Deep Perturbation Learning: Enhancing the Network Performance via Image Perturbations

Zifan Song [1 2] **Xiao Gong** [3] **Guosheng Hu** [4] **Cairong Zhao** [1 2]

Abstract

Image perturbation technique is widely used to generate adversarial examples to attack networks, greatly decreasing the performance of networks. Unlike the existing works, in this paper, we introduce a novel framework Deep Perturbation Learning (DPL), the new insights into understanding image perturbations, to *enhance* the performance of networks rather than decrease the performance. Specifically, we learn image perturbations to amend the data distribution of training set to improve the performance of networks. This optimization w.r.t data distribution is non-trivial. To approach this, we tactfully construct a differentiable optimization target w.r.t. image perturbations via minimizing the empirical risk. Then we propose an alternating optimization of the network weights and perturbations. DPL can easily be adapted to a wide spectrum of downstream tasks and backbone networks. Extensive experiments demonstrate the effectiveness of our DPL on 6 datasets (CIFAR-10, CIFAR-100, ImageNet, MS-COCO, PASCAL VOC, and SBD) over 3 popular vision tasks (image classification, object detection, and semantic segmentation) with different backbone architectures (*e.g.*, ResNet, MobileNet, and ViT).

1. Introduction

Image perturbation learning can greatly change the image distribution, effectively influencing the performance of deep models. In particular, Szegedy et al. (Szegedy et al., 2013) generate perturbed samples with adversarial distributions to fool DNNs, revealing the vulnerability of networks. Motivated by this, the existing works of image perturbations mainly focus on designing adversarial perturbations to attack or defense networks, *i.e.*, adversarial learning (Goodfellow et al., 2015; Carlini & Wagner, 2017; Tsipras et al., 2019; Zhao et al., 2020; Zhang et al., 2020; Xie et al., 2020; Dong et al., 2020; Chen et al., 2021). Can image perturbations only be used to attack networks to decrease the performance of networks? Can we learn perturbations to *enhance* the performance instead? Intuitively, it is feasible if we re-design the optimization targets. If it works well, we can introduce new insights into understanding image perturbations.

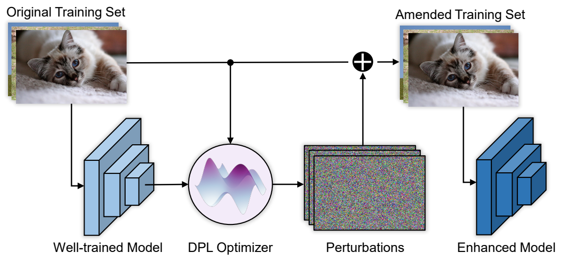

Figure 1. Deep Perturbation Learning (DPL). Given a well-trained model, DPL can optimize image perturbations and generate perturbed samples to amend the training set. Then, the network re-trained on the amended training set can enhance the performance.

Inspired by this, in this paper, we introduce a novel framework, Deep Perturbation Learning (DPL), shown in Fig. 1, to improve the performance of networks via optimizing model-dependent image perturbations. Given any well-trained networks, DPL can learn the image perturbations and add them to the training data to generate an amended training set. We then re-train the network with the new training set to improve the performance. Unlike previous adversarial learning approaches (Xie et al., 2020; Chen et al., 2021; Liu et al., 2022; Mei et al., 2022), we do not play a min-max optimization but minimize the *empirical risk* to improve the network performance. However, the optimization w.r.t. perturbations is not trivial. One intuitive optimization is brute-force search: re-training a large number of the networks with different perturbations and choosing the optimal one. Clearly, this is extremely computationally heavy. To

[1]Department of Computer Science and Technology, Tongji University, Shanghai, China [2]State Key Laboratory of Integrated Services Networks, Xidian University, Shaanxi, China [3]Department of Mathematics, Nanjing University, Nanjing, China [4]Oosto, 38330 Belfast, U.K.. Correspondence to: Cairong Zhao <zhaocairong@tongji.edu.cn>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

avoid the large-scale model re-training, we propose an efficient optimization strategy motivated by the calculus of variations (Gelfand & Fomin, 2000). Specifically, we construct a differentiable optimization objective and conduct an iterative training paradigm of weight stage and perturbation stage to optimize the network weights and image perturbations in an alternating way.

In theory, as a generic strategy, our DPL can be adopted to different backbone networks in a wide range of downstream tasks and DPL is also compatible with existing training strategies (*e.g.*, data augmentation). We evaluate the generalization capacity of DPL on 6 datasets (CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), ImageNet (Deng et al., 2009), MS-COCO (Lin et al., 2014), PASCAL VOC (Everingham et al., 2010), and SBD (Hariharan et al., 2011)) over 3 popular downstream tasks (image classification, object detection and semantic segmentation) with different backbone architectures (*e.g.*, ResNet (He et al., 2016a), DenseNet (Huang et al., 2017), MobileNetV2 (Sandler et al., 2018), and ViT (Dosovitskiy et al., 2020)). Our results demonstrate that, DPL can consistently improve the performance of the given well-trained models on various vision benchmarks. Moreover, we further provide extensive analysis to verify the effectiveness of our proposed scheme.

Our main contributions can be summarized as:

- Previous works of image perturbations focus on adversarial robustness, however, we propose Deep Perturbation Learning (DPL), the new insights into understanding image perturbations, to enhance the performance of networks by learning to amend the data distribution of training set.
- We formulate a differentiable function w.r.t. image perturbations to address the non-trivial optimization w.r.t data distribution. In particular, we introduce a differentiable objective, then DPL can iteratively learn the perturbations to amend the data distribution, aiming to improve the model performance.
- Extensive experimental results show that, with alternating optimization, DPL can consistently improve the performance of popular deep models (*e.g.*, ResNet (He et al., 2016a), DenseNet (Huang et al., 2017), and ViT (Dosovitskiy et al., 2020)) on a wide spectrum of downstream vision tasks (image classification, object detection, and semantic segmentation).

2. Related Work

2.1. Adversarial Examples

Adversarial examples are some well-crafted samples by adding imperceptible perturbations to the inputs leading to incorrect predictions. So far, there have been a lot of attacks such as FGSM (Goodfellow et al., 2015), CW (Carlini & Wagner, 2017). Nowadays, adversarial robustness is widely concerned. To obtain adversarial robust models, Tsipras et al. (Tsipras et al., 2019) propose a min-max optimization and adopt adversarial examples in the training progress. Although both the perturbations generated by adversarial attack methods and our DPL are imperceptible, the goals are different. The adversarial perturbations aim to cheat a given model, while we propose to achieve the better performance. In addition, unlike adversarial training (Shrivastava et al., 2017; Shafahi et al., 2019; Xie et al., 2020; Chen et al., 2021), our DPL does not play a min-max game but encounters a coupled optimization problem of minimizing the empirical risk.

2.2. Data Augmentation

Data augmentation is an effective way to improve model robustness. Specifically, some image processing operations such as rotation, translation, cropping and color distortion, are widely used for this purpose. Instead of these simple operations, policy-based methods such as AutoAugment (Cubuk et al., 2019), FastAutoAugment (Hataya et al., 2020), RandAugment (Cubuk et al., 2020), Adversarial AutoAugment (Zhang et al., 2019b), and PBA (Ho et al., 2019) carefully select a recipe of operations for data augmentation to reduce the model overfitting. The image perturbations observed during data augmentation often exhibit fixed patterns, which include geometric transformations, shearing, contrast adjustments, and more. Typically, a discrete search space is constructed based on these transformations, and the optimal data augmentation is found using non-differentiable search strategies. In contrast, we optimize the optimal image perturbations across the entire perturbation space, which leads to a superior generalization performance of the trained model. Moreover, our DPL does not conflict with existing data augmentation strategies and can be applied simultaneously to improve the generalization capability of models.

2.3. Influence Function

The influence function (IF) is mainly used to quantify the influence of a training sample on a test sample. Koh and Liang (Koh & Liang, 2017) introduce the influence function based on the terminology in robust statistics (Hampel et al., 2011) used a similar technique to estimate the model weights change as perturbing the training samples by an infinitesimal amount. Since the IF is hard to compute, Pruthi et al. (Pruthi et al., 2020) propose TracIn, a fast first-order approximation to IF. To extend influence function to generative models, Kong and Chaudhuri (Kong & Chaudhuri, 2021) introduce VAE-TracIn to give instance-based interpretations for variational auto-encoders. However, they focus on calculating the influence function but we aim to derive the perturbation formula to decrease the empirical risk.

3. Method

In this section, we first formulate the optimization problem and then present our training scheme Deep Perturbation Learning (DPL). Finally, we elaborate the proposed perturbation formula to amend the training samples in a direction that minimizes the empirical risk.

3.1. Preliminaries

Consider a learning problem from some input space $\mathcal{X}$ (*e.g.*, images) to an output space $\mathcal{Y}$ (*e.g.*, labels). We are given training samples $D = \{z_i\}_{i=1}^N$, where $z_i = (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, sampled independently from an unknown distribution P. Commonly, given a trained model f, $f_\theta : \mathcal{X} \to \mathcal{Y}$ is considered as a DNN function with θ as its weights and $L(z;\theta)$ is the loss of a training sample z. Respectively, the expected and empirical risks are defined by $R_{\text{exp}}(\theta) = E_{z\sim P}[L(z;\theta)]$ and $R_{\text{emp}}(\theta; D) = \frac{1}{N}\sum_{i=1}^N L(z_i;\theta)$. The empirical risk minimizer is given by $\hat{\theta} \triangleq \arg\min_\theta R_{\text{emp}}(\theta; D)$. For the sake of simplicity, we assume that the empirical risk is twice-differentiable and strictly convex in θ. Further discussions of relaxing these assumptions are contained in the Appendix.

3.2. Deep Perturbation Learning

The main idea of DPL is to perturb the training samples to amend the data distribution via minimizing the *empirical risk*. The optimization can be formulated as follows:

$$\hat{\delta} = \arg\min_{\delta} R_{\text{emp}}(\hat{\theta}_{D^\delta}; D). \tag{1}$$

In detail, $\delta = (\delta_1, \delta_2, ..., \delta_N)$ and D^δ denotes the perturbations of the training samples and the perturbed training set respectively, and $\hat{\theta}_{D^\delta}$ denotes the model weights trained on the perturbed set D^δ, *i.e.*,

$$\hat{\theta}_{D^\delta} = \arg\min_{\theta} R_{\text{emp}}(\theta; D^\delta). \tag{2}$$

The objective of Eq. (1) is to find an optimal perturbed training set on which the trained model can further minimize the empirical risk on the original training set D. Once Eq. (1) is solved, the weights $\hat{\theta}_{D^\delta}$ obtained in Eq. (2) is our expected model weights.

The bi-level optimization problem (**?**) Eq. (1) is non-trivial, in this work, this optimization is approximatively solved by two nested loops of optimization: (i) weight stage and (ii) perturbation stage. The steps (i) and (ii) are iterated to conduct the optimization. The objective of the weight stage is to minimize the empirical risk on the current training set D_k (the k iteration) by updating the model weights:

$$\hat{\theta}_k = \arg\min_{\theta} R_{\text{emp}}(\theta; D_k). \tag{3}$$

On the perturbation stage, the objective is to find a perturbation $\hat{\delta}_k$ which can amend the current training set D_k to a perturbed version $D_{k+1} \triangleq D_k^{\hat{\delta}_k}$ so that the model retrained on D_{k+1} can achieve smaller empirical risk on the original training set, *i.e.*,

$$\hat{\delta}_k = \arg\min_{\delta}[R_{\text{emp}}(\hat{\theta}_{k+1}; D) - R_{\text{emp}}(\hat{\theta}_k; D)]. \tag{4}$$

In our work, we propose two approaches to amend the training set:

(a) replacement (*i.e.*, $D^\delta \triangleq D + \delta$);

(b) augmentation (*i.e.*, $D^\delta \triangleq D \cup (D + \delta)$),
where $D + \delta \triangleq \{(x_i + \delta_i, y_i)\}_{i=1}^N$.

In general, the main idea of DPL is to find a sequence of datasets D_k and a sequence of weights $\hat{\theta}_k$ through the above two nested loops of optimization that converge to the minima of Eq. (1).

Unfortunately, the perturbation stage Eq. (4) is also intractable due to the same reason of Eq. (1). Thus, in the next section, we derive a perturbation formula to amend the current training samples in a direction that minimizes the empirical risk $R_{\text{emp}}(\theta; D)$ based on the current model $f_{\hat{\theta}}$.

3.3. Approximation of Perturbation Stage

In this section, we propose an effective optimization strategy that only trains *one* model for the perturbation stage formulated by Eq. (4). Our optimization can compute the model weight changes by perturbing the training samples at an infinitesimal amount and considers two scenarios: replacement and augmentation as shown in Eq. (4).

Replacement. We first clarify some terminologies. For a training sample $z = (x, y)$, define $z^\delta \triangleq (x + \delta, y)$. For a perturbation $\delta = (\delta_1, \delta_2, ..., \delta_N)$, the perturbed training set is denoted by $D + \delta = \{z_i^{\delta_i}\}_{i=1}^N$. In the replacement case, the empirical risk on the post-hoc training set $D^\delta \triangleq D + \delta$ is $R_{\text{emp}}(\theta; D^\delta) = \frac{1}{N}\sum_{i=1}^N L(z_i^{\delta_i};\theta)$. Recall that $\hat{\theta}_{D^\delta}$ is the minimizer of $R_{\text{emp}}(\theta; D^\delta)$.

After clarifying some terminologies, we discuss the optimization. As shown in Eq. (4), our goal is to minimize the difference $R_{\text{emp}}(\hat{\theta}_{D^\delta}; D) - R_{\text{emp}}(\hat{\theta}; D)$. The brute-force search has to re-train a large number of models to achieve $\hat{\theta}_{D^\delta}$, leading to heavy computations and the difficulty of optimizing the objective $R_{\text{emp}}(\hat{\theta}_{D^\delta}; D) - R_{\text{emp}}(\hat{\theta}; D)$ is that the objective function is not differentiable. Motivated by the calculus of variations (Gelfand & Fomin, 2000) and to avoid network re-training, we construct a differentiable function where $R_{\text{emp}}(\hat{\theta}_{D^\delta}; D)$ and $R_{\text{emp}}(\hat{\theta}; D)$ can be parameterized as special cases of this function. Thus, we introduce a variable ε and formulate this differentiable function:

$$\hat{\theta}_{\varepsilon,D^\delta} \triangleq \arg\min_{\theta}[\frac{1}{N}\sum_{i=1}^{N} L(z_i;\theta) + \frac{\varepsilon}{n}\sum_{i=1}^{n} L(z_i^{\delta_i};\theta) - \frac{\varepsilon}{n}\sum_{i=1}^{n} L(z_i;\theta)], \tag{5}$$

where n belongs to $\{1, 2, ..., N\}$ and controls the number of perturbed training samples (*i.e.*, the number of nonzero elements in δ). According to Eq. (5), we can see that $\hat{\theta}_{0,D^\delta} = \hat{\theta}$ and $\hat{\theta}_{\frac{n}{N},D^\delta} = \hat{\theta}_{D^\delta}$. Then,

$$\frac{d\hat{\theta}_{\varepsilon,D^\delta}}{d\varepsilon}|_{\varepsilon=0} = -\frac{1}{n}H_{\hat{\theta}}^{-1}[\sum_{i=1}^{n}(\nabla_\theta L(z_i^\delta,\hat{\theta}) - \nabla_\theta L(z_i,\hat{\theta}))], \tag{6}$$

where $H_{\hat{\theta}} \triangleq \frac{1}{N}\sum_{i=1}^{N}\nabla_\theta^2 L(z_i;\hat{\theta})$ is the Hessian and is positive definite (PD) by assumption. The detailed computations of Eq. (6) can be found in Appendix.

With Eq. (5) and (6), the optimization of Eq. (4) is differentiable. Then, we apply Taylor expansion to derive that

$$\begin{aligned} &R_{\text{emp}}(\hat{\theta}_{D^\delta};D) - R_{\text{emp}}(\hat{\theta};D) \\ \approx &\frac{n}{N}\frac{\partial R_{\text{emp}}(\hat{\theta}_{\varepsilon,D^\delta};D)}{\partial\varepsilon}|_{\varepsilon=0} \\ = &\frac{n}{N}\frac{\partial R_{\text{emp}}(\hat{\theta};D)}{\partial\theta}\frac{d\hat{\theta}_{\varepsilon,D^\delta}}{d\varepsilon}|_{\varepsilon=0} \\ = &-\frac{1}{N}\frac{\partial R_{\text{emp}}(\hat{\theta};D)}{\partial\theta}H_{\hat{\theta}}^{-1}[\sum_{i=1}^{n}(\nabla_\theta L(z_i^\delta,\hat{\theta}) - \nabla_\theta L(z_i,\hat{\theta}))] \\ \approx &-\frac{1}{N}\sum_{i=1}^{n}\frac{\partial R_{\text{emp}}(\hat{\theta};D)}{\partial\theta}H_{\hat{\theta}}^{-1}[\nabla_x\nabla_\theta L(z_i,\hat{\theta})\delta_i]. \end{aligned} \tag{7}$$

Obviously, we set δ_i in the same direction of $[\frac{\partial R_{\text{emp}}(\hat{\theta};D)}{\partial\theta}H_\theta^{-1}\nabla_x\nabla_\theta L(z_i,\hat{\theta})]^T$, then we can construct local perturbations of z_i that maximally decrease the risk $R_{\text{emp}}(\hat{\theta}_{D^\delta};D)$. The computations of Eq. (7) is detailed in Appendix.

In order to make the above estimation Eq. (7) sufficiently accurate, the size $\varepsilon = \frac{n}{N}$ and δ_i should be sufficiently small. This tells us that in our strategy we should only perturb small part of the training set. If the number of perturbed samples n is too large, the error of Eq. (7) will be out of tolerance; otherwise, if n is too small, the empirical risk cannot be effectively decreased. On the other hand, to ensure perturbing z_i by an infinitesimal amount, we set another hyper-parameter α to control the magnitude of perturbation δ_i, *i.e.*,

$$\delta_i = \alpha\cdot[\frac{\partial R_{\text{emp}}(\hat{\theta};D)}{\partial\theta}H_\theta^{-1}\nabla_x\nabla_\theta L(z_i,\hat{\theta})]^T. \tag{8}$$

In our implementation, we randomly select a subset of training samples in each loop to estimate the risk $R_{\text{emp}}(\hat{\theta};D)$ for reducing the computations and alleviate overfitting on the training set D. For the sake of discussion, we call the selected subset *guide set* denoted by G.

Augmentation. Since the key idea of augmentation is the same as the replacement case, we focus on discussing their difference.

The main difference is that the post-hoc training set D^δ becomes $D\cup(D+\delta)$, thus, the empirical risk on the post-hoc training set becomes

$$R_{\text{emp}}(\theta;D^\delta) = \frac{1}{N}\sum_{i=1}^{N}L(z_i;\theta) + \frac{1}{N}\sum_{i=1}^{N}L(z_i^{\delta_i};\theta). \tag{9}$$

We still let $\hat{\theta}_{D^\delta}$ be the minimizer of $R_{\text{emp}}(\theta;D^\delta)$. Similar as Eq. (5), the differentiable function is defined by:

$$\hat{\theta}_{\varepsilon,D^\delta} \triangleq \arg\min_\theta \frac{1}{N}\sum_{i=1}^{N}L(z_i;\theta) + \frac{\varepsilon}{n}\sum_{i=1}^{n}L(z_i^{\delta_i};\theta).$$

Accordingly, we have

$$\frac{d\hat{\theta}_{\varepsilon,D^\delta}}{d\varepsilon}|_{\varepsilon=0} = -\frac{1}{n}H_{\hat{\theta}}^{-1}\sum_{i=1}^{n}\nabla_\theta L(z_i^\delta,\hat{\theta}), \tag{10}$$

and

$$\begin{aligned} &R_{\text{emp}}(\hat{\theta}_{D^\delta};D) - R_{\text{emp}}(\hat{\theta};D) \\ \approx &-\frac{1}{N}\sum_{i=1}^{n}[\frac{\partial R_{\text{emp}}(\hat{\theta};D)}{\partial\theta}H_{\hat{\theta}}^{-1}\nabla_\theta L(z_i,\hat{\theta}) \\ &+ \frac{\partial R_{\text{emp}}(\hat{\theta};D)}{\partial\theta}H_{\hat{\theta}}^{-1}\nabla_x\nabla_\theta L(z_i,\hat{\theta})\delta_i]. \end{aligned} \tag{11}$$

Unlike Eq. (7), there is here an additional term, *i.e.*, the first term on the right of Eq. (11). In practice, we should guarantee $R_{\text{emp}}(\hat{\theta}_{D^\delta};D)$ is smaller than $R_{\text{emp}}(\hat{\theta};D)$, meaning that the empirical risk of the optimized model is smaller than the original model. Thus, we still let δ_i be the value in Eq. (8), at the same time, we sample n samples with the largest *positive* values Δ_i via

$$\begin{aligned} \Delta_i = &\frac{\partial R_{\text{emp}}(\hat{\theta};D)}{\partial\theta}H_{\hat{\theta}}^{-1}\nabla_\theta L(z_i,\hat{\theta}) \\ &+ \alpha\|\frac{\partial R_{\text{emp}}(\hat{\theta};D)}{\partial\theta}H_{\hat{\theta}}^{-1}\nabla_x\nabla_\theta L(z_i,\hat{\theta})\|_2. \end{aligned} \tag{12}$$

To illustrate the whole strategy, we present the pseudo-code of DPL (augmentation) in Algorithm 1 and provide the replacement version in the Appendix. In general, our approach is to find a sequence of weights $\hat{\theta}_k$ that iteratively decreases the empirical risk $R_{\text{emp}}(\hat{\theta};D)$. Related experiments demonstrate that our DPL can simultaneously reduce the empirical

Table 1. Performance on image classification and DPL can significantly improve the models' performance on CIFAR and ImageNet. The results of DPL are averaged over four independent runs and we report the DPL Iterations needed to achieve the enhanced networks.

	Model	Top-1 Acc.	Test Loss	**Top-1 Acc. (DPL)**	**Test Loss (DPL)**	**DPL Iter.**
CIFAR-10	VGG-16 (Simonyan & Zisserman, 2014)	92.76	0.251	93.29 (0.53 $\uparrow$)	0.232 (0.019 $\downarrow$)	3 iter.
	ResNet-18 (He et al., 2016a)	94.65	0.227	95.52 (0.87 $\uparrow$)	0.171 (0.056 $\downarrow$)	5 iter.
	ResNet-50 (He et al., 2016a)	94.92	0.207	95.90 (0.98 $\uparrow$)	0.180 (0.027 $\downarrow$)	3 iter.
	PreActResNet-18 (He et al., 2016b)	94.87	0.213	95.48 (0.61 $\uparrow$)	0.185 (0.027 $\downarrow$)	4 iter.
	DenseNet-121 (Huang et al., 2017)	94.84	0.212	95.27 (0.43 $\uparrow$)	0.194 (0.019 $\downarrow$)	5 iter.
	MobileNetV2-1.0 (Sandler et al., 2018)	94.33	0.234	94.77 (0.44 $\uparrow$)	0.205 (0.029 $\downarrow$)	3 iter.
	ViT-B (Dosovitskiy et al., 2020)	98.47	0.208	98.69 (0.22 $\uparrow$)	0.189 (0.019 $\downarrow$)	2 iter.
CIFAR-100	VGG-16 (Simonyan & Zisserman, 2014)	70.36	0.0144	72.37 (2.01 $\uparrow$)	0.0130 (0.0014 $\downarrow$)	4 iter.
	ResNet-18 (He et al., 2016a)	75.53	0.0082	76.32 (0.79 $\uparrow$)	0.0075 (0.0007 $\downarrow$)	3 iter.
	ResNet-50 (He et al., 2016a)	77.37	0.0079	78.74 (1.37 $\uparrow$)	0.0071 (0.0008 $\downarrow$)	5 iter.
	PreActResNet-18 (He et al., 2016b)	72.91	0.0103	73.86 (0.95 $\uparrow$)	0.0092 (0.0011 $\downarrow$)	4 iter.
	MobileNetV2-1.0 (Sandler et al., 2018)	67.75	0.0095	68.88 (1.13 $\uparrow$)	0.0083 (0.0012 $\downarrow$)	4 iter.
ImageNet	ResNet-18 (He et al., 2016a)	69.77	1.236	70.15 (0.38 $\uparrow$)	1.168 (0.068 $\downarrow$)	2 iter.
	ResNet-50 (He et al., 2016a)	76.22	1.108	76.58 (0.36 $\uparrow$)	1.057 (0.051 $\downarrow$)	3 iter.
	MobileNetV2-0.5 (Sandler et al., 2018)	64.47	1.535	65.18 (0.71 $\uparrow$)	1.485 (0.050 $\downarrow$)	1 iter.
	MobileNetV2-0.25 (Sandler et al., 2018)	52.25	2.151	52.59 (0.34 $\uparrow$)	2.122 (0.029 $\downarrow$)	1 iter.

Algorithm 1 Deep Perturbation Learning (DPL)

```
Require: $D_0 = \{(x_i^0, y_i)\}_{i=1}^N$, $\theta_0$, $n$, $\alpha$, $f_\theta$
Ensure: $\theta_T$
1: for t=0,1,...,T-1 do
2:     train $f_\theta$ on $D_t$ with $\theta_t$ as its initialization,
       output $\theta_{t+1}$
3:     compute $\Delta_i$ using Eq.(12)
4:     select $n$ samples with the largest positive $\Delta_i$
5:     $x_i^{t+1} = x_i^t + \alpha \cdot \frac{\partial R_{\text{emp}}(\theta_{t+1})}{\partial \theta} H_{\theta_{t+1}}^{-1} \nabla_x \nabla_\theta L(z_i, \theta_{t+1})$
6:     let $D_{t+1} = \{(x_i^{t+1}, y_i)\}_{i=1}^n \cup D_t$
7: end for
```

risk on the training and test set. However, we do not prove that the sequence converges to the minima of the original problem Eq. (2). It results from that over-reducing the empirical risk may lead to over-fitting. Thus, in practice, we stop the iterations when the loss of validation begins to increase. A detailed derivation of all the formulas can be found in the Appendix.

4. Experiments

Clearly, our DPL can be applied to many downstream tasks and can work with various backbone architectures. To verify the efficacy of our method, we evaluate DPL on different tasks: image classification, object detection and semantic segmentation with various backbone networks. Detailed information about datasets can be found in our Appendix.

4.1. Experimental Settings

Implementation Details. We use PyTorch (Paszke et al., 2017) to implement and train all the corresponding models in this paper. In the training phase, we follow the respective parameter settings of each baseline model. For the image classification task, we train the networks with batch size 128 for 200 epochs on CIFAR datasets and with random cropping and flipping. The networks are trained with stochastic gradient descent (SGD) (momentum = 0.9). The initial learning rate is set to 0.1 and then decay by 0.01 at 160 epochs and again at 180 epochs. For the object detection task, we train the models with a batch size 32 and an input size 300×300 for PASCAL VOC. We use the learning rate of 10^{-3} for 80k iterations, then continue training for 20k, 20k iterations with 10^{-4}, 10^{-5}, respectively. For the semantic segmentation task, we train our networks with batch size 16. We set the initial learning rate, momentum and weight decay to 0.009, 0.9 and 0.0001, respectively. We use scaling (0.5 to 2.0), cropping and flipping for all the training data. Following A^{2}MIM (Li et al., 2023), we adopt Cosine decay for 100 and 300 epochs pre-training for ViTs. In addition, we adopt synchronized batch normalization and multi-grid (Chen et al., 2017). In the perturbation stage, when calculating HVPs, we utilize stochastic estimation (Pearlmutter, 1994) to compute Eq. (8), which involves the inverse of Hessian term. We apply a backpropagation-like approach to compute the sum of gradients from network parameters to the loss, and recursively calculate inverse Hessian-vector products. We set the recursion depth as 5000 to balance precision with memory and time overhead. Meanwhile, the number of calculations needed to get the average value can be calculated as $(\text{training\_set\_size}/\text{recursion\_depth})$. The damping factor and scaling factor are set to 0.01 and 25, respectively.

Evaluation Protocols. For image classification, we adopt the commonly used Top-1 accuracy. For object detection, we use the mean Average Precision (mAP) and COCO standard metric (mAP@[.5, .95]). For semantic segmentation, the performance is evaluated in mIoU.

4.2. DPL on Downstream Tasks

Image Classification. We evaluate DPL on different backbone architectures, including VGG (Simonyan & Zisserman, 2014), ResNet (He et al., 2016a), PreActResNet (He et al., 2016b), DenseNet (Huang et al., 2017), MobileNetV2 (Sandler et al., 2018), and ViT-B (Dosovitskiy et al., 2020). We use the augmentation strategy and apply multiple iterations of DPL to each model until convergence. We report the averaged results over four independent runs with different initializations.

Table 2. Performance comparison among vanilla training (ResNet-50), other image perturbation methods and DPL in top-1 accuracy. AdvProp$^{\triangle}$ and AdvProp* represent versions that have applied 35-epoch PGD-1 and 70-epoch PGD-1, respectively.

Method	CIFAR-100	ImageNet
Vanilla Training	77.37	76.22
I-FGSM (Goodfellow et al., 2015)	-7.41	-4.26
PGD (Madry et al., 2017)	-5.67	-3.25
IAT (Lamb et al., 2019)	-2.14	-2.01
AdvProp$^{\triangle}$ (Xie et al., 2020)	-1.97	-2.19
AdvProp* (Xie et al., 2020)	+0.73	+0.12
Fast-AdvProp (Mei et al., 2022)	-	+0.30
DPL (ours)	**+1.37**	**+0.36**

We first thoroughly report our results on CIFAR-10, CIFAR-100 and ImageNet in Tab. 1. Compared with the baseline models, DPL can consistently improve the performance on both small-scale and large-scale datasets. In terms of top-1 accuracy, DPL achieves up to 0.98%, 2.01% and 0.71% improvement on CIFAR-10, CIFAR-100 and ImageNet, respectively. For test loss, ResNet-18 (He et al., 2016a) with DPL reaches a loss as low as 0.171, gaining nearly 25% reduction. In addition, we compare DPL with the previous works of image perturbations (*i.e.*, adversarial training) using ResNet-50, including I-FGSM (Goodfellow et al., 2015), AdvProp (Xie et al., 2020), and Fast-AdvProp (Mei et al., 2022). As reported in Tab. 2, DPL outperforms the vanilla training in top-1 accuracy of 1.37% and 0.36% on CIFAR-100 and ImageNet respectively, which is significantly higher than the adversarial training methods.

Object Detection. For object detection, we conduct experiments on PASCAL VOC and COCO. According to Tab. 3, compared to the vanilla training, our method consistently improves the performance. In additional, DPL outperforms AutoAugment (Zoph et al., 2020) and achieves very competitive performance against Det-AdvProp (Chen et al., 2021), but Det-AdvProp uses much more additional training data (100% vs. 8%). DPL also yields more competitive performance than the recently proposed method of leveraging adversarial examples, Fast-AdvProp (Mei et al., 2022). These results suggest that, compared to the heavy additional data burden imposed by other methods, DPL is sufficient to let networks achieve enhanced performance trained with a mixture of nearly 92% original images and only 8% perturbed samples, therefore making re-training networks for "free".

Table 3. Performance on object detection. Extra data means the augmented samples added to the training set. DPL gives rise to performance gains for all detectors with less extra data.

	Model	mAP
PASCAL VOC	Faster R-CNN (Ren et al., 2015)	73.1
	+ DPL (2 iter.)	73.5 (0.4 ↑)
	SSD300 (Liu et al., 2016)	77.2
	+ DPL (4 iter.)	77.6 (0.4 ↑)
	EfficientDet-D0 (Tan & Le, 2019)	77.4
	+ DPL (5 iter.)	77.9 (0.5 ↑)
COCO	MobileNetV2 (Sandler et al., 2018)	22.3
	+ DPL (3 iter.)	22.6 (0.3 ↑)
	EfficientDet-D0 (Tan & Le, 2019)	34.3
	+ AutoAugment (Zoph et al., 2020)	34.4 (0.1 ↑)
	+ Det-AdvProp (Chen et al., 2021)	34.7 (0.4 ↑)
	+ DPL (2 iter.)	34.6 (0.3 ↑)
	RetinaNet (Lin et al., 2017)	35.8
	+ Fast-AdvProp (Mei et al., 2022)	35.8 (0.0 ↑)
	+ DPL (2 iter.)	36.0 (0.2 ↑)

Semantic Segmentation. We conduct experiments on the PASCAL VOC and SBD datasets for the semantic segmentation task. We choose two popular methods EMANet (Li et al., 2019) and CFNet (Zhang et al., 2019a) with 3 backbones ResNet-50, ResNet-101 and ResNet-151. From Tab. 4, we can see that our DPL can consistently improve the performance in terms of mIoU on various methods and backbones. In particular, the improvement margin of small backbones is larger than bigger ones. We do not compare with adversarial training methods on semantic segmentation, since we do not find they are applied on this task to improve the performance, showing the advantage of our method, model- and task-agnostic.

Table 4. Performance comparison between DPL and the vanilla baseline on PASCAL VOC and SBD for semantic segmentation.

Method	mIoU	Test Loss
EMANet (R-50) (Li et al., 2019)	78.63	0.1407
+ DPL (4 iter.)	78.92	0.1315
EMANet (R-151) (Li et al., 2019)	87.94	0.0949
+ DPL (2 iter.)	87.99	0.0813
CFNet (R-101) (Zhang et al., 2019a)	85.90	0.1126
+ DPL (3 iter.)	86.14	0.1022

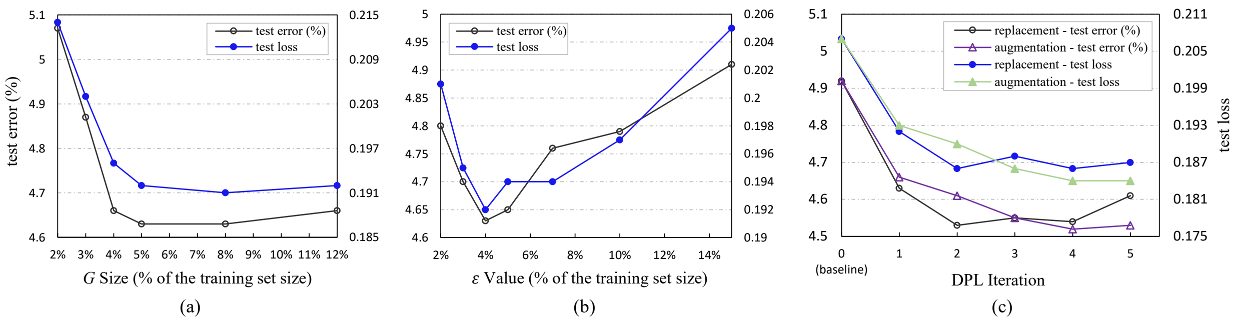

Figure 2. Ablation experiments conducted on CIFAR-10. (a): The size of guide set G. (b): The hyper-parameter ε. (c): The approaches of replacement and augmentation with DPL iterations.

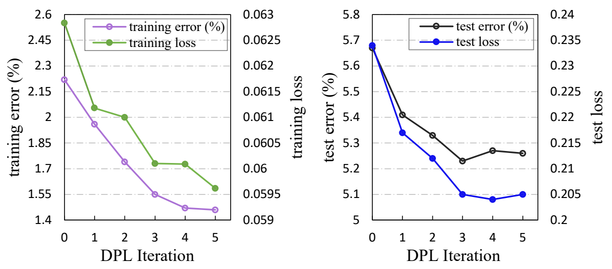

Figure 3. The efficacy of empirical risk minimization. Left: variation of empirical risk with DPL iterations. Right: variation of expected risk with DPL iterations.

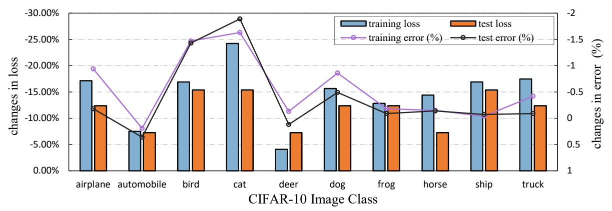

Figure 4. The changes of losses and error rates by DPL on CIFAR-10 classes. We invert the y-axis scales for better visualization.

4.3. Analytical Experiments

In this section, we conduct extensive experiments on CIFAR-10 to study the effects of the hyper-parameter α in Eq. (8), the size of our guide set, the hyper-parameter ε in Eq. (5), DPL iterations, and the cases of replacement and augmentation in Sec. 3.3. In addition, we provide qualitative results via class activation map (Zhou et al., 2016) and t-SNE (Linderman et al., 2017) to verify the effectiveness of our DPL.

Effects of α. As discussed in Sec. 3.3, the hyper-parameter α controls the strength of images perturbations. We conduct experiments with different values of α from 10^{-2} to 10^{-4}. And it should be noted that in these experiments, we replace the original images with amended ones, setting the value of ε and the guide set size as 4%, 4% of the training set size, respectively. As reported in Tab. 5, neither larger (α = 0.01) nor smaller (α=0.0001) values can lead to the best performance. α = 0.005 yields the lowest test error rate and we maintain this value in the performance experiments.

Table 5. Effects of the hyper-parameter α measured by test loss and error rates (%) with single-iteration DPL on CIFAR-10.

α	Test Loss	Top-1 Err.
0 (baseline)	0.207	4.92
0.01	0.195 (0.012 ↓)	4.84 (0.08 ↓)
0.005	0.195 (0.012 ↓)	**4.66 (0.26 ↓)**
0.001	**0.194 (0.013 ↓)**	4.79 (0.13 ↓)
0.0005	0.197 (0.010 ↓)	4.72 (0.20 ↓)
0.0001	0.199 (0.008 ↓)	4.74 (0.18 ↓)

Effects of G Size and ε. The guide set G in Sec. 3.3 and the hyper-parameter ε are key components of our approach. As a random subset of the original training set, the Guide Set G needs to be at a suitable size to better estimate the expected risk while maintaining efficiency. Fig. 2 (a) presents that our approach is particularly enhanced when the size of guide set increases from 2% to 4% of the training set, and nearly peaks in performance at 5%. Thus, we take 5% of the training set as our guide set to balance performance with computational cost. For the hyper-parameter ε, we perform image amendation for 2% to 15% of the images in training set. As shown in Fig. 2 (b), when the number of amended images reaches 4% of the training set size, the PreAct ResNet applied with single-iteration DPL achieves 4.63% error rate. In addition, the phenomenon of the error rate rebounding afterward verifies that amending too many images may overly disturb the feature distribution of the training samples.

Effects of DPL Iterations. DPL is iterative to keep improving the performance. Here, we conduct ablation experiments for DPL iterations on CIFAR-10 to explore the potential of our approach. Fig. 2 (c) indicates that multi-iterations DPL can bring additional enhancements to the model. Compared

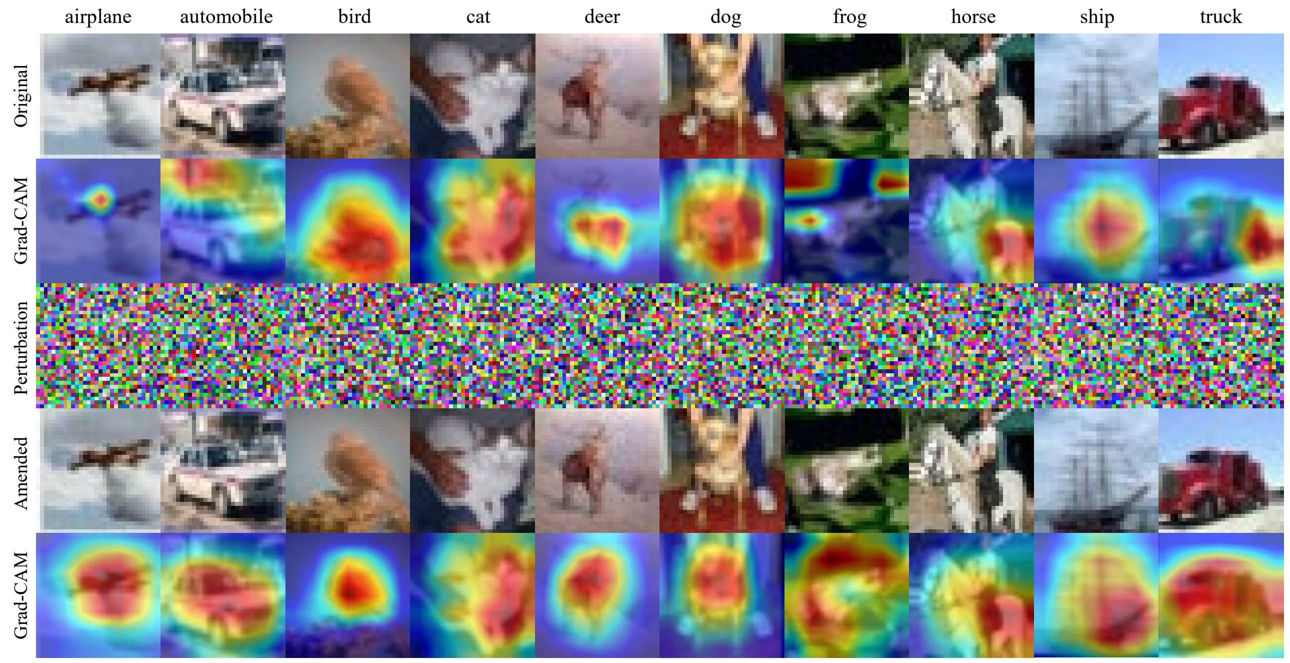

Figure 5. The visualization results on CIFAR-10 with Grad-CAM++. The first and fourth rows represent the original training samples and their amended version, respectively, and the second and fifth rows provide their Grad-CAM results with MobileNetV2. The third row shows the perturbations generated by our DPL.

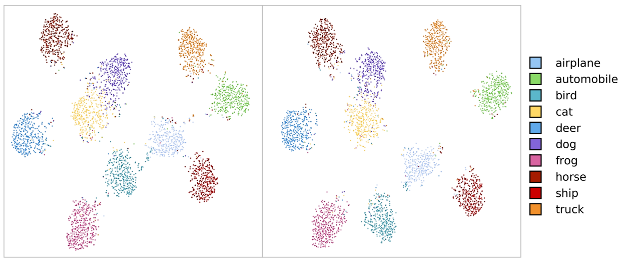

Figure 6. The t-SNE visualization of features extracted by MobileNetV2 on CIFAR-10. Left: features of training samples and test samples extracted by MobileNetV2. Right: features of amended training samples and test samples extracted by MobileNetV2 (DPL). DPL optimizes more discriminative features for those samples close to the class boundaries and leads to clearer class boundaries, especially between the classes of frogs, birds, airplanes, and automobiles.

to the baseline model, the test error rate and test loss are reduced by at most 0.31% and 0.20, respectively.

Replacement vs. Augmentation. In Fig. 2 (c), we compare two strategies (replacement and augmentation) in performance and observe that the performance of augmentation converges later than replacement. It is not surprising since the augmentation strategy uses more training data (original and amended images). In short, the replacement approach provides a quick performance improvement while the augmentation approach gradually improve the performance over iterations. In the performance experiments, we use the augmentation strategy to achieve better accuracy.

The Efficacy of Empirical Risk Minimization. As discussed in Sec. 3.2, we cannot minimize expected risk directly since the data distribution of test set is unknown. Instead, we propose DPL to minimize empirical risk. In Fig. 3, it is clear that the test loss (expected risk) changes towards the same trend as the training loss (empirical risk) within 3 DPL iterations. And as presented in Fig. 4, the class that get more reduction in the loss tend to have a greater decrease in the error rate. These phenomena verify that the further minimization of the empirical risk $R_{\text{emp}}(\theta)$ by DPL positively leads to a further minimization of the expected risk $R_{\text{exp}}(\theta)$, resulting in further reduction in the error rate. In addition, we notice that the trends of the expected risk and the empirical risk diverge when the number of DPL iterations exceed 3. This indicates the need to conduct appropriate DPL iterations to avoid overfitting.

Qualitative Evaluations. Class Activation Map (CAM) (Zhou et al., 2016) is a great tool to diagnose the models by visualizing the discriminative areas in an image. To compare the models with and without DPL training, we use Grad-CAM++ (Chattopadhay et al., 2018) for visualization on CIFAR-10. Fig. 5 presents the qualitative comparisons between the original samples and their amended version. As shown in rows 2 and 5, after applying our amendation to the

Table 6. Top-1 accuracy (%)↑ on image classification when combining DPL with existing data augmentations using ResNet-50.

Method	CIFAR-100	ImageNet
Vanilla Training	77.37	76.22
MixUp (Zhang et al., 2018)	78.36	77.33
+ DPL	78.86	77.51
CutMix (Yun et al., 2019)	78.57	78.46
+ DPL	79.08	78.57

samples, the network is able to more precisely focus on the salient areas of the images, implying a more discriminative model achieved by DPL.

To understand the efficacy of DPL in depth, we compare the features without and with DPL in the feature space. We use t-SNE (Linderman et al., 2017) to project high-dimensional features into 2D vectors. We compare the features of 2000 original and amended training samples, extracted by MobileNetV2 (Sandler et al., 2018) on CIFAR-10. In addition, 2000 randomly selected test samples are also added to show generalization capability. In Fig. 6, compared with the original training samples, the features of the amended training samples are better clustered with the test samples. The boundaries of DPL between different classes are clearer, especially between the classes of frogs, birds, airplanes and automobiles, verifying that the samples amended by DPL are more discriminative.

Combination with Data Augmentations. When applied with augmentation approach, perturbed samples optimized by DPL can be regarded as a kind of individual data augmentation from the perspective of increasing the size and diversity of the dataset using perturbed samples. In theory, as a generic strategy, DPL is compatible with existing training strategies (*e.g.*, data augmentation). To push the limit of our method, we further combine DPL with common data augmentation methods including Mixup (Zhang et al., 2018) and CutMix (Yun et al., 2019). As shown in Tab. 6, when combining with CutMix, our method beats the vanilla training (ResNet50) by 2.35% on ImageNet. These experimental results demonstrate that DPL can be applied with existing data augmentations simultaneously to furthering the network performance.

5. Conclusion

In this paper, we introduce new insights into utilizing image perturbations and propose Deep Perturbation Learning (DPL), to improve the performance of well-trained models. Unlike previous works focusing on adversarial examples, we formulate an optimization process of minimizing the empirical risk and construct a differentiable function w.r.t image perturbations. As a generic paradigm, DPL can be adopted to many downstream tasks with different backbone architectures. Extensive experiments demonstrate that the plug-and-play DPL can consistently furthering the network performance across multiple downstream tasks, achieving highly competitive performance compared to recent works. In the future, we will investigate the application of DPL to more tasks, including natural language processing tasks.

Acknowledgements

Z. Song and C. Zhao acknowledge that the work was supported by the National Natural Science Fund of China (62076184, 61976158, 61976160, 62076182, 62276190), in part by Fundamental Research Funds for the Central Universities and State Key Laboratory of Integrated Services Networks (Xidian University), in part by Shanghai Innovation Action Project of Science and Technology (20511100700) and Shanghai Natural Science Foundation (22ZR1466700).

References

Carlini, N. and Wagner, D. A. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, pp. 39–57, 2017.

Chattopadhay, A., Sarkar, A., Howlader, P., and Balasubramanian, V. N. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pp. 839–847. IEEE, 2018.

Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

Chen, X., Xie, C., Tan, M., Zhang, L., Hsieh, C.-J., and Gong, B. Robust and accurate object detection via adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16622–16631, 2021.

Cubuk, E. D., Zoph, B., Mané, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation strategies from data. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 113–123, 2019.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. Randaugment: Practical automated data augmentation with a reduced search space. In *Advances in neural information processing systems*, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Dong, Y., Deng, Z., Pang, T., Zhu, J., and Su, H. Adversarial distributional training for robust deep learning. *Advances in Neural Information Processing Systems*, 33: 8270–8283, 2020.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Ducoffe, M. and Precioso, F. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*, 2018.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88 (2):303–338, 2010.

Gelfand, I. M. and Fomin, S. V. *Calculus of Variations*. Dover Publications, 2000.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations*, 2015.

Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J., and Stahel, W. A. *Robust statistics: the approach based on influence functions*, volume 196. John Wiley & Sons, 2011.

Hariharan, B., Arbeláez, P., Bourdev, L., Maji, S., and Malik, J. Semantic contours from inverse detectors. In *2011 International Conference on Computer Vision*, pp. 991–998. IEEE, 2011.

Hataya, R., Zdenek, J., Yoshizoe, K., and Nakayama, H. Faster autoaugment: Learning augmentation strategies using backpropagation. In *16th European Conference on Computer Vision*, pp. 1–16. Springer, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016a.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016b.

Ho, D., Liang, E., Chen, X., Stoica, I., and Abbeel, P. Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning*, pp. 2731–2741, 2019.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pp. 1885–1894. PMLR, 2017.

Kong, Z. and Chaudhuri, K. Understanding instance-based interpretability of variational auto-encoders. *arXiv preprint arXiv:2105.14203*, 2021.

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

Lamb, A., Verma, V., Kannala, J., and Bengio, Y. Interpolated adversarial training: Achieving robust neural networks without sacrificing too much accuracy. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, pp. 95–103, 2019.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Li, S., Wu, D., Wu, F., Zang, Z., and Stan.Z.Li. Architecture-agnostic masked image modeling–from vit back to cnn. In *Proceedings of the 40th international conference on Machine learning (ICML)*, 2023.

Li, X., Zhong, Z., Wu, J., Yang, Y., Lin, Z., and Liu, H. Expectation-maximization attention networks for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9167–9176, 2019.

Lim, S., Kim, I., Kim, T., Kim, C., and Kim, S. Fast autoaugment. *Advances in Neural Information Processing Systems*, 32, 2019.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Linderman, G. C., Rachh, M., Hoskins, J. G., Steinerberger, S., and Kluger, Y. Efficient algorithms for t-distributed stochastic neighborhood embedding. *arXiv preprint arXiv:1712.09005*, 2017.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. Ssd: Single shot multibox detector. In *European conference on computer vision*, pp. 21–37. Springer, 2016.

Liu, Y., Chen, X., Cheng, M., Hsieh, C.-J., and You, Y. Concurrent adversarial learning for large-batch training. In *ICLR*, 2022.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Mei, J., Han, Y., Bai, Y., Zhang, Y., Li, Y., Li, X., Yuille, A., and Xie, C. Fast advprop. In *ICLR*, 2022.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. 2017.

Pearlmutter, B. A. Fast exact multiplication by the hessian. *Neural computation*, 6(1):147–160, 1994.

Pruthi, G., Liu, F., Sundararajan, M., and Kale, S. Estimating training data influence by tracing gradient descent. *arXiv preprint arXiv:2002.08484*, 2020.

Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.

Shafahi, A., Najibi, M., Ghiasi, M. A., Xu, Z., Dickerson, J., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019.

Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., and Webb, R. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2107–2116, 2017.

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

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Tan, M. and Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pp. 6105–6114. PMLR, 2019.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In *7th International Conference on Learning Representations*, 2019.

Xie, C., Tan, M., Gong, B., Wang, J., Yuille, A. L., and Le, Q. V. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 819–828, 2020.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Zhang, C., Benz, P., Imtiaz, T., and Kweon, I. S. Understanding adversarial examples from the mutual influence of images and perturbations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14521–14530, 2020.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

Zhang, H., Zhang, H., Wang, C., and Xie, J. Co-occurrent features in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 548–557, 2019a.

Zhang, X., Wang, Q., Zhang, J., and Zhong, Z. Adversarial autoaugment. In *International Conference on Learning Representations*, 2019b.

Zhao, Z., Liu, Z., and Larson, M. Towards large yet imperceptible adversarial image perturbations with perceptual color distance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1039–1048, 2020.

Zheng, Y., Zhang, Z., Yan, S., and Zhang, M. Deep autoaugment. *arXiv preprint arXiv:2203.06172*, 2022.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929, 2016.

Zoph, B., Cubuk, E. D., Ghiasi, G., Lin, T.-Y., Shlens, J., and Le, Q. V. Learning data augmentation strategies for object detection. In *European conference on computer vision*, pp. 566–583. Springer, 2020.

A. Derivation of the Formulas

For completeness, we provide a detailed but not fully rigorous derivation of the estimation for $R_{\text{emp}}(\hat{\theta}_{D^\delta}; D) - R_{\text{emp}}(\hat{\theta}; D)$. Following Sec. 2.3, our derivation considers two scenarios: replacement and augmentation.

A.1. Replacement

Recall that $\hat{\theta}$ minimizes the empirical risk:

$$R_{\text{emp}}(\theta; D) \triangleq \frac{1}{N}\sum_{i=1}^{N} L(z_i; \theta). \tag{13}$$

Due to the assumptions that the empirical risk $R_{\text{emp}}(\theta; D)$ is twice-differentiable and strongly convex in θ,

$$H_{\hat{\theta}} \triangleq \nabla_\theta^2 R_{\text{emp}}(\hat{\theta}; D) = \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta^2 L(z_i; \hat{\theta}) \tag{14}$$

exists and is positive definite, which guarantees the existence of $H_{\hat{\theta}}^{-1}$.

Similarly, the perturbed weights $\hat{\theta}_{\varepsilon, D^\delta}$ can be written as

$$\begin{aligned}\hat{\theta}_{\varepsilon, D^\delta} \triangleq \arg\min_\theta [\frac{1}{N}\sum_{i=1}^{N} L(z_i; \theta) + \frac{\varepsilon}{n}\sum_{i=1}^{n} L(z_i^{\delta_i}; \theta) \\ - \frac{\varepsilon}{n}\sum_{i=1}^{n} L(z_i; \theta)],\end{aligned} \tag{15}$$

Since $\hat{\theta}_{\varepsilon, D^\delta}$ is a minimizer of Eq. (15), we have:

$$\begin{aligned}0 &= \nabla_\theta [R_{\text{emp}}(\theta; D) + \frac{\varepsilon}{n}\sum_{i=1}^{n} (L(z_i^{\delta_i}; \theta) - L(z_i; \theta))]|_{\theta = \hat{\theta}_{\varepsilon, D^\delta}} \\ &= \nabla_\theta R_{\text{emp}}(\hat{\theta}_{\varepsilon, D^\delta}; D) \\ &\quad + \varepsilon \cdot \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta [L(z_i^{\delta_i}; \hat{\theta}_{\varepsilon, D^\delta}) - L(z_i; \hat{\theta}_{\varepsilon, D^\delta})].\end{aligned} \tag{16}$$

Since $\hat{\theta}_{\varepsilon, D^\delta} \to \hat{\theta}$ as $\varepsilon \to 0$, we perform a Taylor expansion for the right-hand side of the above equation:

$$\begin{aligned}0 &= \nabla_\theta R_{\text{emp}}(\hat{\theta}; D) + \varepsilon \cdot \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta [L(z_i^{\delta_i}; \hat{\theta}) - L(z_i; \hat{\theta})] \\ &\quad + [\nabla_\theta^2 R_{\text{emp}}(\hat{\theta}; D) + \varepsilon \cdot V]\Delta_\varepsilon + o(\|\Delta_\varepsilon\|),\end{aligned} \tag{17}$$

where $\Delta_\varepsilon \triangleq \hat{\theta}_{\varepsilon, D^\delta} - \hat{\theta} \to 0$ as $\varepsilon \to 0$ and

$$V \triangleq \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta^2 [L(z_i^{\delta_i}; \hat{\theta}) - L(z_i; \hat{\theta})], \tag{18}$$

for the sake of simplicity.

Dropping the higher order term $o(\|\Delta_\varepsilon\|)$ and solving Δ_ε, we get:

$$\begin{aligned}\Delta_\varepsilon &\approx -[\nabla_\theta^2 R_{\text{emp}}(\hat{\theta}; D) + \varepsilon \cdot V]^{-1} \\ &\quad \cdot \{\nabla_\theta R_{\text{emp}}(\hat{\theta}; D) + \varepsilon \cdot \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta [L(z_i^{\delta_i}; \hat{\theta}) - L(z_i; \hat{\theta})]\}.\end{aligned} \tag{19}$$

Since $\hat{\theta}$ minimizes $R_{\text{emp}}(\theta; D)$, we have $\nabla_\theta R_{\text{emp}}(\hat{\theta}; D) = 0$. Then combining Eq. (14), we can derive that

$$\begin{aligned}\Delta_\varepsilon &\approx -(H_{\hat{\theta}} + \varepsilon \cdot V)^{-1} \cdot \varepsilon \cdot \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta [L(z_i^{\delta_i}; \hat{\theta}) - L(z_i; \hat{\theta})] \\ &= -(H_{\hat{\theta}}^{-1} + O(\varepsilon)) \cdot \varepsilon \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta [L(z_i^{\delta_i}; \hat{\theta}) - L(z_i; \hat{\theta})] \\ &= -\varepsilon H_{\hat{\theta}}^{-1} \cdot \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta [L(z_i^{\delta_i}; \hat{\theta}) - L(z_i; \hat{\theta})] + o(\varepsilon).\end{aligned}$$

Taking derivatives of both sides, we obtain

$$\begin{aligned}\frac{d\hat{\theta}_{\varepsilon, D^\delta}}{d\varepsilon}|_{\varepsilon=0} &= \frac{d\Delta_\varepsilon}{d\varepsilon}|_{\varepsilon=0} \\ &= -\frac{1}{n} H_{\hat{\theta}}^{-1} [\sum_{i=1}^{n} (\nabla_\theta L(z_i^\delta, \hat{\theta}) - \nabla_\theta L(z_i, \hat{\theta}))].\end{aligned}$$

Then we can estimate $R_{\text{emp}}(\hat{\theta}_{D^\delta}; D) - R_{\text{emp}}(\hat{\theta}; D)$.

Note that $R_{\text{emp}}(\hat{\theta}_{\varepsilon, D^\delta}; D)$ is a differentiable function w.r.t. ε, satisfying $R_{\text{emp}}(\hat{\theta}_{0, D^\delta}; D) = R_{\text{emp}}(\hat{\theta}; D)$ and $R_{\text{emp}}(\hat{\theta}_{\frac{n}{N}, D^\delta}; D) = R_{\text{emp}}(\hat{\theta}_{D^\delta}; D)$. By the Taylor expansion of $R_{\text{emp}}(\hat{\theta}_{\varepsilon, D^\delta}; D)$ at $\varepsilon = 0$, we have

$$\begin{aligned}&R_{\text{emp}}(\hat{\theta}_{D^\delta}; D) - R_{\text{emp}}(\hat{\theta}; D) \\ &\approx \frac{n}{N}\frac{\partial R_{\text{emp}}(\hat{\theta}_{\varepsilon, D^\delta}; D)}{\partial \varepsilon}|_{\varepsilon=0} \\ &= \frac{n}{N}\frac{\partial R_{\text{emp}}(\hat{\theta}; D)}{\partial \theta}\frac{d\hat{\theta}_{\varepsilon, D^\delta}}{d\varepsilon}|_{\varepsilon=0} \\ &= -\frac{1}{N}\frac{\partial R_{\text{emp}}(\hat{\theta}; D)}{\partial \theta} H_{\hat{\theta}}^{-1} [\sum_{i=1}^{n} (\nabla_\theta L(z_i^\delta, \hat{\theta}) - \nabla_\theta L(z_i, \hat{\theta}))] \\ &\approx -\frac{1}{N}\sum_{i=1}^{n} \frac{\partial R_{\text{emp}}(\hat{\theta}; D)}{\partial \theta} H_{\hat{\theta}}^{-1} [\nabla_x \nabla_\theta L(z_i, \hat{\theta})\delta_i].\end{aligned} \tag{20}$$

In the last step of the above derivation, we have used a Taylor expansion of $\nabla_\theta L(z_i^\delta, \hat{\theta})$ at $\delta = 0$, *i.e.*,

$$\nabla_\theta L(z_i^\delta, \hat{\theta}) - \nabla_\theta L(z_i, \hat{\theta}) \approx \nabla_x \nabla_\theta L(z_i, \hat{\theta}) \cdot \delta_i. \tag{21}$$

In conclusion, we have the estimation

$$\begin{aligned}&R_{\text{emp}}(\hat{\theta}_{D^\delta}; D) - R_{\text{emp}}(\hat{\theta}; D) \\ &\approx -\frac{1}{N}\sum_{i=1}^{n} [\frac{\partial R_{\text{emp}}(\hat{\theta}; D)}{\partial \theta} \cdot H_{\hat{\theta}}^{-1} \cdot \nabla_x \nabla_\theta L(z_i, \hat{\theta})] \cdot \delta_i.\end{aligned}$$

And here we present the pseudo-code of replacement for DPL in Algorithm 2.

Algorithm 2 DPL (Replacement)

```
Require: $D_0 = \{(x_i^0, y_i)\}_{i=1}^N$, $\theta_0$, $n$, $\alpha$, $f_\theta$
Ensure: $\theta_T$
1: for t=0,1,...,T-1 do
2:     train $f_\theta$ on $D_t$ with $\theta_t$ as its initialization, output $\theta_{t+1}$
3:     compute
       $$\Delta_i \triangleq \alpha \| \frac{\partial R_{\text{emp}}(\theta_{t+1})}{\partial \theta} H_{\theta_{t+1}}^{-1} \nabla_x \nabla_\theta L(z_i, \theta_{t+1}) \|_2$$
4:     select $n$ samples with the largest $\Delta_i$
5:     $x_i^{t+1} = x_i^t + \alpha \cdot \frac{\partial R_{\text{emp}}(\theta_{t+1})}{\partial \theta} H_{\theta_{t+1}}^{-1} \nabla_x \nabla_\theta L(z_i, \theta_{t+1})$
6:     let $D_{t+1} = \{(x_i^{t+1}, y_i)\}_{i=1}^n \cup \{(x_i^t, y_i)\}_{i=n+1}^N$
7: end for
```

A.2. Augmentation

Since the key derivation of the augmentation case is the same as the replacement case, we focus on discussing their difference. Similar as Eq. (15), the differentiable function is defined by:

$$\hat{\theta}_{\varepsilon, D^\delta} \triangleq \arg\min_\theta [\frac{1}{N} \sum_{i=1}^N L(z_i; \theta) + \frac{\varepsilon}{n} \sum_{i=1}^n L(z_i^{\delta_i}; \theta)]. \quad (22)$$

The main difference between the two cases is the term $\frac{1}{n} \sum_{i=1}^n L(z_i^{\delta_i}; \theta)$ in Eq. (22) and $\frac{1}{n} \sum_{i=1}^n \nabla_\theta [L(z_i^{\delta_i}; \theta) - L(z_i; \theta)]$ in Eq. (15). Treating this term as a symbol and following the same derivation, we have

$$\frac{d\hat{\theta}_{\varepsilon, D^\delta}}{d\varepsilon}|_{\varepsilon=0} = -\frac{1}{n} H_{\hat{\theta}}^{-1} \sum_{i=1}^n \nabla_\theta L(z_i^\delta, \hat{\theta}).$$

In this case, with Eq. (21), we can obtain

$$\begin{aligned}
& R_{\text{emp}}(\hat{\theta}_{D^\delta}; D) - R_{\text{emp}}(\hat{\theta}; D) \\
\approx & \frac{n}{N} \frac{\partial R_{\text{emp}}(\hat{\theta}_{\varepsilon, D^\delta}; D)}{\partial \varepsilon}|_{\varepsilon=0} \\
= & \frac{n}{N} \frac{\partial R_{\text{emp}}(\hat{\theta}; D)}{\partial \theta} \frac{d\hat{\theta}_{\varepsilon, D^\delta}}{d\varepsilon}|_{\varepsilon=0} \\
= & -\frac{1}{N} \frac{\partial R_{\text{emp}}(\hat{\theta}; D)}{\partial \theta} H_{\hat{\theta}}^{-1} \sum_{i=1}^n \nabla_\theta L(z_i^\delta, \hat{\theta}) \\
\approx & -\frac{1}{N} \sum_{i=1}^n \frac{\partial R_{\text{emp}}(\hat{\theta}; D)}{\partial \theta} H_{\hat{\theta}}^{-1} [\nabla_\theta L(z_i, \hat{\theta}) + \nabla_x \nabla_\theta L(z_i, \hat{\theta}) \delta_i] \\
= & -\frac{1}{N} \sum_{i=1}^n [\frac{\partial R_{\text{emp}}(\hat{\theta}; D)}{\partial \theta} H_{\hat{\theta}}^{-1} \nabla_\theta L(z_i, \hat{\theta}) \\
& + \frac{\partial R_{\text{emp}}(\hat{\theta}; D)}{\partial \theta} H_{\hat{\theta}}^{-1} \nabla_x \nabla_\theta L(z_i, \hat{\theta}) \cdot \delta_i]. \quad (23)
\end{aligned}$$

B. Relaxing the Assumptions of Empirical Risk

Recall that our estimations are asymptotic approximations under the assumptions that (i) the model weights $\hat{\theta}$ minimizes the empirical risk, and (ii) the empirical risk is twice-differentiable and strictly convex in θ. Here, we give some discussions on the following two aspects:
(a) non-differentiable losses;
(b) non-convexity and non-convergence;
to relax the assumptions. Our analysis indicates that our DPL is still effective even though these assumptions (i) and (ii) are violated.

B.1. Non-differentiable Losses

Our DPL is a generic paradigm. In theory, it can further reduce empirical risk and be adopted to any downstream task. However, some of the commonly used loss functions are non-differentiable, resulting in the failure of our DPL. Here, we propose an empirical approach to overcome this difficulty. Specifically, we take hinge loss $\text{Hinge}(s) = \max(0, 1 - s)$ as an example.

For hinge loss, since $\text{Hinge}(s)$ is not differentiable at $s = 1$, the derivatives of the loss in the perturbation formula (*i.e.*, Eq. (9) in Sec. 2.3) is not always available. To calculate the perturbation of training sample, we need to approximate $\text{Hinge}(s)$ with $\text{SmoothHinge}(s; t) = t \log(1 + \exp(\frac{1-s}{t}))$, which approaches the hinge loss as $t \to 0$. As long as t is small enough, the difference between $\text{SmoothHinge}(s; t)$ and $\text{Hinge}(s)$ can be ignored. Thus using $\text{SmoothHinge}(s; t)$ to generate the perturbations of training samples can reduce the empirical risk based on both $\text{SmoothHinge}(s; t)$ and $\text{Hinge}(s)$.

In addition, since smooth function has good approximation property, theoretically , for any non-differentiable loss function, we can always find a smooth function sufficiently close to it. Thus, our DPL can be theoretically effective for any loss functions.

B.2. Non-convexity and Non-convergence

Nowadays, deep learning is widely used in various computer vision tasks. As is well known, deep neural networks with widely used loss functions lead to non-convex empirical risk (*i.e.*, non-convexity) and practical optimization like running SGD cannot achieve strict local minima (*i.e.*, non-convergence). Thus, generally, after weight stage, the model weights we obtain which is denoted by $\tilde{\theta}$ is close but not equal to a local minima $\hat{\theta}$, *i.e.*, $\tilde{\theta} \neq \hat{\theta}$. As a result, $\nabla_\theta R_{\text{emp}}(\tilde{\theta} : D) \neq 0$ and $H_{\tilde{\theta}}$ could have negative eigenvalues.

To attack this issue, our approach is to form a convex quadratic approximation of the loss around $\tilde{\theta}$, *i.e.*, for a training sample z,

$$\tilde{L}(z;\theta) \triangleq L(z;\tilde{\theta}) + \nabla_\theta L(z;\tilde{\theta})^T \cdot (\theta - \tilde{\theta}) + \frac{1}{2}(\theta - \tilde{\theta})^T(\nabla^2_\theta L(z;\tilde{\theta}) + \lambda I)(\theta - \tilde{\theta}), \quad (24)$$

where λ is a damping term. It is easy to verify that $\tilde{L}(z;\theta)$ satisfies:
(1) $\tilde{L}(z;\tilde{\theta}) = L(z;\tilde{\theta})$;
(2) $\nabla_\theta \tilde{L}(z;\tilde{\theta}) = \nabla_\theta L(z;\tilde{\theta})$;
(3) $\nabla^2_\theta \tilde{L}(z;\tilde{\theta}) = \nabla^2_\theta L(z;\tilde{\theta}) + \lambda I$;
Then on the entire training set, we have

$$\nabla^2_\theta R_{\text{emp}}(\tilde{\theta}; D) = H_{\tilde{\theta}} + \lambda I. \quad (25)$$

We then calculate the perturbation of training samples using $\tilde{L}(z;\theta)$. By adjusting λ, we can ensure that $\nabla^2_\theta R_{\text{emp}}(\tilde{\theta}; D)$ is positive definite because of Eq. (25). Thus we can compute the inverse hessian term in our perturbation formula. In addition, by (1) and (2), the $\tilde{L}(z;\theta)$ can maintain the derivatives of the loss below the second order in the perturbation formula.

C. Dataset Details

Image Classification. The CIFAR-10 (Krizhevsky et al., 2009) and CIFAR-100 (Krizhevsky et al., 2009) datasets consist of 60,000 images with 10 and 100 categories, respectively. The training set includes 50,000 images and the test set includes 10,000 images. The ImageNet-1k (Deng et al., 2009) dataset contains 1,281,167 training samples and 50,000 validation samples of 1000 classes.

Object Detection. PASCAL VOC (Everingham et al., 2010) consists of VOC 2007 and VOC 2012 versions, containing 9963 images with 24640 labeled objects and 11540 images with 27450 labeled objects, respectively. We train our models on the merged trainval set of VOC 2007 and VOC 2012, and then test on the test set of VOC 2007. The COCO 2017 object detection (Lin et al., 2014) contains 118K training images and 5K validation.

Semantic Segmentation. The PASCAL VOC 2012 (Everingham et al., 2010) consists of 1464 (train), 1449 (val), and 1456 (test) images. The SBD (Hariharan et al., 2011) dataset contains 11355 images taken from the PASCAL VOC 2011 dataset with different train and val splits. Both the PASCAL VOC and SBD datasets include segmentations and boundaries for 20 object categories. We use the official training and test splits in our experiments. For the ImageNet and COCO datasets, we conduct experiments on 4 Tesla V100S. The experiments on the other datasets are performed on 2 RTX 3090.

Table 7. Top-1 test accuracy (%) with variance on CIFAR-10/100 and ImageNet for image classification.

Dataset	Model	Baseline	**DPL**
CIFAR-10	VGG-16	92.76	**93.29** ± 0.12
	ResNet-18	94.65	**95.52** ± 0.15
	ResNet-50	94.92	**95.90** ± 0.08
	PreActResNet-18	94.87	**95.48** ± 0.17
	DenseNet-121	94.84	**95.27** ± 0.14
	MobileNetV2-1.0	94.33	**94.77** ± 0.15
	ViT-B	98.47	**98.69** ± 0.17
CIFAR-100	VGG-16	70.36	**72.37** ± 0.06
	ResNet-18	75.53	**76.32** ± 0.13
	ResNet-50	77.37	**78.74** ± 0.22
	PreActResNet-18	72.91	**73.86** ± 0.19
	MobileNetV2-1.0	67.75	**68.88** ± 0.25
ImageNet	ResNet-18	69.77	**70.15** ± 0.11
	ResNet-50	76.22	**76.58** ± 0.07
	MobileNetV2-0.5	64.47	**65.18** ± 0.10
	MobileNetV2-0.25	52.25	**52.59** ± 0.09

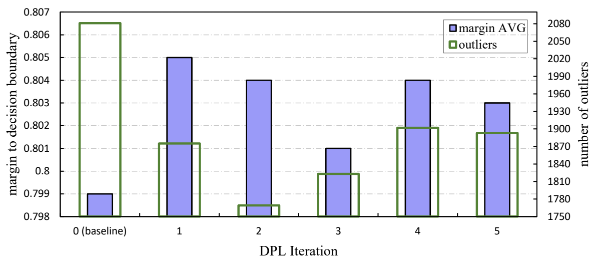

Figure 7. The changes in the margins to decision boundary and the number of outliers brought by DPL.

D. Additional Experimental Results

Empirical Results with variances. In the main work, we report the experimental results of DPL averaged over four independent runs. To further verify the efficacy of the proposed method, we supplement the performance variances on image classification in Tab. 7.

DPL Copes Well with Hard Samples. Following (Ducoffe & Precioso, 2018), we calculate the margin to decision boundary for each training sample and report the changes after DPL iterations in Fig. 7. The purple bars represent the average margin value of all training samples to the decision boundary, while the green bars record the number of samples with outlier margin values (close to the decision boundary).

Table 8. Time cost of DPL (1 iter.) on CIFAR-10/100, ImageNet, PASCAL VOC, and COCO in GPU hours, estimated on NVIDIA Tesla V100S.

CIFAR-10	CIFAR-100	ImageNet	PASCAL	COCO
4	6	17	8	15

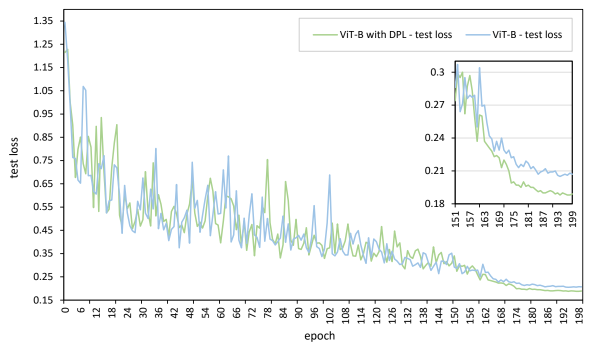

Figure 8. Comparison of the test loss decline on CIFAR-10.

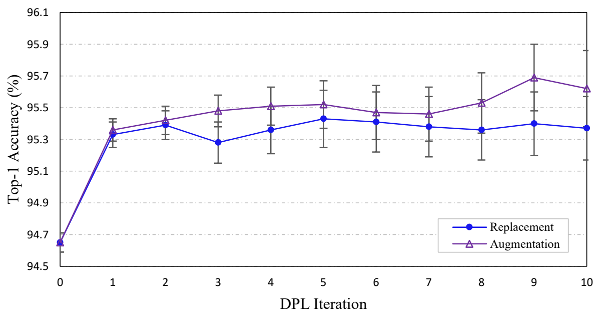

Figure 9. Deeper ablation study of DPL iteration on CIFAR-10 using ResNet-18. We report the average results with error bars of 3 independent runs.

As the iteration progresses, the mean value of the margins to decision boundary shows an increasing trend, while the number of outliers decreases. It indicates that models with DPL can obtain better ability to discriminate hard samples and achieve enhanced performance.

Details on the Test Loss Decline During Training. According to Fig. 8, DPL helps Vit-B converge to a lower test loss, and this improvement is especially significant in the late training period (after epoch 150). It confirms that the performance improvement from our proposed DPL is solid and have room for further improvement.

Deeper Ablation Study of DPL Iteration. In the main work, we report the results within 5 DPL iterations in the performance experiments and ablation study of DPL Iteration. Despite the expensive experiment costs, we wonder what deeper DPL iterations would bring. As shown in Fig. 9, it takes 9 DPL iterations to further achieve 1.04% improvement in top-1 accuracy for ResNet-18 on CIFAR-10. In addition, we notice that the augmentation strategy outperforms the replacement strategy in deeper iterations.

Perturbation Optimization Cost. In Tab. 8, we record the time cost of perturbation optimization progress of our DPL. Compared to the hundreds of GPU hours required by the data augmentation search policies (Zoph et al., 2020; Lim et al., 2019; Zheng et al., 2022) and adversarial training methods (Xie et al., 2020; Chen et al., 2021; Lamb et al., 2019; Mei et al., 2022), DPL can provide significant improvement for the network performance within an acceptable time consumption.

E. Limitations and Future Work

As illustrated in the main work, DPL requires multiple rounds of optimization with Hessian-vector product computing to achieve further enhanced performance for the networks. To speed up the DPL optimization, we have formulated an efficient approximation of perturbation stage in Sec. 3.3 and achieved acceptable time cost as reported in Tab. 8. Nevertheless, subject to the costly calculation of the Hessian matrix and the retraining process of multi-rounds DPL, it is still expensive to pursue the ultimate performance improvement brought by DPL (*e.g.*, DPL with 9 iterations further boosts ResNet-18 to 1.04% improvement in top-1 accuracy on CIFAR-10 as shown in Fig. 9). Additionally, in theory, the perturbation stage formulated in this work is generic and we will explore the potential of DPL in transfer learning and out-of-distribution detection in the future.