all the n samples before the BN operation, $i.e.$, $\frac{\partial \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{x}_d}$, and the gradient of $\text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})$ $w.r.t.$ the d-th feature dimension of all the n samples after the standardization phase of the BN operation, $i.e.$, $\frac{\partial \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{y}_d}$, are connected by chain rule as follows.

$$\frac{\partial \text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})}{\partial \mathbf{x}_d} = \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\frac{\partial \text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})}{\partial \mathbf{y}_d} = \mathbf{J}_d \frac{\partial \text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})}{\partial \mathbf{y}_d}, \tag{15}$$

where we use $\mathbf{J}_d = \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}$ to denote the Jacobian matrix of $\mathbf{y}_d$ $w.r.t.$ $\mathbf{x}_d$.

According to Equation (4), $\mathbf{y}_d = \frac{\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n}{\boldsymbol{\sigma}_d}$, Therefore,

$$\begin{aligned}
\mathbf{J}_d &= \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d} \\
&= \left(\frac{\partial \boldsymbol{\mu}_d}{\partial \mathbf{x}_d}\frac{\partial \mathbf{y}_d^\top}{\partial \boldsymbol{\mu}_d} + \frac{\partial \boldsymbol{\sigma}_d^2}{\partial \mathbf{x}_d}\frac{\partial \mathbf{y}_d^\top}{\partial \boldsymbol{\sigma}_d^2} + \frac{\partial \mathbf{x}_d^\top}{\partial \mathbf{x}_d}\frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\right) \\
&= \left(\frac{\partial \left(\frac{1}{n}\mathbf{x}_d^\top \mathbf{1}_n\right)}{\partial \mathbf{x}_d}\frac{\partial \mathbf{y}_d^\top}{\partial \boldsymbol{\mu}_d} + \frac{\partial \left(\frac{1}{n}(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)^\top(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)\right)}{\partial \mathbf{x}_d}\frac{\partial \mathbf{y}_d^\top}{\partial \boldsymbol{\sigma}_d^2} + \frac{\partial \mathbf{x}_d^\top}{\partial \mathbf{x}_d}\frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\right) \\
&= \left(\frac{1}{n}\mathbf{1}_n\frac{\partial \mathbf{y}_d^\top}{\partial \boldsymbol{\mu}_d} + \frac{2}{n}(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)\frac{\partial \mathbf{y}_d^\top}{\partial \boldsymbol{\sigma}_d^2} + I\frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\right) \\
&= \left(\frac{1}{n}\mathbf{1}_n\frac{\partial \left(\frac{\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n}{\boldsymbol{\sigma}_d}\right)^\top}{\partial \boldsymbol{\mu}_d} + \frac{2}{n}(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)\frac{\partial \left(\frac{\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n}{\boldsymbol{\sigma}_d}\right)^\top}{\partial \boldsymbol{\sigma}_d^2} + \frac{\partial \left(\frac{\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n}{\boldsymbol{\sigma}_d}\right)^\top}{\partial \mathbf{x}_d}\right) \\
&= \left(-\frac{1}{n}\mathbf{1}_n\frac{1}{\boldsymbol{\sigma}_d}\mathbf{1}_n^\top + \frac{2}{n}(\mathbf{x}_1 - \boldsymbol{\mu}_d\mathbf{1}_n)(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)^\top\left(-\frac{1}{2}{\boldsymbol{\sigma}_d}^{-3}\right) + \frac{1}{\boldsymbol{\sigma}_d}I\right) \\
&= \frac{1}{\boldsymbol{\sigma}_d}\left(I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top - \frac{1}{n\boldsymbol{\sigma}_d^2}(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)^\top\right).
\end{aligned} \tag{16}$$

Based on *Lemma 1* and *Lemma 2*, we obtain the decomposition of $\frac{\partial \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{x}_d}$ and $\frac{\partial \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{y}_d}$ as follows.

$$\frac{\partial \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{x}_d} = \frac{\partial}{\partial \mathbf{x}_d}\left(\text{Loss}^{\text{constant}} + \text{Loss}^{\text{grad}}(\mathbf{g}) + \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) + \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) + \sum_{i=1}^{n} R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})\right) \tag{17}$$

$$= \frac{\partial \text{Loss}^{\text{constant}}}{\partial \mathbf{x}_d} + \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_d} + \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_d} + \frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} + \frac{\partial (\sum_{i=1}^{n} R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}))}{\partial \mathbf{x}_d}. \tag{18}$$

$$= \frac{\partial \text{Loss}^{\text{constant}}}{\partial \mathbf{x}_d} + \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_d} + \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_d} + \frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} + \frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} + \frac{\partial (\sum_{i=1}^{n} R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}))}{\partial \mathbf{x}_d}. \tag{19}$$

$$\frac{\partial \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{y}_d} = \frac{\partial}{\partial \mathbf{y}_d}\left(\text{Loss}^{\text{constant}} + \text{Loss}^{\text{grad}}(\mathbf{g}) + \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) + \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) + \sum_{i=1}^{n} R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}})\right) \tag{20}$$

$$= \frac{\partial \text{Loss}^{\text{constant}}}{\partial \mathbf{y}_d} + \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d} + \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d} + \frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{y}_d} + \frac{\partial (\sum_{i=1}^{n} R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}))}{\partial \mathbf{y}_d}. \tag{21}$$

$$= \frac{\partial \text{Loss}^{\text{constant}}}{\partial \mathbf{y}_d} + \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d} + \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d} + \frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{y}_d} + \frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{y}_d} + \frac{\partial (\sum_{i=1}^{n} R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}))}{\partial \mathbf{y}_d}. \tag{22}$$

Similarly by chain rule, we have

$$\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_d} = \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d} = \mathbf{J}_d \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d}. \tag{23}$$

$$\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_d} = \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d} = \mathbf{J}_d \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d}. \tag{24}$$

$$\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} = \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{y}_d} = \mathbf{J}_d \frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{y}_d}. \tag{25}$$

$$\frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} = \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{y}_d} = \mathbf{J}_d \frac{\partial L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{y}_d} \tag{26}$$

$$\frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} = \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{y}_d} = \mathbf{J}_d \frac{\partial L_d^{\text{non}}(\mathbf{H}^{\text{off}}_{d,:})}{\partial \mathbf{y}_d}. \tag{27}$$

Table 1: Several notations are summarized for reference.

n	the number of samples in a mini-batch
$\mathbf{x}^{(i)} \in \mathbb{R}^D$	the D-dimensional feature of the i-th sample **before** the BN operation
$\mathbf{y}^{(i)} \in \mathbb{R}^D$	the D-dimensional feature of the i-th sample **after** the standardization phase
$\mathbf{x}_d \in \mathbb{R}^n$	the d-th feature dimension of all the n samples **before** the BN operation
$\mathbf{y}_d \in \mathbb{R}^n$	the d-th feature dimension of all the n samples **after** the standardization phase
$\mathbf{J}_d \in \mathbb{R}^{n\times n}$	the Jacobian matrix of $\mathbf{y}_d$ $w.r.t.$ $\mathbf{x}_d$, $i.e.$ $\mathbf{J}_d = \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}$
$\tilde{\mathbf{y}} \in \mathbb{R}^D$	the point to conduct Taylor expansion of the loss function
$\mathbf{o}_d \in \mathbb{R}^D$	the unit vector in the direction of $\mathbf{y}_d$ $i.e.$ $\mathbf{o}_d = \frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}$

D.1 Proof of Theorem 1: the effects of $\text{Loss}^{\text{constant}}$ and $\text{Loss}^{\text{grad}}(\mathbf{G})$

In this section, we prove *Theorem 1* in Section 3 of the main paper, which shows the effects of the zeroth order term and the first order term of $\text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})$.

Theorem 1. $\text{Loss}^{\text{grad}}(\mathbf{g})$ *and* $\text{Loss}^{\text{constant}}$ *have no gradients on input features* $\mathbf{X}$ *of the BN operation, i.e.,* $\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{X}} = \mathbf{0}$ *and* $\frac{\partial \text{Loss}^{\text{constant}}}{\partial \mathbf{X}} = \mathbf{0}$.

Proof. (1) $\text{Loss}^{\text{constant}} = n\text{Loss}(\tilde{\mathbf{y}})$ is a constant, thus $\frac{\partial \text{Loss}^{\text{constant}}}{\partial \mathbf{x}_d} = \mathbf{0}$, $d = 1, \cdots, D$.
(2) In consideration of the connection that $\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_d} = \mathbf{J}_d \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d}$, which is shown in Equation (23), we first derive $\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d}$ as follows.

$$\begin{aligned}\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d} &= \frac{\partial \left(\sum_{i=1}^{n} (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{g}\right)}{\partial \mathbf{y}_d} = \frac{\partial \left(\mathbf{1}_n^\top \left(\mathbf{Y}^\top - \mathbf{1}_n \tilde{\mathbf{y}}^\top\right) \mathbf{g}\right)}{\partial \mathbf{y}_d} \\ &= \frac{\partial \left(\mathbf{1}_n^\top \sum_{j=1}^{D} \mathbf{g}_j (\mathbf{y}_j - \tilde{\mathbf{y}}_j \mathbf{1}_n)\right)}{\partial \mathbf{y}_d} = \frac{\partial \left(\mathbf{g}_d \mathbf{1}_n^\top (\mathbf{y}_d - \tilde{\mathbf{y}}_d \mathbf{1}_n)\right)}{\partial \mathbf{y}_d} = \mathbf{g}_d \mathbf{1}_n \in \mathbb{R}^n,\end{aligned} \tag{28}$$

where $\mathbf{g}_d \in \mathbb{R}$ is the d-th element of the gradient $\mathbf{g} \in \mathbb{R}^D$. Then,

$$\begin{aligned}\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_d} &= \mathbf{J}_d \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d} \\ &= \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top - \frac{1}{n \boldsymbol{\sigma}_d^2} (\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n) (\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n)^\top\right) \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d} \\ &= \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top - \frac{1}{n \boldsymbol{\sigma}_d^2} (\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n) (\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n)^\top\right) (\mathbf{g}_d \mathbf{1}_n) \\ &= \frac{\mathbf{g}_d}{\boldsymbol{\sigma}_d} \left(\mathbf{1}_n - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top \mathbf{1}_n - \frac{1}{n \boldsymbol{\sigma}_d^2} (\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n) (\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n)^\top \mathbf{1}_n\right) \\ &= \frac{\mathbf{g}_d}{\boldsymbol{\sigma}_d} \left(\mathbf{1}_n - \mathbf{1}_n - \frac{1}{n \boldsymbol{\sigma}_d^2} (\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n) \left(\mathbf{x}_d^\top \mathbf{1}_n - \boldsymbol{\mu}_d \mathbf{1}_n^\top \mathbf{1}_n\right)\right) \\ &= \frac{\mathbf{g}_d}{\boldsymbol{\sigma}_d} \left(\mathbf{0} - \frac{1}{n \boldsymbol{\sigma}_d^2} (\mathbf{x}_1 - \boldsymbol{\mu}_d \mathbf{1}_n) (n \boldsymbol{\mu}_d - n \boldsymbol{\mu}_d)\right) \\ &= \mathbf{0} \in \mathbb{R}^n.\end{aligned}$$

□

D.2 Proof of Theorem 2: the effects of $\text{Loss}^{\text{DIAG}}(\mathbf{H}^{\text{DIAG}})$.

In this section, we prove *Theorem 2* in Section 3 of the main paper, which shows the effects of diagonal elements in the Hessian matrix at the fixed point $\tilde{\mathbf{y}}$.

Theorem 2. $\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})$ *has no gradients on input features* $\mathbf{X}$ *of the BN operation, i.e.,* $\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{X}} = 0$.

Proof. Since that $\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{X}} = \left[\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_1}, \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_2}, \ldots; \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_D}\right]^\top$, it is equivalent to prove that $\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_d} = \mathbf{0}$, $\forall d \in \{1, \cdots, D\}$. According to Equation (24), $\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_d} = \mathbf{J}_d \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d}$, thus we first derive $\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d}$ as follows.

$$
\begin{aligned}
\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d} &= \frac{\partial \left(\sum_{i=1}^{n} \frac{1}{2} (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \mathbf{H}^{\text{diag}} (\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \right)}{\partial \mathbf{y}_d} \\
&= \frac{1}{2} \frac{\partial \left(\sum_{i=1}^{n} \sum_{j=1}^{D} \mathbf{H}^{\text{diag}}_{j,j} (\mathbf{y}^{(i)}_j - \tilde{\mathbf{y}}_j)^2 \right)}{\partial \mathbf{y}_d} \qquad //\tilde{\mathbf{y}}_j \in \mathbb{R} \text{ is the } j\text{-th element of point } \tilde{\mathbf{y}} \in \mathbb{R}^D \\
&= \frac{1}{2} \frac{\partial \left(\sum_{j=1}^{D} \mathbf{H}^{\text{diag}}_{j,j} \sum_{i=1}^{n} ((\mathbf{y}^{(i)}_j - \tilde{\mathbf{y}}_j)^2 \right)}{\partial \mathbf{y}_d} \\
&= \frac{1}{2} \frac{\partial \left(\sum_{j=1}^{D} \mathbf{H}^{\text{diag}}_{j,j} \left(\mathbf{y}_j - \tilde{\mathbf{y}}_j \mathbf{1}_n \right)^\top \left(\mathbf{y}_j - \tilde{\mathbf{y}}_j \mathbf{1}_n \right) \right)}{\partial \mathbf{y}_d} \\
&= \frac{1}{2} \frac{\partial \left(\mathbf{H}^{\text{diag}}_{d,d} \left(\mathbf{y}_d - \tilde{\mathbf{y}}_d \mathbf{1}_n \right)^\top \left(\mathbf{y}_d - \tilde{\mathbf{y}}_d \mathbf{1}_n \right) \right)}{\partial \mathbf{y}_d} \qquad //\tilde{\mathbf{y}}_d \in \mathbb{R} \text{ is the } d\text{-th element of point } \tilde{\mathbf{y}} \in \mathbb{R}^D \\
&= \mathbf{H}^{\text{diag}}_{d,d} \left(\mathbf{y}_d - \tilde{\mathbf{y}}_d \mathbf{1}_n \right),
\end{aligned}
$$

Then,

$$
\begin{aligned}
\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_d} &= \mathbf{J}_d \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d} \qquad // \text{ Equation (24)} \\
&= \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right)^\top \right) \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d} \\
&= \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right)^\top \right) \mathbf{H}^{\text{diag}}_{d,d} \left(\mathbf{y}_d - \tilde{\mathbf{y}}_d \mathbf{1}_n \right) \\
&= \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right)^\top \right) \mathbf{H}^{\text{diag}}_{d,d} \frac{\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n}{\boldsymbol{\sigma}_d} \\
&\quad - \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right)^\top \right) \mathbf{H}^{\text{diag}}_{d,d} \tilde{\mathbf{y}}_d \mathbf{1}_n \\
&= \frac{\mathbf{H}^{\text{diag}}_{d,d}}{\boldsymbol{\sigma}_d^2} \left(\left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right)^\top \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \right) \\
&\quad - \frac{\mathbf{H}^{\text{diag}}_{d,d} \tilde{\mathbf{y}}_d}{\boldsymbol{\sigma}_d} \left(\mathbf{1}_n - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top \mathbf{1}_n - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right)^\top \mathbf{1}_n \right) \\
&= \frac{\mathbf{H}^{\text{diag}}_{d,d}}{\boldsymbol{\sigma}_d^2} \left(\left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) - \frac{1}{n} \mathbf{1}_n \left(\mathbf{1}_n^\top \mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n^\top \mathbf{1}_n \right) - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) n \boldsymbol{\sigma}_d^2 \right) \\
&\quad - \frac{\mathbf{H}^{\text{diag}}_{d,d} \tilde{\mathbf{y}}_d}{\boldsymbol{\sigma}_d} \left(\mathbf{0} - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \left(\mathbf{x}_d^\top \mathbf{1}_n - \boldsymbol{\mu}_d \mathbf{1}_n^\top \mathbf{1}_n \right) \right) \\
&= \frac{\mathbf{H}^{\text{diag}}_{d,d}}{\boldsymbol{\sigma}_d^2} \left(\left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) - \frac{1}{n} \mathbf{1}_n \left(n \boldsymbol{\mu}_d - n \boldsymbol{\mu}_d \right) - \frac{n \boldsymbol{\sigma}_d^2}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \right) \\
&\quad - \frac{\mathbf{H}^{\text{diag}}_{d,d} \tilde{\mathbf{y}}_d}{\boldsymbol{\sigma}_d} \left(\mathbf{0} - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \left(n \boldsymbol{\mu}_d - n \boldsymbol{\mu}_d \right) \right) \\
&= \frac{\mathbf{H}^{\text{diag}}_{d,d}}{\boldsymbol{\sigma}_d^2} \left(\left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) - \mathbf{0} - \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \right) - \frac{\mathbf{H}^{\text{diag}}_{d,d} \tilde{\mathbf{y}}_d}{\boldsymbol{\sigma}_d} \left(\mathbf{0} - \frac{1}{n \boldsymbol{\sigma}_d^2} \left(\mathbf{x}_d - \boldsymbol{\mu}_d \mathbf{1}_n \right) \mathbf{0} \right) \\
&= \mathbf{0}.
\end{aligned}
$$

□

D.3 Proof of Theorem 3: the effects of $\text{Loss}^{\text{OFF}}(\mathbf{H}^{\text{OFF}})$.

In this section, we prove *Theorem 3* in Section 3 of the main paper, which shows the effects of off-diagonal elements in the Hessian matrix at the fixed point $\tilde{\mathbf{y}}$.

Theorem 3. *$\forall d$, $L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})$ has no gradients on the d-th feature dimension over all n samples $\mathbf{x}_d \in \mathbb{R}^n$, i.e.,*

$$\forall d,\ \frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d} = \mathbf{0},$$

thus $\frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} = \frac{\partial^2}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}}(A[\mathbf{o}_d^\top \mathbf{y}_1, \mathbf{o}_d^\top \mathbf{y}_2, \cdots, \mathbf{o}_d^\top \mathbf{y}_D]^\top) = \mathbf{0}$*, s.t.* $A = \|\mathbf{y}_d\| \cdot \mathbf{H}_{d,:}^{\text{off}}$*. On the other hand,* $\frac{\partial^2 L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} = \frac{1}{\boldsymbol{\sigma}_d} \cdot [(\mathbf{y}_1 - \|\mathbf{y}_1\| \cos(\mathbf{y}_1, \mathbf{y}_d)\mathbf{o}_d), \ldots, (\mathbf{y}_D - \|\mathbf{y}_D\| \cos(\mathbf{y}_D, \mathbf{y}_d)\mathbf{o}_d)] \neq \mathbf{0}$.

Proof. (1) For $L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}}) = \mathbf{H}_{d,:}^{\text{off}} \mathbf{Y}^{\text{linear}} \mathbf{y}_d$, we prove that $\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d} = \mathbf{0}$ as follows.

$$\begin{aligned}
\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d} &= \mathbf{J}_d \frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d} \qquad \text{//Equation (26)} \\
&= \mathbf{J}_d \frac{\partial \left(\mathbf{H}_{d,:}^{\text{off}} \mathbf{Y}^{\text{linear}} \mathbf{y}_d\right)}{\partial \mathbf{y}_d} \\
&= \mathbf{J}_d \left(\mathbf{H}_{d,:}^{\text{off}} \mathbf{Y}^{\text{linear}}\right)^\top \\
&= \mathbf{J}_d \left(\mathbf{H}_{d,:}^{\text{off}} \left[\mathbf{o}_d^\top \mathbf{y}_1, \mathbf{o}_d^\top \mathbf{y}_2, \cdots, \mathbf{o}_d^\top \mathbf{y}_D\right]^\top \mathbf{o}_d^\top\right)^\top \\
&= \mathbf{J}_d \mathbf{o}_d \left[\mathbf{o}_d^\top \mathbf{y}_1, \mathbf{o}_d^\top \mathbf{y}_2, \cdots, \mathbf{o}_d^\top \mathbf{y}_D\right] \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&= \mathbf{J}_d \mathbf{o}_d \boldsymbol{\lambda}_d^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \qquad \text{//Let } \boldsymbol{\lambda}_{\mathrm{d}} \in \mathbb{R}^{\mathrm{D}} \text{denote } \left[\mathbf{o}_{\mathrm{d}}^\top \mathbf{y}_1, \mathbf{o}_{\mathrm{d}}^\top \mathbf{y}_2, \cdots, \mathbf{o}_{\mathrm{d}}^\top \mathbf{y}_{\mathrm{D}}\right]^\top \text{forsimplicity.} \\
&= \mathbf{J}_d \frac{\mathbf{y}_d}{\|\mathbf{y}_d\|} \boldsymbol{\lambda}_d^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \qquad \text{//} \mathbf{o}_{\mathrm{d}} = \frac{\mathbf{y}_{\mathrm{d}}}{\|\mathbf{y}_{\mathrm{d}}\|} \\
&= \mathbf{J}_d \left(\frac{\mathbf{x}_d - \mu_d \mathbf{1}_n}{\sigma_d \|\mathbf{y}_d\|}\right) \boldsymbol{\lambda}_d^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&= \frac{1}{\sigma_d} \left(I - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top - \frac{1}{n\sigma_d^2} (\mathbf{x}_d - \mu_d \mathbf{1}_n)(\mathbf{x}_d - \mu_d \mathbf{1}_n)^\top\right) \left(\frac{\mathbf{x}_d - \mu_d \mathbf{1}_n}{\sigma_d \|\mathbf{y}_d\|}\right) \boldsymbol{\lambda}_d^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \qquad \text{//Equation (16)} \\
&= \frac{\boldsymbol{\lambda}_d^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top}{\sigma_d^2 \|\mathbf{y}_d\|} \left((\mathbf{x}_d - \mu_d \mathbf{1}_n) - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top (\mathbf{x}_d - \mu_d \mathbf{1}_n) - \frac{(\mathbf{x}_d - \mu_d \mathbf{1}_n)(\mathbf{x}_d - \mu_d \mathbf{1}_n)^\top (\mathbf{x}_d - \mu_d \mathbf{1}_n)}{n\sigma_d^2}\right) \\
&= \frac{\boldsymbol{\lambda}_d^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top}{\sigma_d^2 \|\mathbf{y}_d\|} \left((\mathbf{x}_d - \mu_d \mathbf{1}_n) - \frac{1}{n} \mathbf{1}_n \left(\mathbf{1}_n^\top \mathbf{x}_d - \mu_d \mathbf{1}_n^\top \mathbf{1}_n\right) - \frac{1}{n\sigma_d^2} (\mathbf{x}_d - \mu_d \mathbf{1}_n) n\sigma_d^2\right) \\
&= \frac{\boldsymbol{\lambda}_d^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top}{\sigma_d^2 \|\mathbf{y}_d\|} \left((\mathbf{x}_d - \mu_d \mathbf{1}_n) - \frac{1}{n} \mathbf{1}_n (n\mu_d - n\mu_d) - \frac{n\sigma_d^2}{n\sigma_d^2} (\mathbf{x}_d - \mu_d \mathbf{1}_n)\right) \\
&= \frac{\boldsymbol{\lambda}_d^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top}{\sigma_d^2 \|\mathbf{y}_d\|} ((\mathbf{x}_d - \mu_d \mathbf{1}_n) - \mathbf{0} - (\mathbf{x}_d - \mu_d \mathbf{1}_n)) \\
&= \mathbf{0}.
\end{aligned}$$

(2) For $L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})$, we first prove the mathematical expression of $\frac{\partial L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d}$ as follows.

$$
\begin{aligned}
\frac{\partial L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d} &= \mathbf{J}_d \frac{\partial \left(\mathbf{H}_{d,:}^{\text{off}}(\mathbf{Y}^{\text{non}} - \tilde{\mathbf{y}}\mathbf{1}_n^\top)\mathbf{y}_d\right)}{\partial \mathbf{y}_d} \quad \text{//Equation (27)} \\
&= \mathbf{J}_d \left(\mathbf{H}_{d,:}^{\text{off}} \mathbf{Y}^{\text{non}}\right)^\top - \mathbf{J}_d \left(\mathbf{H}_{d,:}^{\text{off}} (\tilde{\mathbf{y}}\mathbf{1}_n^\top)\right)^\top \\
&= \mathbf{J}_d (\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top - \mathbf{J}_d (\tilde{\mathbf{y}}\mathbf{1}_n^\top)^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&= \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top - \frac{1}{n\boldsymbol{\sigma}_d^2}(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)^\top\right)(\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&\quad - \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top - \frac{1}{n\boldsymbol{\sigma}_d^2}(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)(\mathbf{x}_d - \boldsymbol{\mu}_d\mathbf{1}_n)^\top\right)(\tilde{\mathbf{y}}\mathbf{1}_n^\top)^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&= \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top - \frac{1}{n}\mathbf{y}_d\mathbf{y}_d^\top\right)(\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top - \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top - \frac{1}{n}\mathbf{y}_d\mathbf{y}_d^\top\right)(\mathbf{1}_n\tilde{\mathbf{y}}^\top) \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&= \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top - \frac{1}{n}\mathbf{y}_d\mathbf{y}_d^\top\right)(\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top - \frac{1}{\boldsymbol{\sigma}_d} \left((\mathbf{1}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top\mathbf{1}_n) - \frac{1}{n}\mathbf{y}_d(\mathbf{y}_d^\top\mathbf{1}_n)\right)\tilde{\mathbf{y}}^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&= \frac{1}{\boldsymbol{\sigma}_d} \left(I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top - \frac{1}{n}\mathbf{y}_d\mathbf{y}_d^\top\right)(\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top - \mathbf{0} \\
&= \frac{1}{\boldsymbol{\sigma}_d} \left((\mathbf{Y}^{\text{non}})^\top - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top(\mathbf{Y}^{\text{non}})^\top - \frac{1}{n}\mathbf{y}_d\mathbf{y}_d^\top(\mathbf{Y}^{\text{non}})^\top\right)\left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&= \frac{1}{\boldsymbol{\sigma}_d} (\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top - \frac{1}{\boldsymbol{n}\boldsymbol{\sigma}_d}\mathbf{1}_n\mathbf{1}_n^\top (\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top - \frac{1}{\boldsymbol{n}\boldsymbol{\sigma}_d}\mathbf{y}_d\mathbf{y}_d^\top (\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&= \frac{1}{\boldsymbol{\sigma}_d} (\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top - \frac{1}{\boldsymbol{n}\boldsymbol{\sigma}_d}\mathbf{1}_n \left[\mathbf{1}_n^\top (\mathbf{Y}^{\text{non}})^\top\right] \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top - \frac{1}{\boldsymbol{n}\boldsymbol{\sigma}_d}\mathbf{y}_d \left[\mathbf{y}_d^\top (\mathbf{Y}^{\text{non}})^\top\right] \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top \\
&= \frac{1}{\boldsymbol{\sigma}_d} (\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top - \mathbf{0} - \mathbf{0} \quad \text{//}\mathbf{1}_n^\top(\mathbf{Y}^{\text{non}})^\top = \mathbf{0} \text{ and } \mathbf{y}_d^\top(\mathbf{Y}^{\text{non}})^\top = \mathbf{0} \text{ are proved later.} \\
&= \frac{1}{\boldsymbol{\sigma}_d} (\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}}\right)^\top.
\end{aligned}
$$

where $\mathbf{1}_n^\top(\mathbf{Y}^{\text{non}})^\top = \mathbf{0}$ and $\mathbf{y}_d^\top(\mathbf{Y}^{\text{non}})^\top = \mathbf{0}$ are computed as follows.

Note that $\mathbf{Y}^{\text{linear}} = \left[\mathbf{o}_d^\top\mathbf{y}_1, \mathbf{o}_d^\top\mathbf{y}_2, \cdots, \mathbf{o}_d^\top\mathbf{y}_D\right]^\top \mathbf{o}_d^\top = \boldsymbol{\lambda}_d\mathbf{o}_d^\top$, where $\mathbf{o}_d^\top\mathbf{y}_j \in \mathbb{R}, j = 1, \cdots, D$.

$$
\begin{aligned}
\mathbf{1}_n^\top(\mathbf{Y}^{\text{non}})^\top &= \mathbf{1}_n^\top(\mathbf{Y} - \mathbf{Y}^{\text{linear}})^\top = \mathbf{1}_n^\top\mathbf{Y}^\top - \mathbf{1}_n^\top\mathbf{o}_d\boldsymbol{\lambda}_d^\top \\
&= \mathbf{1}_n^\top\mathbf{Y}^\top - \mathbf{1}_n^\top\frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}\boldsymbol{\lambda}_d^\top = \mathbf{0} - 0 \cdot \frac{\boldsymbol{\lambda}_d^\top}{\|\mathbf{y}_d\|} \\
&= \mathbf{0} \in \mathbb{R}^{1\times D}.
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{y}_d^\top(\mathbf{Y}^{\text{non}})^\top &= \mathbf{y}_d^\top \left(\mathbf{Y} - \mathbf{Y}^{\text{linear}}\right)^\top \\
&= \mathbf{y}_d^\top \left(\mathbf{y}_1 - (\mathbf{o}_d^\top\mathbf{y}_1)\mathbf{o}_d, \cdots, \mathbf{y}_D - (\mathbf{o}_d^\top\mathbf{y}_D)\mathbf{o}_d\right) \\
&= \mathbf{y}_d^\top \left(\mathbf{y}_1 - \frac{\mathbf{y}_d^\top\mathbf{y}_1}{\|\mathbf{y}_d\|}\cdot\frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}, \cdots, \mathbf{y}_D - \frac{\mathbf{y}_d^\top\mathbf{y}_D}{\|\mathbf{y}_d\|}\cdot\frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}\right) \\
&= \left(\mathbf{y}_d^\top \left(\mathbf{y}_1 - \frac{\mathbf{y}_d^\top\mathbf{y}_1}{\|\mathbf{y}_d\|}\cdot\frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}\right), \cdots, \mathbf{y}_d^\top \left(\mathbf{y}_D - \frac{\mathbf{y}_d^\top\mathbf{y}_D}{\|\mathbf{y}_d\|}\cdot\frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}\right)\right) \\
&= \left(\mathbf{y}_d^\top\mathbf{y}_1 - \frac{\mathbf{y}_d^\top\mathbf{y}_1}{\|\mathbf{y}_d\|^2}\mathbf{y}_d^\top\mathbf{y}_d, \cdots, \mathbf{y}_d^\top\mathbf{y}_D - \frac{\mathbf{y}_d^\top\mathbf{y}_D}{\|\mathbf{y}_d\|^2}\mathbf{y}_d^\top\mathbf{y}_d\right) \\
&= \left(\mathbf{y}_1^\top\mathbf{y}_d - \mathbf{y}_1^\top\mathbf{y}_d, \cdots, \mathbf{y}_D^\top\mathbf{y}_d - \mathbf{y}_D^\top\mathbf{y}_d\right) \\
&= (0, \cdots, 0) \\
&= \mathbf{0} \in \mathbb{R}^{1\times D}.
\end{aligned}
$$

Then,

$$\frac{\partial^2 L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} = \frac{\partial}{\partial \mathbf{H}_{d,:}^{\text{off}}} \left(\frac{\partial L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d} \right) = \frac{\partial}{\partial \mathbf{H}_{d,:}^{\text{off}}} \left(\frac{1}{\boldsymbol{\sigma}_d} (\mathbf{Y}^{\text{non}})^\top \left(\mathbf{H}_{d,:}^{\text{off}} \right)^\top \right)^\top = \frac{1}{\boldsymbol{\sigma}_d} (\mathbf{Y}^{\text{non}})^\top$$
$$= \frac{1}{\boldsymbol{\sigma}_d} \cdot [\mathbf{y}_1 - \|\mathbf{y}_1\| \cos(\mathbf{y}_1, \mathbf{y}_d)\mathbf{o}_d, \cdots, \mathbf{y}_D - \|\mathbf{y}_D\| \cos(\mathbf{y}_D, \mathbf{y}_d)\mathbf{o}_d] .$$

□

Note that $\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d}$ reflects the interaction utility yielded by feature components in all dimensions that are linear-correlated with $\mathbf{y}_d$, *i.e.*, $[\mathbf{o}_d^\top \mathbf{y}_1, \mathbf{o}_d^\top \mathbf{y}_2, \cdots, \mathbf{o}_d^\top \mathbf{y}_D]^\top$, subject to $\mathbf{o}_d = \frac{\mathbf{y}_d}{\|\mathbf{y}_d\|}$. $\frac{\partial L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d}$ reflects the interaction utility between $\mathbf{y}_d$ and feature components that are not linearly-correlated with $\mathbf{y}_d$, *i.e.*, removing linearly-correlated components, $\forall j, \mathbf{y}_j - \|\mathbf{y}_j\| \cos(\mathbf{y}_j, \mathbf{y}_d)\mathbf{o}_d$. Empirically, linearly-correlated feature components usually represent similar concepts, thereby having stronger interaction utilities than non-correlated feature components. According to Theorem 3, the interaction utility between linearly-correlated feature components cannot pass through the BN operation, *i.e.*, $\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d} = \mathbf{0}$. Therefore, for each dimension d, we can consider that a considerable ratio of the influence of $\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d}$ cannot pass through the BN operation.

D.4 Proof of Corollary 1.

Corollary 1. *Based on Theorems 1, 2, and 3, we can prove that in the training phase of a neural network with BN operations,* $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{g}} = \mathbf{0}$, $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} = \mathbf{0}$, *and* $\forall d$, $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} = \frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} + \frac{\partial^2 L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}}$; *where* $\frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} = \mathbf{0}$. *In contrast, in the testing phase,* $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{g}} \neq \mathbf{0}$, $\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g},\mathbf{H})}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} \neq \mathbf{0}$, *and* $\frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} \neq \mathbf{0}$.

Proof. Remember that $\text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})$ is decomposed into five components, as follows,

$$\text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H}) = \text{Loss}^{\text{constant}} + \text{Loss}^{\text{grad}}(\mathbf{g}) + \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) + \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) + \sum\nolimits_i R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \tag{29}$$

Therefore, we get

$$\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})}{\partial \mathbf{X} \partial \mathbf{g}} = \frac{\partial^2}{\partial \mathbf{X} \partial \mathbf{g}} \left(\text{Loss}^{\text{constant}} + \text{Loss}^{\text{grad}}(\mathbf{g}) + \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) + \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) + \sum\nolimits_i R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \right)$$
$$= \frac{\partial^2}{\partial \mathbf{X} \partial \mathbf{g}} \left(\text{Loss}^{\text{grad}}(\mathbf{g}) \right) = \frac{\partial}{\partial \mathbf{g}} \left(\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{X}} \right) \overset{\textbf{Thm 1}}{=\!=} \frac{\partial}{\partial \mathbf{g}} (\mathbf{0}) = \mathbf{0};$$

$$\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} = \frac{\partial^2}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} \left(\text{Loss}^{\text{constant}} + \text{Loss}^{\text{grad}}(\mathbf{g}) + \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) + \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) + \sum\nolimits_i R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \right)$$
$$= \frac{\partial^2}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} \left(\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) \right) = \frac{\partial}{\partial \mathbf{H}^{\text{diag}}} \left(\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{X}} \right) \overset{\textbf{Thm 2}}{=\!=} \frac{\partial}{\partial \mathbf{H}^{\text{diag}}} (\mathbf{0}) = \mathbf{0};$$

$$\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} = \frac{\partial^2}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} \left(\text{Loss}^{\text{constant}} + \text{Loss}^{\text{grad}}(\mathbf{g}) + \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) + \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) + \sum\nolimits_i R_2(\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) \right)$$
$$= \frac{\partial^2}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} \left(\text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}}) \right) = \frac{\partial}{\partial \mathbf{H}_{d,:}^{\text{off}}} \left(\frac{\partial \text{Loss}^{\text{off}}(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} \right) \overset{\textbf{Lemma 2}}{=\!=} \frac{\partial}{\partial \mathbf{H}_{d,:}^{\text{off}}} \left(\frac{\partial L_d(\mathbf{H}^{\text{off}})}{\partial \mathbf{x}_d} \right)$$
$$\overset{\textbf{Lemma 2}}{=\!=} \frac{\partial}{\partial \mathbf{H}_{d,:}^{\text{off}}} \left(\frac{\partial \left(L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}}) + L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}}) \right)}{\partial \mathbf{x}_d} \right) = \frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} + \frac{\partial^2 L_d^{\text{non}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}},$$

where

$$\frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} = \frac{\partial}{\partial \mathbf{H}_{d,:}^{\text{off}}} \left(\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d} \right) \overset{\textbf{Thm 3}}{=\!=} \frac{\partial}{\partial \mathbf{H}_{d,:}^{\text{off}}} (\mathbf{0}) = \mathbf{0}.$$

In contrast, in the testing phase, people use the the population statistics, rather than mini-batch statistics for the standardization phase. Therefore, the means and variances are fixed during inference, the standardization phase is simply a linear transform applied to each activation. In this way, the Jacobian matrix $\mathbf{J}_d = \frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}$ reduces to an identity matrix, $i.e.$, $\mathbf{J}_d = I \in \mathbb{R}^{n \times n}$, which causes that the gradient of each loss term in Equation (29) $w.r.t.$ $\mathbf{x}_d$ is equal to the gradient $w.r.t.$ $\mathbf{y}_d$. Specifically,

$$\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_d} = \mathbf{J}_d \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d} = I \frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d} = I \left(\mathbf{g}_d \mathbf{1}_n \right) = \mathbf{g}_d \mathbf{1}_n, \tag{30}$$

$$\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_d} = \mathbf{J}_d \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d} = I \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d} = \mathbf{H}_{d,d}^{\text{diag}} \left(\mathbf{y}_d - \tilde{\mathrm{y}}_d \mathbf{1}_n \right), \tag{31}$$

$$\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d} = \mathbf{J}_d \frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d} = I \frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{y}_d} = \left(\mathbf{H}_{d,:}^{\text{off}} \mathbf{Y}^{\text{linear}} \right)^\top. \tag{32}$$

Further, we get

$$\begin{aligned}
\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})}{\partial \mathbf{X} \partial \mathbf{g}} &= \frac{\partial^2}{\partial \mathbf{X} \partial \mathbf{g}} \left(\text{Loss}^{\text{grad}}(\mathbf{g}) \right) = \frac{\partial}{\partial \mathbf{g}} \left(\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{X}} \right) \\
&= \frac{\partial}{\partial \mathbf{g}} \left(\left[\left(\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_1} \right)^\top ; \left(\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_2} \right)^\top ; \ldots ; \left(\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_D} \right)^\top \right] \right) \\
&= \frac{\partial}{\partial \mathbf{g}} \left(\left[\mathbf{g}_1 \mathbf{1}_n^\top ; \mathbf{g}_2 \mathbf{1}_n^\top ; \ldots ; \mathbf{g}_D \mathbf{1}_n^\top \right] \right) = \frac{\partial}{\partial \mathbf{g}} \left(\mathbf{g} \mathbf{1}_n^\top \right) \neq \mathbf{0} \in \mathbb{R}^{D^2 \times n},
\end{aligned}$$

$$\begin{aligned}
\frac{\partial^2 \text{Loss}^{\text{batch}}(\mathbf{g}, \mathbf{H})}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} &= \frac{\partial^2}{\partial \mathbf{X} \partial \mathbf{H}^{\text{diag}}} \left(\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}}) \right) = \frac{\partial}{\partial \mathbf{H}^{\text{diag}}} \left(\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{X}} \right) \\
&= \frac{\partial}{\partial \mathbf{H}^{\text{diag}}} \left(\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{X}} = \left[\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_1}, \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_2}, \ldots ; \frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_D} \right]^\top \right) \\
&= \frac{\partial}{\partial \mathbf{H}^{\text{diag}}} \left(\left[\mathbf{H}_{1,1}^{\text{diag}} \left(\mathbf{y}_1^\top - \tilde{\mathrm{y}}_1 \mathbf{1}_n^\top \right) ; \mathbf{H}_{2,2}^{\text{diag}} \left(\mathbf{y}_2^\top - \tilde{\mathrm{y}}_2 \mathbf{1}_n^\top \right) ; \ldots ; \mathbf{H}_{D,1}^{\text{diag}} \left(\mathbf{y}_D^\top - \tilde{\mathrm{y}}_D \mathbf{1}_n^\top \right) \right] \right) \\
&= \frac{\partial}{\partial \mathbf{H}^{\text{diag}}} \left(\mathbf{H}^{\text{diag}} \left(\mathbf{Y} - \tilde{\mathbf{y}} \mathbf{1}_n^\top \right) \right) \neq \mathbf{0} \in \mathbb{R}^{D^2 \times D \cdot n},
\end{aligned}$$

$$\frac{\partial^2 L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d \partial \mathbf{H}_{d,:}^{\text{off}}} = \frac{\partial}{\partial \mathbf{H}_{d,:}^{\text{off}}} \left(\frac{\partial L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})}{\partial \mathbf{x}_d} \right) = \frac{\partial}{\partial \mathbf{H}_{d,:}^{\text{off}}} \left(\left(\mathbf{H}_{d,:}^{\text{off}} \mathbf{Y}^{\text{linear}} \right)^\top \right) \neq \mathbf{0} \in \mathbb{R}^{n \times D}.$$

□

E CONCLUSIONS OF GENERAL CASES AND THEIR PROOFS

In this section, we discuss the general cases where (1) different samples in a mini-batch have different analytic formulas of loss functions; (2) The gradient $\mathbf{g}$ and the Hessian matrix $\mathbf{H}$ of the loss are unstable, because the change of gating states in gating layers (*e.g.*, the ReLU layer) is discontinuous and unpredictable. In these cases, we can still get a shared Taylor expansion series by *Theorem 4*. To be specific, $\text{Loss}(\mathbf{y}^{(i)}) = \text{Loss}(\tilde{\mathbf{y}}) + (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \widehat{\mathbf{g}} + \frac{1}{2!} (\mathbf{y}^{(i)} - \tilde{\mathbf{y}})^\top \widehat{\mathbf{H}} (\mathbf{y}^{(i)} - \tilde{\mathbf{y}}) + \Delta^{(i)}$.

Just like the simplest case, we prove that $\text{Loss}^{\text{grad}}(\widehat{\mathbf{g}})$ and $\text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}})$ have no gradients on features before the BN operation, and $\text{Loss}^{\text{off}}(\widehat{\mathbf{H}}^{\text{off}})$ does not have strong effects on features before the BN operation.

Proof. Based on Equation (14) in *Theorem 4*, we get an analytic formula shared by all samples in the mini-batch, *i.e.*, $\text{Loss}^{\text{share}} = \text{Loss}(\tilde{\mathbf{y}}) + \text{Loss}^{\text{grad}}(\widehat{\mathbf{g}}) + \text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}}) + \text{Loss}^{\text{off}}(\widehat{\mathbf{H}}^{\text{off}})$, which reduces to the simplest case. Theorems 1, 2, and 3 tells that $\frac{\partial \text{Loss}^{\text{grad}}(\widehat{\mathbf{g}})}{\partial \mathbf{x}_d} = \mathbf{0}$, $\frac{\partial \text{Loss}^{\text{diag}}(\widehat{\mathbf{H}}^{\text{diag}})}{\partial \mathbf{x}_d} = \mathbf{0}$, $\frac{\partial L_d^{\text{linear}}(\widehat{\mathbf{H}}^{\text{off}}_{d,:})}{\partial \mathbf{x}_d} = \mathbf{0}$. □

F Proof of Discussion: the third and higher-order derivatives of the sigmoid function have small strengths when the classification is confident

In this section, we prove that the third and higher-order derivatives of the sigmoid function have small strengths when the classification is confident, which is mentioned in the discussion in the main paper on applying the same Taylor series expansion to all samples.

Proof. Let us take the loss function as $L = -\log \text{Sigmoid}(z) = -\log \frac{e^z}{1+e^z} = -z + \log(1+e^z)$ for example, where $z \in \mathbb{R}$ is the output of the network. Let $L^{(m)}(z) = \frac{d^m L}{dz}(z)$ denote the m-th derivative of L $w.r.t.$ z. We use the mathematical induction to prove that the highest order term of $L^{(m)}(z)$ is $\frac{(-1)^m e^{(m-1)z}}{(1+e^z)^m}$, thus we can write $L^{(m)}(z)$ as $L^{(m)}(z) = \frac{\sum_{k=1}^{m-2} a_k^{(m)} e^{k\cdot z} + (-1)^m e^{(m-1)z}}{(1+e^z)^m}$, where $a_k^{(m)}$ is the coefficient of $e^{k\cdot z}$ in $L^{(m)}(z)$.

Base case: When $m = 1$,

$$L'(z) = \frac{-1}{1+e^z} = \frac{(-1)^1 e^{(1-1)z}}{(1+e^z)^1}, \tag{33}$$

$$\tag{34}$$

Actually, cases for $m = 2, 3, 4$ can also be verified directly,

$$L''(z) = \frac{e^z}{(1+e^z)^2}, \tag{35}$$

$$L^{(3)}(z) = \frac{e^z - e^{2z}}{(1+e^z)^3}, \tag{36}$$

$$L^{(4)}(z) = \frac{e^z - 4e^{2z} + e^{3z}}{(1+e^z)^4}. \tag{37}$$

Inductive step: Assuming that the highest order term of $L^{(m)}(z)$ is $\frac{(-1)^m e^{(m-1)z}}{(1+e^z)^m}$, $L^{(m)}(z) = \frac{\sum_{k=1}^{m-2} a_k^{(m)} e^{k\cdot z} + (-1)^m e^{(m-1)z}}{(1+e^z)^m}$. Then the derivative of $\frac{(-1)^m e^{(m-1)z}}{(1+e^z)^m}$ gives the highest order term of $L^{(m+1)}(z)$,

$$
\begin{aligned}
&\frac{d}{dz}\left((-1)^m \cdot \frac{e^{(m-1)z}}{(1+e^z)^m}\right) \\
&= (-1)^m \cdot \frac{\left(e^{(m-1)z} \cdot (m-1) \cdot (1+e^z)^m - e^{(m-1)z} \cdot m \cdot (1+e^z)^{m-1} e^z\right)}{(1+e^z)^{2m}} \\
&= (-1)^m \cdot \frac{e^{(m-1)z} \cdot (m-1) \cdot (1+e^z) - e^{(m-1)z} \cdot m \cdot e^z}{(1+e^z)^{m+1}} \\
&= (-1)^m \cdot \frac{(m-1) \cdot e^{(m-1)z} + (m-1) \cdot e^{mz} - m \cdot e^{mz}}{(1+e^z)^{m+1}} \\
&= (-1)^m \cdot \frac{(m-1) \cdot e^{(m-1)z} - e^{mz}}{(1+e^z)^{m+1}} \\
&= \frac{(-1)^m \cdot (m-1) \cdot e^{(m-1)z}}{(1+e^z)^{m+1}} + \frac{(-1)^{m+1} \cdot e^{mz}}{(1+e^z)^{m+1}}.
\end{aligned}
$$

Conclusion: Since both the base case and the inductive step have been proven to be true, we have proved that the highest order term of $L^{(m)}(z)$ is $\frac{(-1)^m e^{(m-1)z}}{(1+e^z)^m}$, $L^{(m)}(z)$ can be written as $L^{(m)}(z) = \frac{\sum_{k=1}^{m-2} a_k^{(m)} e^{k \cdot z} + (-1)^m e^{(m-1)z}}{(1+e^z)^m}$.

Further, $\lim_{z \to +\infty} \frac{\sum_{k=1}^{m-2} a_k^{(m)} e^{k \cdot z} + (-1)^m e^{(m-1)z}}{(1+e^z)^m} = 0$ and $\lim_{z \to -\infty} \frac{\sum_{k=1}^{m-2} a_k^{(m)} e^{k \cdot z} + (-1)^m e^{(m-1)z}}{(1+e^z)^m} = 0$ mean that, when $z \in \mathbb{R}$ is far away from zero, each m-th derivative $L^{(m)}(z)$ has a small absolute value.

In this way, when the classification is confident, the output of the network $z \in \mathbb{R}$ would be far away from zero, thereby the third and higher-order derivatives of the sigmoid function have small strengths according to above analysis. □

G Proof of Discussion: our conclusions can be extended to DNNs having no second derivatives or zero second derivatives

In this section, we prove that if the neural network has no second derivations or has zero second derivatives, all our conclusions are still valid for the equivalent Hessian matrix.

Proof. (1) Firstly, the finite difference method (Gonnet & Scholl, 2009) it a classic method to compute derivatives numerically, even for functions which are not differentiable. Thus, we can compute second derivatives by the finite difference method to get an equivalent Hessian matrix.

(2) Secondly, all proofs of theorems in this paper are not involved with how we compute the Hessian matrix. Thus all our conclusions are valid for the equivalent Hessian matrix. In fact, the proof is the same as Section C and D. □

H Proof of the reason of blindness

In this section, we discuss the reason for the blindness of the BN operation. According to Equations (1) and (2), the BN operation contains two phases, the affine transformation phase and the standardization phase. We prove that derivatives of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ in the standardization phase eliminate the influence of $\text{Loss}^{\text{grad}}(\mathbf{g}), \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})$, and $L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})$, as follows.

Proof. As we can see in the proofs for Theorems 1, 2, 3, and Corollary 1, $\text{Loss}^{\text{grad}}(\mathbf{g}), \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})$, and $L_d^{\text{linear}}(\mathbf{H}_{d,:}^{\text{off}})$ still have influence on each $\mathbf{y}_d$, their gradients $w.r.t.$ $\mathbf{y}_d$ are not zero. However, the gradient of each loss term $w.r.t.$ $\mathbf{x}_d$ become zero, which is resulted by the elimination effect of the Jacobian matrix $\mathbf{J}_d =$

Table 2: Architectures of the revised AlexNets for experimental verification of Theorem 3.

Traditional AlexNet	AlexNet-1	AlexNet-2	AlexNet-3
Conv	Conv	Conv	Conv
ReLU	ReLU	ReLU	ReLU
MaxPool	MaxPool	MaxPool	MaxPool
Conv	Conv	Conv	Conv
ReLU	ReLU	ReLU	ReLU
MaxPool	MaxPool	MaxPool	MaxPool
Conv	Conv	Conv	Conv
ReLU	ReLU	ReLU	ReLU
Conv	Conv	Conv	Conv
ReLU	ReLU	ReLU	ReLU
Conv	Conv	Conv	Conv
ReLU	ReLU	ReLU	ReLU
MaxPool	MaxPool	MaxPool	MaxPool
FC	FC	FC	FC
ReLU, dropout	ReLU	ReLU	ReLU
FC	FC	FC	FC
ReLU, dropout	ReLU	ReLU	ReLU
FC	FC	FC	FC
	ReLU	ReLU	ReLU
	FC	FC	FC
	ReLU	ReLU	ReLU
	FC	FC	FC
	ReLU	ReLU	ReLU
	FC	FC	**BN**
	ReLU	ReLU	FC
	FC	**BN**	ReLU
	ReLU	FC	FC
	BN	ReLU	ReLU
	FC	FC	FC

$\frac{1}{\boldsymbol{\sigma}_d}\left(I-\frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top-\frac{1}{n\boldsymbol{\sigma}_d^2}\left(\mathbf{x}_d-\boldsymbol{\mu}_d\mathbf{1}_n\right)\left(\mathbf{x}_d-\boldsymbol{\mu}_d\mathbf{1}_n\right)^\top\right)\in\mathbb{R}^{n\times n}$. Specifically, let *loss* denote any one of the above loss terms, we have

$$\frac{\partial loss}{\partial \mathbf{x}_d}==\frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}\frac{\partial loss}{\partial \mathbf{y}_d}=\mathbf{J}_d\frac{\partial loss}{\partial \mathbf{y}_d},$$

For $\text{Loss}^{\text{grad}}(\mathbf{g})$, $\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{y}_d}=\mathbf{g}_d\mathbf{1}_n\in\mathbb{R}^n$. $\mathbf{J}_d\cdot\mathbf{1}_n=\mathbf{0}$, thereby $\frac{\partial \text{Loss}^{\text{grad}}(\mathbf{g})}{\partial \mathbf{x}_d}=\mathbf{0}$; for $\text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})$, $\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{y}_d}=\mathbf{H}^{\text{diag}}_{d,d}\left(\mathbf{y}_d-\tilde{\text{y}}_d\mathbf{1}_n\right)$. $\mathbf{J}_d\cdot\mathbf{1}_n=\mathbf{0}$ and $\mathbf{J}_d\cdot\mathbf{y}_d=\mathbf{0}$, thereby $\frac{\partial \text{Loss}^{\text{diag}}(\mathbf{H}^{\text{diag}})}{\partial \mathbf{x}_d}=\mathbf{0}$; for $L_d^{\text{linear}}(\mathbf{H}^{\text{off}}_{d,:})$, the gradient $w.r.t.$ $\mathbf{x}_d$ becoming zero is also caused by the effect that $\mathbf{J}_d\cdot\mathbf{y}_d=\mathbf{0}$.

However, as we have shown in the proof of Corollary 1, when the means and variances are fixed, the Jacobian matrix $\mathbf{J}_d=\frac{\partial \mathbf{y}_d^\top}{\partial \mathbf{x}_d}$ reduces to an identity matrix, which would not eliminate each $\frac{\partial loss}{\partial \mathbf{y}_d}$ to zero. Therefore, we can attribute the blindnesss problem to derivatives of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ in the standardization phase.

□

I MORE EXPERIMENTAL SETTINGS AND RESULTS

I.1 ARCHITECTURES OF THE REVISED ALEXNETS AND THE REVISED LENETS FOR EXPERIMENTAL VERIFICATION OF THEOREM 3

This subsection provides more experimental details about the verification of Theorem 3 in Section 4 in the paper. In order to comprehensively test the BN on different layers, we revised the AlexNet (Krizhevsky et al., 2012) by adding five additional FC layers before the top FC layer, and revised the LeNet (LeCun et al., 1989) by adding seven additional FC layers before the top FC layer.

Table 3: Architectures of the revised LeNets for experimental verification of Theorem 3.

Traditional LeNet	LeNet-1	LeNet-2	LeNet-3
Conv	Conv	Conv	Conv
Sigmoid	ReLU	ReLU	ReLU
AveragePool	MaxPool	MaxPool	MaxPool
Conv	Conv	Conv	Conv
Sigmoid	ReLU	ReLU	ReLU
AveragePool	MaxPool	MaxPool	MaxPool
FC	FC	FC	FC
Sigmoid	ReLU	ReLU	ReLU
FC	FC	FC	FC
Sigmoid	ReLU	ReLU	ReLU
FC	FC	FC	FC
	ReLU	ReLU	ReLU
	FC	FC	FC
	ReLU	ReLU	ReLU
	FC	FC	FC
	ReLU	ReLU	ReLU
	FC	FC	FC
	ReLU	ReLU	ReLU
	FC	FC	FC
	ReLU	ReLU	ReLU
	FC	FC	**BN**
	ReLU	ReLU	FC
	FC	**BN**	ReLU
	ReLU	FC	FC
	BN	ReLU	ReLU
	FC	FC	FC

Each added FC layer contained 20 neurons and followed by a ReLU layer. For each DNN, we added a BN layer (where $\varepsilon = 1e-20$) before the 1st, 2nd, and 3rd top FC layers, respectively, to construct *AlexNet-1, AlexNet-2, AlexNet-3* and *LeNet-1, LeNet-2, LeNet-3*, as shown in Tables 2 and 3. All DNNs were trained on the MNIST dataset (LeCun et al., 1998).

Note that accept for adding additional FC layers on the top of the traditional AlexNet and the traditional LeNet, we also made the following revisions to improve the training efficiency of the above DNNs. For the revised architectures of AlexNe, we removed the dropout operation to facilitate the calculation of the Hessian matrix (Cohen et al., 2020). For the revised architectures of LeNet, we replaced all Sigmoid functions with currently-widely-used ReLU functions, and replaced all AveragePool operations with currently-widely-used MaxPool operations.

I.2 Details about how to inverse features in RealNVP-LN

This subsection provides details about how to inverse features in RealNVP-LN, which was constructed by replacing all BN operations with LN operations in the traditional architecture of RealNVP. Given an input image, the traditional RealNVP (*i.e.*, the RealNVP-BN in the paper) contained two learning phases, the forward phase that outputted a latent variable and predicted the log-likelihood of the input image estimated by the model, and the inverse phase that inversed the input image from a output variable of the forward phase. In the inverse phase of the RealNVP-BN, the BN operation depended on a population mean and a population variance to inverse the normalized features to features before the BN operation. The population mean and the population variance were computed based on paremeters μ and σ of all mini-batches, thereby completely representing all mini-batches. In order to inverse features in RealNVP-LN, each time we performed the LN operation, we performed the BN operation in parallel, and updated a population mean and a population variance after the conduction of the BN operation. Then, in the inverse phase of the RealNVP-LN, we made the LN operation depend on such population mean and such population variance of the BN operaiton.

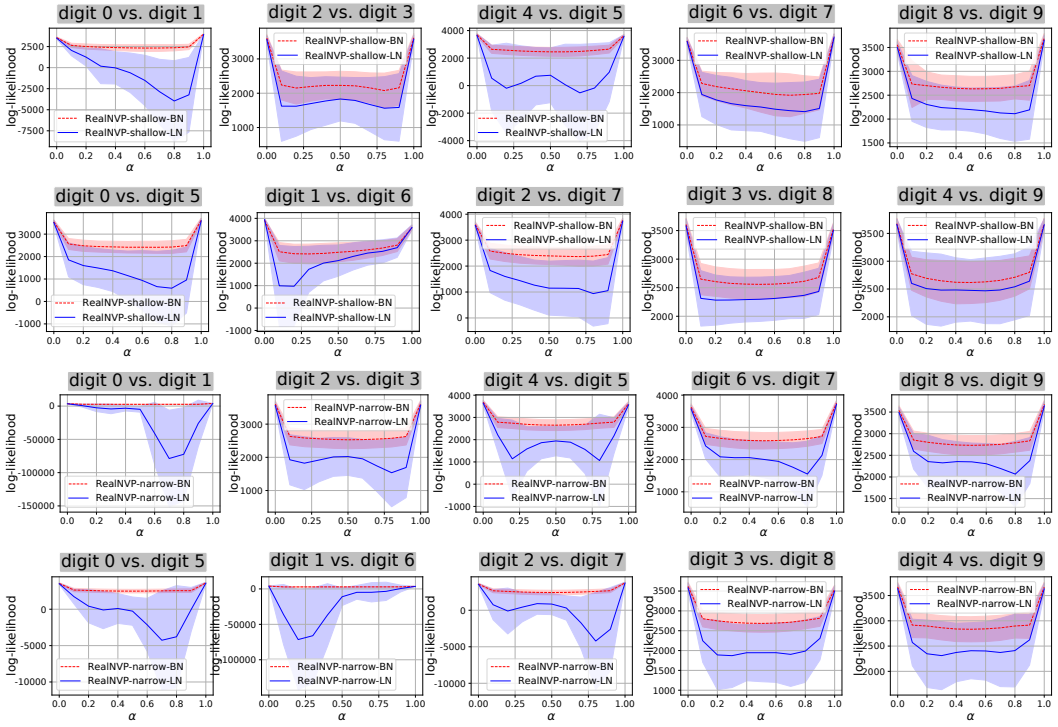

Figure 1: Log-likelihood of real images (at $\alpha = 0$ and $\alpha = 1$) and interpolated images generated by the RealNVP-shallow-BN, RealNVP-shallow-LN, RealNVP-narrow-BN, and RealNVP-narrow-LN. The shaded area represents the standard deviation.

I.3 More RealNVP models with various revised architectures

This subsection provides results on more RealNVP models with various revised architectures, which also yielded the similar conclusion as the conclusion in the paper, *i.e.*, the RealNVP-BN could not significantly distinguish real images and interpolated images. Specifically, we revised the traditional RealNVP architecture to obtain two different versions of RealNVP, *i.e.*, a shallow one (termed RealNVP-shallow) and a narrow one (termed RealNVP-narrow). The RealNVP-shallow was constructed by reducing the number of residual blocks in each residual module to 1/2 of that of the traditional RealNVP, and the RealNVP-narrow was constructed by reducing the number of feature maps of each residual block to 1/2 of that of the traditional RealNVP. For the RealNVP-shallow and the RealNVP-narrow, we also constructed the DNN-BN and the DNN-LN. According to Figure 1, we could obtain the same conclusion that the RealNVP-shallow-LN and the RealNVP-narrow-LN usually assigned much higher log-likelihood with real images (*i.e.*, images at the points of $\alpha = 0$ and $\alpha = 1$) than interpolated images. In comparison, the RealNVP-shallow-BN and the RealNVP-narrow-BN could not significantly distinguish real images and interpolated images.

I.4 More experiments to measure the BN's effects on feature representations for classification

This subsection provides experiments on additional CNNs to measure the BN's effects on feature representations for classification. Specifically, we used two additional CNNs for comparison, including ResNet-50 and DenseNet-201. The original ResNet-50 and DenseNet-201 were denoted by the *ResNet-50-BN* and the *DenseNet-201-BN*, because these two CNNs already contained BN operations. We also constructed two competing neural networks by replacing all BN operations in the ResNet-50-BN and the DenseNet-201-BN with LN operations, termed the *ResNet-50-LN* and the *DenseNet-201-LN*, respectively. Each of all the four CNNs, i.e., the ResNet-50-BN, the ResNet-50-LN, the DenseNet-201-BN, and the DenseNet-201-LN, could be trained in two different manners, i.e., a CNN could be trained either when all samples in a mini-batch had the same label or when all

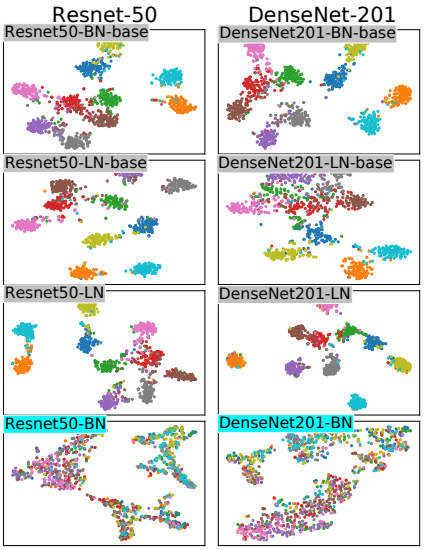

Figure 2: Comparing features of DNNs with BN layers and DNNs without BN layers. Points of different colors indicate samples of different categories. The DNN-BN usually learned much less clustered features than the DNN-LN, when these two DNNs were learned with the same settings.

samples in a mini-batch had different labels. Specifically, the CNN trained when all samples in a mini-batch had different labels was denoted with the suffix of *-base*. In this way, we trained a total of eight CNNs for comparison.

New experimental results in Figure 2 also support the conclusions in our paper, that is, when losses of all samples in a mini-batch shared the same analytic formula, features of the ResNet-50-BN/DenseNet-201-BN on different samples were far less clustered than those of the ResNet-50-LN/DenseNet-201-LN. This experiment indicated that the BN operation prevented the CNN from learning discriminative features. In comparison, when all samples in a mini-batch had different labels (thereby having different analytic formulas), features of the ResNet-50-BN-base/DenseNet-201-BN-base on different samples could be well clustered, which verified that the blindness problem could be alleviated by making the CNN be trained when all samples in a mini-batch had different labels in the classification task.

I.5 More experiments with more different settings of the Gaussian distribution

In the paper, in order to verify Theorem 1 and Theorem 2, we manually added noisy first derivatives and noisy second derivatives, and measured whether noisy derivatives could pass through the BN operation, so as to prove that the BN operation would block the back-propagation of elements in $\mathbf{g}$ and diagonal elements in $\mathbf{H}$.

In this section, we conducted more experiments to verify the blindness problem of the BN operation using more different settings of the Gaussian distribution. Specifically, we used Gaussian distributions with large positive/negative μ and large σ. We trained a VGG-11 network for image classification on the CIFAR-10 dataset, in which we added a BN layer before the second top FC layer. The VGG-11 was trained when all samples in a mini-batch shared the same label. We verified the blindness of the BN operation in the same way as used in the paper. Specifically, in the paper, we used metrics $\Delta\text{grad}^{\text{first}}$, $\Delta\text{grad}^{\text{second,diag}}$, and $\Delta\text{grad}^{\text{second,off}}$ to measure the influence of the first derivatives in $\mathbf{g}$, diagonal elements in $\mathbf{H}$, and off-diagonal elements in $\mathbf{H}$ on the gradient *w.r.t.* features before the BN operation.

We conducted experiments using four Gaussian distributions with large μ and large σ, including $\mu = -1, \sigma = 1$; $\mu = 1, \sigma = 1$; $\mu = -10, \sigma = 10$; and $\mu = +10, \sigma = 10$, respectively. Experimental results in Table 4 verified the conclusions in the paper. *I.e.*, $\Delta\text{grad}^{\text{first}} \approx 0$ and $\Delta\text{grad}^{\text{second,diag}} \approx 0$

Table 4: Verifying the BN's effects on the back-propagation of the first and second derivatives. $\Delta\text{grad}^{\text{first}} \approx 0$ and $\Delta\text{grad}^{\text{second,diag}} \approx 0$ indicated that the gradient $\mathbf{g}$ and diagonal elements in Hessian matrix $\mathbf{H}$ could not pass their influence through the BN operation. The small deviation was caused by the accumulation of tiny systematic computational errors in a DNN.

	$\Delta\text{grad}^{\text{first}}$	$\Delta\text{grad}^{\text{second,diag}}$	$\Delta\text{grad}^{\text{second,off}}$
$\mu = -1; \sigma = 1$	2.87e-14±4.38e-15	4.03e-08±8.75e-08	8.87e4±1.12e4
$\mu = 1; \sigma = 1$	2.80e-14±4.26e-15	4.02e-08±8.68e-08	8.85e4±1.12e4
$\mu = -10; \sigma = 10$	2.10e-13±3.24e-14	2.99e-07±6.48e-07	8.81e5±1.12e5
$\mu = 10; \sigma = 10$	2.07e-13±3.16e-14	2.99e-07±6.45e-07	8.80e5±1.12e5

proved that elements in $\mathbf{g}$ and diagonal elements in $\mathbf{H}$ could not pass their influence through the BN operation. The small deviation was caused by the accumulation of tiny systematic computational errors in a DNN. Besides, the large value of $\Delta\text{grad}^{\text{second,off}}$ indicated that off-diagonal elements in $\mathbf{H}$ could pass their influence through the BN operation.