SELF-ENSEMBLE PROTECTION: TRAINING CHECKPOINTS ARE GOOD DATA PROTECTORS

Anonymous authors
Paper under double-blind review

ABSTRACT

As data become increasingly vital for deep learning, a company would be very cautious about releasing data, because the competitors could use the released data to train high-performance models, thereby posing a tremendous threat to the company's commercial competence. To prevent training good models on the data, imperceptible perturbations could be added to it. Since such perturbations aim at hurting the entire training process, they should reflect the vulnerability of DNN training, rather than that of a single model. Based on this new idea, we seek adversarial examples that are always unrecognized (never correctly classified) in training. In this paper, we uncover them by modeling checkpoints' gradients, forming the proposed self-ensemble protection (SEP), which is very effective because (1) learning on examples ignored during normal training tends to yield DNNs ignoring normal examples; (2) checkpoints' cross-model gradients are close to orthogonal, meaning that they are as diverse as DNNs with different architectures in conventional ensemble. That is, our amazing performance of ensemble only requires the computation of training one model. By extensive experiments with 9 baselines on 3 datasets and 5 architectures, SEP is verified to be a new state-of-the-art, e.g., our small $\ell_\infty = 2/255$ perturbations reduce the accuracy of a CIFAR-10 ResNet18 from 94.56% to 14.68%, compared to 41.35% by the best-known method.

1 INTRODUCTION

Large-scale datasets have become increasingly important in training high-performance deep neural networks (DNNs). Thus, it is a common practice to collect data online (Mahajan et al., 2018; Sun et al., 2017), an almost unlimited data source. This poses a great threat to the commercial competence of data owners such as social media companies since the competitors could also train good DNNs from their data. Therefore, great efforts have been devoted to protecting data from unauthorized use in model training. The most typical way is to add imperceptible perturbations to the data, so that DNNs trained on it have poor generalization (Huang et al., 2020a; Fowl et al., 2021b).

Existing data protection methods use a single DNN to generate incorrect but DNN-sensitive features (Huang et al., 2020a; Fu et al., 2021; Fowl et al., 2021b) for training data by, *e.g.*, adversarial attacks (Goodfellow et al., 2015). **[R1.3, R2.3]** However, the data protectors cannot know what DNN and what training strategies the unauthorized users will adopt. Thus, the protective examples should aim at **hurting the DNN training, a whole dynamic process, instead of a static DNN**. Therefore, it would be interesting to study the *vulnerability of DNN training*. Recall that the vulnerability of a DNN is revealed by the adversarial examples which are similar to clean ones but unrecognized by the model (Madry et al., 2018). Similarly, we depict the vulnerability of training by perturbed training data that are never predicted correctly during training. Learning on examples ignored during normal training tends to yield DNNs ignoring normal examples.

Such examples could be easily uncovered by the gradients from the ensemble of model training checkpoints. However, ensemble methods have never been explored in data protection to the best of our knowledge, so it is natural to wonder

*Can we use these intermediate checkpoint models for data protection in a **self-ensemble** manner?*

An effective ensemble demands high diversity of sub-models, which is generally quantified by their gradient similarity (Pang et al., 2019; Yang et al., 2021), *i.e.*, the gradients on the same image from different sub-models should be orthogonal. Surprisingly, we found that checkpoints' gradients are as orthogonal as DNNs with different architectures in the conventional ensemble. In this regard, we argue that intermediate checkpoints are very diverse to form the proposed self-ensemble protection (SEP), challenging existing beliefs on their similarity (Li et al., 2022).

By SEP, effective ensemble protection is achieved by the computation of training only one DNN. Since the scale of data worth protected is mostly very large, SEP avoids tremendous costs by training many models individually in conventional ensemble. Therefore, our study enables practical ensemble for large-scale data, which may help improve the generalization, increase the attack transferability, and study DNN training dynamics.

[R1.1] Multiple checkpoints offer us a pool of good features for an input. Thus, we could additionally take the advantage of diverse features besides diverse gradients at no cost. Inspired by neural collapse theory (Papyan et al., 2020), which demonstrates that the mean feature of samples in a class is a highly representative depiction of this class, we bring about a novel feature alignment loss that induces a sample's last-layer feature collapse into the mean of incorrect-class features. With features from multiple checkpoints, FA robustly injects incorrect features so that DNNs are deeply confounded.

Equipping SEP with FA, our method achieves astonishing performance by revealing the vulnerability of DNN training: our examples are mostly mis-classified in any training processes compared to a recent method (Sandoval-Segura et al., 2022), and clean samples are always much closer to each other than to protected samples, indicating that the latter belong to another distribution that could not be noticed by normal training. By setting $\ell_\infty = 2/255$, a very small bound, SEP perturbations on the CIFAR-10 training set reduce the testing accuracy of a ResNet18 from 94.56% to 14.68%, while the best-known results could only reach 41.35% with the same amount of overall calculation to craft the perturbations. The superiority of our method is also observable in the study on CIFAR-100 and ImageNet subset on 5 architectures. **[R2.1]** We also study perturbations under different norms, and found that mixing ℓ_∞ and ℓ_0 perturbations (Wu et al., 2022) is the only effective way to resist ℓ_∞ adversarial training, which could recover the accuracy for all other types of perturbations. Our contributions could be summarized as below.

- We propose that protective perturbations should reveal the vulnerability of DNN training process, which we depict by the examples never classified correctly in training.
- We uncover such examples by the self-ensemble of model checkpoints, which are found to be surprisingly diverse as data protectors (SEP).
- SEP is very effective even when using the training computation of one DNN. Equipped with a novel feature alignment loss, our $\ell_\infty = 8/255$ perturbations lead DNNs to have < 5.7% / 3.2% / 0.6% accuracy on CIFAR-10 / CIFAR-100 / ImageNet subset.

2 Related Work

Small perturbations are known to be able to fool DNNs into incorrect predictions (Szegedy et al., 2014). Such test-time adversarial perturbations are crafted effectively by adversarially updating samples with model gradients (Carlini & Wagner, 2017), and the produced adversarial examples (AEs) transfer to hurt other DNNs as well (Chen et al., 2022). Similarly, training-time adversarial perturbations, *i.e.*, poisoning examples, are also obtainable by adversarially modify training samples using DNN gradients (Koh & Liang, 2017; Fowl et al., 2021b). All DNNs trained on poisoning examples generalize poorly on clean examples, making poisoning methods helpful in protecting data from unauthorized use of training. Besides adversarial noise, it has been demonstrated that error-minimization (Huang et al., 2020a), gradient alignment (Fowl et al., 2021a) and influence functions (Fang et al., 2020) are also useful in protecting data. However, current methods only use one DNN because the scale of data worth protection is very large for training multiple models.

Ensemble is validated as a panacea for boosting adversarial attacks (Liu et al., 2017; Dong et al., 2018). By aggregating the probabilities (Liu et al., 2017), logits or losses (Dong et al., 2018) of multiple models, ensemble attacks significantly increase the black-box attack success rate. Ensemble attacks could be further enhanced by reducing the gradient variance of sub-models (Xiong et al., 2022), and such an optimization way is also adopted in our method. Besides, ensemble has also been shown effective as a defense method by inducing low diversity across sub-models (Pang et al., 2019; Yang et al., 2020; 2021) or producing diverse AEs in adversarial training (Tramèr et al., 2018; Wang & Wang, 2021). Despite the good performance of ensemble in attacks and defenses, they have not been introduced to protect datasets due to its inefficiency. In this regard, we adopt the self-ensemble strategy, which only requires the computation of training one DNN. Its current applications are focused on semi-supervised learning (Zhao et al., 2019; Liu et al., 2022).

Two similar but different tasks besides poisoning-based data protection are adversarial training and backdoor attacks. Adversarial training (Madry et al., 2018; Zhang et al., 2019; Stutz et al., 2020) continuously generates AEs with current checkpoint gradients to improve the model's robustness towards worst-case perturbations. In contrast, data protection produces fixed poisoning examples so that unauthorized training yields low clean accuracy. Backdoor attacks (Geiping et al., 2020; Huang et al., 2020b) perturb a small proportion of training set to make the DNNs mispredict certain samples, but remain well-functional on other clean samples. While data protectors perturb the whole training set to degrade the model's performance on all clean samples.

3 THE PROPOSED METHOD

3.1 PRELIMINARIES

We first introduce the threat model and problem formulation of the data protection task. The data owner company wishes to release data for users while preventing an unauthorized *appropriator* to collect data for training DNNs. Thus, the data *protector* would add imperceptible perturbations to samples, so that humans have no obstacle to seeing the data, while the appropriator cannot train DNNs to achieve an acceptable testing accuracy. Since the protector has access to the whole training set, it can craft perturbations for each sample for effective protection (Shen et al., 2019; Feng et al., 2019; Huang et al., 2020a; Fowl et al., 2021b). Mathematically, the problem can be formulated as

$$\max_{\boldsymbol{\delta}\in\Pi_\varepsilon} \sum_{(\boldsymbol{x},y)\in\mathcal{D}} \mathcal{L}_{\text{CE}}(f_{\text{a}}(\boldsymbol{x},\theta^*),y), \quad \text{s.t. } \theta^* \in \arg\min_{\theta} \sum_{(\boldsymbol{x},y)\in\mathcal{T}} \mathcal{L}_{\text{CE}}\left(f_{\text{a}}\left(\boldsymbol{x}+\boldsymbol{\delta},\theta\right),y\right), \tag{1}$$

where the perturbations $\boldsymbol{\delta}$ bounded by ε are added to the training set $\mathcal{T}$ so that an appropriator model trained on it $f_{\text{a}}(\cdot,\theta^*)$ have a low accuracy on the test set $\mathcal{D}$, *i.e.*, a high cross-entropy loss $\mathcal{L}_{\text{CE}}(\cdot,\cdot)$. The $\boldsymbol{\delta}$ could be effectively calculated by targeted attacks (Fowl et al., 2021b), which use a well-trained protecting DNN f_{p} to produce targeted adversarial examples (AEs) that have the non-robust features in the incorrect class $g(y)$ as

$$\boldsymbol{x}_{t+1} = \Pi_\epsilon\left(\boldsymbol{x}_t - \alpha\cdot\text{sign}\left(\boldsymbol{G}_{\text{CE}}(f_{\text{p}},\boldsymbol{x}_t)\right)\right), \quad \boldsymbol{G}_{\text{CE}}(f_{\text{p}},\boldsymbol{x}) = \nabla_{\boldsymbol{x}}\mathcal{L}_{\text{CE}}\left(f_{\text{p}}\left(\boldsymbol{x}\right),g(y)\right), \tag{2}$$

where Π_ϵ clips the sample into the ε ℓ_∞-norm bound after each update with a step size α. $g(\cdot)$ stands for a permutation on the label space. Here we also adopt the optimization in (2).

3.2 DEPICTING THE VULNERABILITY OF DNN TRAINING

Current methods (Shen et al., 2019; Feng et al., 2019; Huang et al., 2020a; Fowl et al., 2021b) craft protective perturbations that are supposed to generalize to poison different unknown architectures by a single DNN f_{p}. **[R1.3, R2.3, R3.4]** However, the data protectors cannot know what DNN and what training strategies the unauthorized users will adopt. Thus, the protective examples should aim at hurting the DNN training, a whole dynamic process, instead of a static DNN. Therefore, it would be interesting to study the vulnerability of DNN training.

Recall that the vulnerability of a DNN is represented by AEs (Madry et al., 2018), because they are slightly different from clean testing samples but are totally unrecognizable by a static model. Similarly, the vulnerability of DNN training could be depicted by examples that are slightly different from clean training samples but are always unrecognized in the training process, *i.e.*, the perturbed

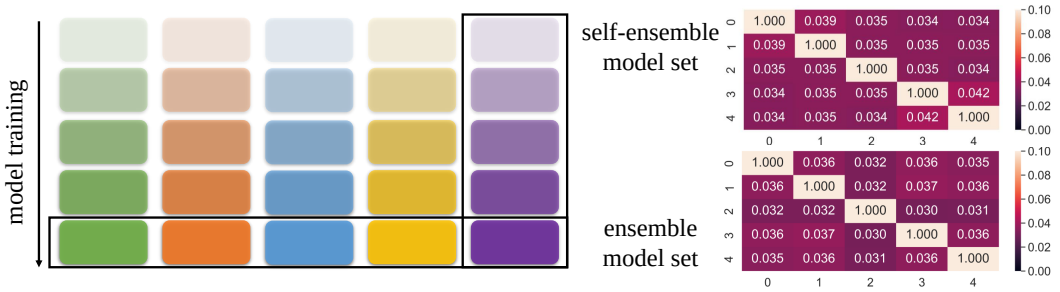


Figure 1: Ensemble and Self-Ensemble. Ensemble trains multiple models (left to right and top to bottom), while self-ensemble trains once and collects intermediate checkpoints (only top to bottom). Thus, self-ensemble costs much less training calculation. Moreover, checkpoints could provide diverse gradients. The right figures show the average absolute cosine value of gradients (on 1K CIFAR-10 images) for each model pair in ensemble (bottom) and self-ensemble (top) sub-models. In ensemble, models 0 to 4 stand for ResNet18, SENet18, VGG16, DenseNet121, and GoogLeNet, respectively, and in self-ensemble, they mean the ResNet18 after training for 24, 48, 72, 96, 120 epochs. Gradient analysis on CIFAR-100 and ImageNet are in Appendix B.

data never correctly predicted by the training model. If we view the training process as the generation of checkpoint models, the problem becomes finding the examples that are adversarial to checkpoints, which could be easily solved by the ensemble attack (Dong et al., 2018).

Let us investigate whether the training checkpoints, which are similar in architecture and parameters (Li et al., 2022), could be diverse sub-models for effective self-ensemble. To measure the diversity, we adopt the common gradient similarity metric (Pang et al., 2019; Yang et al., 2021). In Fig. 1 (upper right), we plot the average of absolute cosine value for gradients in different checkpoints, and the low value indicates that gradients on images are close to orthogonal for different checkpoints like in the ensemble of different architectures (bottom right). This means, surprisingly, intermediate checkpoints are very diverse, and sufficient to form the proposed self-ensemble protection (SEP) as

$$\boldsymbol{G}_{\text{SEP}}(f_{\text{p}}, \boldsymbol{x}) = \sum_{k=0}^{n-1} \boldsymbol{G}_{\text{CE}}(f_{\text{p}}^{k}, \boldsymbol{x}), \tag{3}$$

where f_{p}^{k} is the kth equidistant intermediate checkpoint and $\boldsymbol{G}_{\text{SEP}}$ is the gradients for update in (2). **[R1.3, R2.3]** As illustrated in 1 (left), SEP (vertical box) requires the computation of training only one DNN compared to conventional ensemble (horizontal box) that needs time- and resource-consuming training processes to obtain a large number of ensemble models. This efficiency is especially important for data protection. Because only a large amount of data, if stolen, could be used to train competitive DNNs. In this regard, the scale of data requiring particular protection would be large, and saving the calculation of training extra models makes a significant difference.

SEP is differently motivated compared to conventional ensemble. Ensemble attacks aim to produce architecture-invariant adversarial examples, and such transferable examples reveal the common vulnerability of different architectures (Chen et al., 2022). SEP, in contrast, targets the vulnerability of DNN training. By enforcing consistent mis-classification, SEP produces examples ignored during normal training, and learning on them would thus yield DNNs ignoring normal examples.

3.3 Protecting Data by Self-Ensemble

[R1.1, R3.4] Multiple checkpoints offer us a pool of features for an input. Those representations, though distinctive, all contribute to accurate classification. Thus, we could additionally take the advantage of diverse features besides diverse gradients at no cost. Motivated by this, we resort to the neural collapse theory (Papyan et al., 2020) because it unravels the characteristics of DNN features.

There are four manifestations of neural collapse of DNN. (1) In-class variability of last-layer activation collapse to class means. (2) Class means converge to simplex equiangular tight frame. (3) Linear

classifiers approach class means. (4) Classifier converges to choose the nearest class mean. They demonstrate that the last-layer features of well-trained DNNs center closely on class means. In this regard, the mean feature of in-class samples is a highly representative depiction of this class.

Based on this, we develop the feature alignment loss to jointly use different but good representations of a class from multiple checkpoints. Specifically, for every checkpoint, we encourage the last-layer feature of a sample to approximate the mean feature of target-class samples. In this way, FA promotes neural collapse to incorrect centers so that a sample has the exact high-dimensional feature of the target-class samples. Therefore, non-robust features of that target class could be robustly injected to data so that DNNs are deeply confounded. Mathematically, FA in SEP can be expressed as

$$\boldsymbol{G}_{\text{SEP-FA}}(h_{\text{p}}, \boldsymbol{x}) = \sum_{k=0}^{n-1} \boldsymbol{G}_{\text{FA}}(h_{\text{p}}^k, \boldsymbol{x}) = \sum_{k=0}^{n-1} \nabla_{\boldsymbol{x}} \|h_{\text{p}}^k(\boldsymbol{x}) - h_{\text{c}}^k(g(y))\|, \quad h_{\text{c}}^k(y) = \frac{\sum_{\boldsymbol{x} \in \mathcal{T}_y} h_{\text{p}}^k(\boldsymbol{x})}{|\mathcal{T}_y|}, \tag{4}$$

where h_{p}^k stands for the feature extractor (except for the last linear layer) of f_{p}^k, and $h_{\text{c}}^k(g(y))$ is for the mean (center) feature in the target class $g(y)$ calculated by h_{p}^k. $\|\cdot\|$ means the MSE loss.

Our overall algorithm is summarized in Alg. 1, where we use a stochastic variance reduction (VR) gradient method (Johnson & Zhang, 2013) to avoid bad local minima in optimization. Our method first calculates the FA gradients $\boldsymbol{g}_k$ by each training checkpoint (line 4). Then before updating the sample by the accumulated gradients (line 11), we reduce the variance of ensemble gradients (from $\boldsymbol{g}_{\text{ens}}$ to $\boldsymbol{g}_{\text{upd}}$) in a predictive way by M inner virtual optimizations on $\hat{\boldsymbol{x}}_m$, which has been verified to boost ensemble adversarial attacks (Xiong et al., 2022).

Algorithm 1 Self-Ensemble Protection with Feature Alignment and Variance Reduction

Input: Dataset $\mathcal{T} = \{(\boldsymbol{x}, y)\}$, ℓ_∞ bound ε, step size α, number of protection iterations T, number of training iterations N, number of checkpoints in self-ensemble n, number of inner updates M
Output: Protected dataset $\boldsymbol{x}'$

1: Train a DNN for N epochs and save n equidistant checkpoints
2: $\boldsymbol{x}_0 = \boldsymbol{x}$
3: **for** $t = 0 \to T-1$ **do**
4: **for** $k = 0 \to n-1$ **do** $\boldsymbol{g}_k = \boldsymbol{G}_{\text{FA}}(h_{\text{p}}^k, \boldsymbol{x}_t)$ *# get the gradients by each checkpoints as (4)*
5: $\boldsymbol{g}_{\text{ens}} = \frac{1}{n}\sum_{k=0}^{n-1} \boldsymbol{g}_k$, $\boldsymbol{g}_{\text{upd}} = 0$, $\hat{\boldsymbol{x}}_0 = \boldsymbol{x}_t$ *# initialize variables for inner optimization*
6: **for** $m = 0 \to M-1$ **do**
7: Pick a random index k *# stochastic variance reduction (Johnson & Zhang, 2013)*
8: $\boldsymbol{g}_{\text{upd}} = \boldsymbol{g}_{\text{upd}} + \boldsymbol{G}_{\text{FA}}(h_{\text{p}}^k, \hat{\boldsymbol{x}}_m) - (\boldsymbol{g}_k - \boldsymbol{g}_{\text{ens}})$ *# accumulate gradients with variance*
9: $\hat{\boldsymbol{x}}_{m+1} = \Pi_\epsilon\left(\hat{\boldsymbol{x}}_m - \alpha \cdot \text{sign}(\boldsymbol{g}_{\text{upd}})\right)$ *# virtual update on* $\hat{\boldsymbol{x}}_m$
10: **end for**
11: $\boldsymbol{x}_{t+1} = \Pi_\epsilon\left(\boldsymbol{x}_t - \alpha \cdot \text{sign}(\boldsymbol{g}_{\text{upd}})\right)$ *# update samples with variance-reduced gradients*
12: **end for**
13: **return** $\boldsymbol{x}' = \boldsymbol{x}_{T-1}$

With efficient self-ensemble, our method only requires $1 \times N$ training epochs, and $T \times (n + M)$ times of backward calculation to update samples. However, our method could be amazingly effective in reducing the appropriator model's accuracy as demonstrated in the next section.

4 Experiments

4.1 Setup

We evaluate SEP along with 7 data protection baselines, including adding random noise, TensorClog aiming to cause gradient vanishing (Shen et al., 2019), Gradient Alignment to target-class gradients (Fowl et al., 2021a), DeepConfuse that protects by an autoencoder (Feng et al., 2019), Unlearnable Examples (ULEs) using error-minimization noise (Huang et al., 2020a), Robust ULEs (RULEs) that use adversarial training (Fu et al., 2021), Adversarial Poison (AdvPoison) resorting to targeted attacks (Fowl et al., 2021b), and **[R3.1, R3.3]** AutoRegressive (AR) Poison (Sandoval-Segura et al., 2022) with Markov chain. Hyperparameters of baselines are shown in Appendix D. We use the reproduced results in (Fowl et al., 2021b) in Table 1 & 6.

For our method, we optimize class-y samples to have the mean feature of class $g(y)$, where $g(y) = (y+5)\%10$ for CIFAR-10 (Krizhevsky et al., 2009) and ImageNet (Krizhevsky et al., 2017) protected classes, and $g(y) = (y+50)\%100$ for CIFAR-100. For the ImageNet subset, we train f_{a} and f_{p} with the first 100 classes in ImageNet-1K, but only protect samples in 10 significantly different classes, which are African chameleon, black grouse, electric ray, hammerhead, hen, house finch, king snake, ostrich, tailed frog, and wolf spider. This establishes the class-wise data protection setting, and the reported accuracy is calculated on the testing samples in these 10 classes. We train a ResNet18 for $N = 120$ epochs as f_{p} following (Huang et al., 2020a; Fowl et al., 2021b). 15 equidistant intermediate checkpoints (epoch 8, 16, ..., 120) are adopted with $M = 15, T = 30$ if not otherwise stated. Experiments are conducted on an NVIDIA Tesla A100 GPU but could be run on GPUs with 4GB+ memory, because we store checkpoints on hardware instead of memory.

The data protection methods are assessed by training models with 5 architectures, which are ResNet18 (He et al., 2016), SENet18 (Hu et al., 2018), VGG16 (Simonyan & Zisserman, 2015), DenseNet121 (Huang et al., 2017), and GoogLeNet (Szegedy et al., 2015). They are implemented from Pytorch vision (Paszke et al., 2019). We train appropriator DNNs f_{a} for 120 epochs by an SGD optimizer with an initial learning rate of 0.1, which is divided by 10 in epochs 75 and 90. The momentum item in training is 0.9 and the weight decay is 5e-4. In this setting, DNNs trained on clean data could get great accuracy, *i.e.*, 95% / 75% / 78% for CIFAR-10 / CIFAR-100 / ImageNet subset. Training configurations are the same for f_{a} and f_{p}. In the ablation study, we denote the pure self-ensemble (3) as SEP, SEP with feature alignment (4) as SEP-FA, and SEP-FA with variance reduction as SEP-FA-VR (Alg. 1). In other experiments, "ours" stands for our final method, *i.e.*, SEP-FA-VR. We put the confusion matrix of f_{a} in Appendix C to provide class-wise analysis.

4.2 Uncovering the Vulnerability of DNN Training

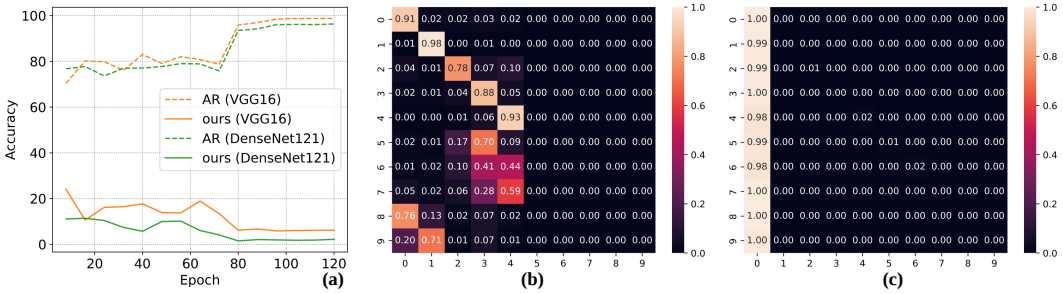


Figure 2: **[R1.3, R2.3]** (a) Our examples generated by ResNet18 are mostly mis-classified during the training of DenseNet121 and VGG16 compared to AR poison $\ell_2 = 1$. (b) The confusion matrix of training with clean (classes 0-4) and protective examples (classes 5-9). (c) The confusion matrix of training with clean (class 0) and protective examples (classes 1-9).

We first investigate whether our examples successfully reveal the vulnerability of DNN training. If so, we should be able to hurt different training processes regardless of the model architecture. In this regard, we test the "cross-training transferability" of our protective data. We report the results in Fig. 2 (a), where one could see that SEP samples generated by ResNet18 training tend to be mis-predicted in training DenseNet121 and VGG16. In contrast, AR samples behave like clean training data and can be well recognized. This demonstrates that our method well depicts the vulnerability of DNN training compared to the recent baseline (Sandoval-Segura et al., 2022).

We also perform a class-wise study to illustrate how DNNs treat clean and protective examples. We first train a CIFAR-10 DNN with clean samples (classes 0-4) and protective ones (classes 5-9, with features of classes 0-4). The DNN performs well in classes 0-4 but mis-classifies all clean samples in classes 5-9 to classes 0-4, seeing Fig. 2 (b). This indicates in DNN's view, clean samples (from different classes) are much closer to each other than to protective ones (which look very similar). More extremely, if we inject features of class 0 to classes 1-9 examples, and use them with clean samples (class 0) to train, the DNN would classify all testing samples to class 0, seeing Fig. 2 (c).

4.3 PROTECTIVE PERFORMANCE

By depicting the vulnerability of DNN training, our method achieves amazing protective performance on 3 datasets and 5 architectures against 8 data protection methods. In table 1, our method surpasses existing state-of-the-art baselines by a large margin, leading DNNs to have < 5.7% / 3.2% / 0.6% accuracy on CIFAR-10 / CIFAR-100 / ImageNet subset. The great performance enables use to set an extremely small bound of $\varepsilon = 2/255$, even for high-resolution ImageNet samples. The perturbations would be invisible even under meticulous inspection, seeing Appendix A. However, the appropriator can only reach no more than 30% accuracy in most cases, seeing Table 2.

Table 1: **[R1.2]** Model testing accuracy trained on the protected dataset ($\ell_\infty = 8/255$).

Dataset	CIFAR-10			CIFAR-100			ImageNet subset		
Model	RN18	VGG16	DN121	RN18	VGG16	DN121	RN18	VGG16	DN121
ULEs	19.93	28.34	20.25	14.81	17.56	13.71	12.20	11.14	15.44
RULEs	27.09	28.17	24.96	10.14	14.39	13.96	13.74	12.77	14.36
AdvPoison	6.25	6.88	6.22	3.49	4.46	3.57	2.30	5.40	4.80
Ours	**4.73**	**5.61**	**3.76**	**2.65**	**3.15**	**2.43**	**0.00**	**0.60**	**0.20**

Table 2: Model testing accuracy trained on slight perturbed dataset ($\ell_\infty = 2/255$).

Dataset	RN18	SENet18	VGG16	DN121	GoogLeNet
CIFAR-10	14.68	15.93	23.66	15.02	17.99
CIFAR-100	21.16	19.48	66.73	21.85	27.49
ImageNet subset	8.80	10.70	27.00	9.40	30.80

[R1.2, R2.1, R3.1] Adversarial training (AT) has been validated as the most effective strategy to recover the accuracy of training with protective perturbations. It does not hinder the practicability of data protection methods because AT significantly decreases the accuracy and requires several-fold training computation. However, it would be interesting to study the effect of different types of perturbations in different AT settings. Here we study with AR Poison, which is claimed to resist AT. We set the perturbation bound as $\ell_2 = 1$ (step size $\alpha = 0.2$) (Sandoval-Segura et al., 2022) and $\ell_0 = 1$ (Wu et al., 2022), and the latter means perturbing one pixel. We keep the AT bound the same as the perturbations bound, and find that in this case, both ℓ_∞ and ℓ_2 ATs could recover accuracy of ℓ_∞, ℓ_2, and $\ell_\infty + \ell_2$ perturbations. The only type of perturbations able to resist ℓ_∞ AT is the mixture of ℓ_∞ and ℓ_0 perturbations. Besides, our method is significantly better than AR Poison in normal training.

Table 3: Performance of perturbations under different norms $\ell_\infty = 8/255, \ell_2 = 1, \ell_0 = 1$ in CIFAR-10 adversarial training of ResNet18. $\ell_\infty + \ell_2$ means mixing (adding) perturbations.

Pert.	ℓ_∞			ℓ_2			$\ell_\infty + \ell_2$			$\ell_\infty + \ell_0$		
AT	None	ℓ_∞	ℓ_2	None	ℓ_∞	ℓ_2	None	ℓ_∞	ℓ_2	None	ℓ_∞	ℓ_2
AR	20.49	84.73	82.66	12.99	82.03	82.86	12.29	82.87	83.47	15.25	**12.28**	**70.26**
ours	**4.73**	**82.51**	**81.91**	**3.67**	**81.52**	**81.89**	**5.05**	**81.54**	**82.25**	**3.43**	13.19	71.01

4.4 ABLATION STUDY

We study the performance improvements from SEP, FA, and VR separately along with the best baseline AdvPoison (Fowl et al., 2021b), seeing Table 1. We first control the overall computation the same for all experiments. Then we vary the number of sub-models n to see its effect on our method.

In Table 4, we maintain our methods to have comparative computation with AdvPoison, which trains the protecting DNN for 40 epochs and craft perturbations by 250 steps for 8 times (we modify it to 4).

In SEP and SEP-FA, we train for $N = 120$ epochs and use $n = 30$ checkpoints to update samples for $T = 30$ times, aligning the computation with AdvPoison. In SEP-FA-VR with inner update $M = 15$, we alter the number of checkpoints $n = 15$ so that the overall computation is the same, which is also the default setting for all experiments as in Sec. 4.1. **[R1.2]** We use ResNet18 as f_{p} on CIFAR-10 dataset here. In the conventional ensemble, 30 DNNs with 5 architectures are trained with 6 seeds.

As shown in Table 4, SEP is able to more than halve the accuracy of AdvPoison within the same computation budget, indicating that knowledge of multiple models is much more important than additional update steps. In comparison with the conventional ensemble, which requires $30\times$ training computation, SEP performs just a little worse. Moreover, equipped with FA, which consumes no additional calculation, efficient SEP-FA could be as effective as the conventional ensemble. With VR, SEP-FA-VR is stably better, and reduces the accuracy from 45.10% to 17.47% on average.

Table 4: Ablation study on Self-Ensemble Protection, Feature Alignment, and Variance Reduction.

Method ($\epsilon = 2/255$)	Train (Epoch)	Crafting (Step)	CIFAR-10 Model Testing Accuracy (↓) RN18	SENet18	VGG16	DN121	GoogLeNet
AdvPoison	40	1000	41.35	40.54	52.22	43.28	48.04
Ensemble	30×120	30×30	16.32	16.91	29.19	16.74	20.34
SEP	120	30×30	19.92	17.81	28.12	18.07	22.48
SEP-FA	120	30×30	16.91	17.01	26.82	16.88	21.15
SEP-FA-VR (ours)	120	30×(15+15)	**14.68**	**15.93**	**23.66**	**15.02**	**17.99**

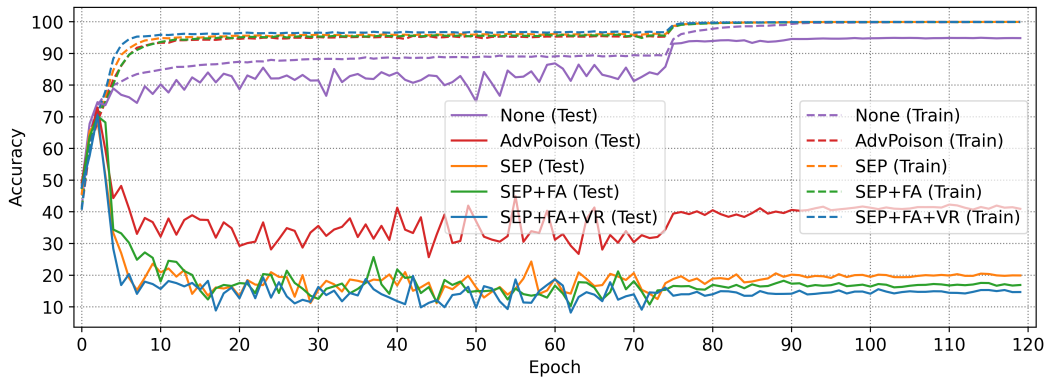


Figure 3: The accuracy trend of training a CIFAR-10 ResNet18 using different data protection methods ($\varepsilon = 2/255$, None means no protection). All methods have high training accuracy v.s. low testing accuracy, but SEP, equipped with FA and VR, reduces model's generalization most effectively.

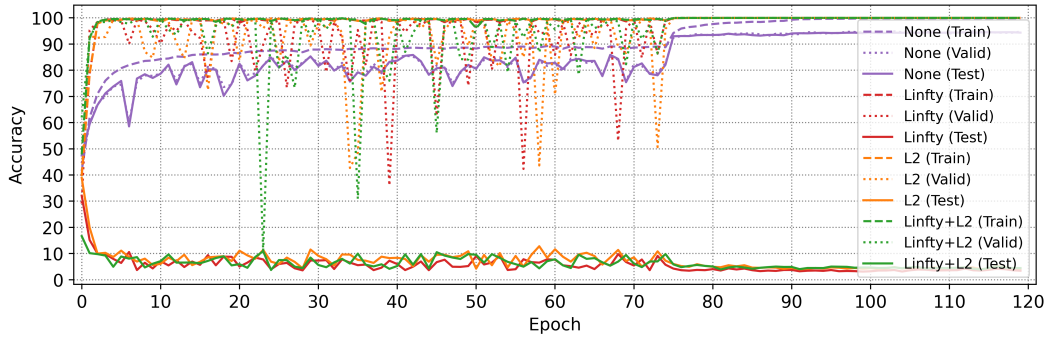


Figure 4: **[R3.2]** The validation (using 2500 samples separated from training data) performance of different types of perturbations by our method (CIFAR-10, $\ell_\infty = 8/255$, $\ell_2 = 1$, $f_{\text{p}} = f_{\text{a}} =$ ResNet18).

We illustrate the training process of Table 4 experiments in Fig. 3. Compared to the training on clean data (purple line), data protection methods accelerate the model's convergence on training data, but the DNN's testing accuracy would suddenly drop at the initial stage of training. After the learning rate decay at epoch 75, the protection performance of different methods could be clearly observed. SEP accounts for the majority of performance improvements, and FA and VR could also further decrease the accuracy, making our method finally outperform the conventional inefficient ensemble. **[R3.2]** We also show the validation accuracy (on unlearned protective examples) of different perturbations in Fig. 4, where it is obvious that early stopping could not be a good defense because validation accuracy is mostly close to training accuracy. However, a huge and unusual gap between them may be a signal for the existence of protective examples.

We also vary the number of intermediate checkpoints n used in self-ensemble to perform ablation study under different computation budgets. We set $n = 3, 5, 10, 30, 120$ without changing other hyper-parameters in self-ensemble, and plot the results in Fig. 5. Similar results could be seen, *i.e.*, FA, and VR could stably contribute to the performance. We also discover that although n is increasing exponentially, the resulting performance increase would not be that significant as n is large. Most prominently, raising n from 30 to 120 is not bound to yield better results, meaning that the performance would saturate around $n = 30$, and it would be unnecessary to use all checkpoints.

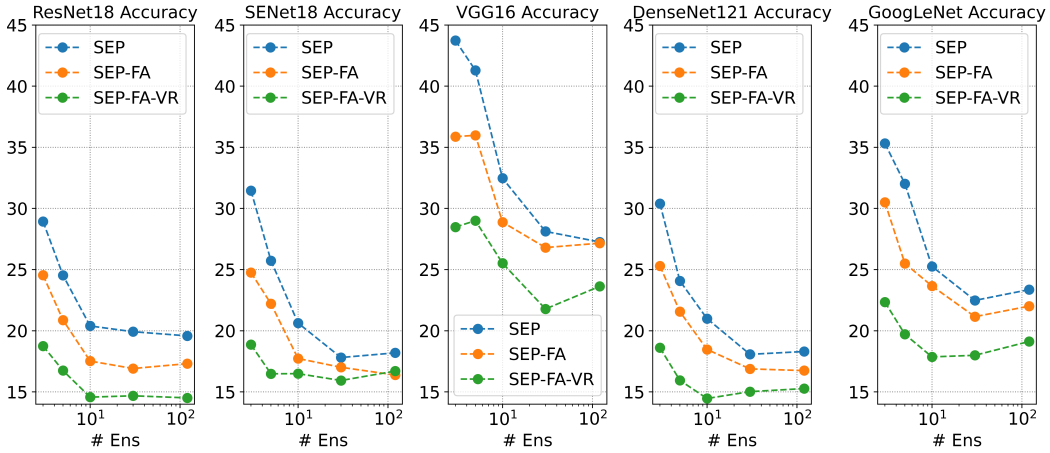


Figure 5: Ablation study on FA and VR with a different number of sub-models in self-ensemble. Results are produced on CIFAR-10 models with the bound of protective perturbations $\varepsilon = 2/255$.

5 Discussion and Conclusion

In this paper, we **[R1.3, R2.3]** propose that data protection should target at the vulnerability of DNN training, which we successfully depict as the examples never classified correctly in training. Such examples could be easily found by model checkpoints, which are found to have surprisingly diverse gradients. By self-ensemble, effective ensemble performance could be achieved by the computation of training one model, and we could also take advantage of the diverse features from checkpoints to further boost the performance by the novel feature alignment loss. Our method exceeds current baselines by a large margin, reducing the appropriator model's accuracy from 45.10% (best-known results) to 17.47% on average.

Our work reveals the possibility of protecting data by extremely small perturbations, *e.g.*, $\varepsilon = 2/255$, without a huge computation budget. The key is to aggregate the knowledge of multiple models, even if they come from a single training process. Our method could also serve as a potential benchmark to evaluate the DNN's learning process, *e.g.*, how to prevent DNNs from learning non-robust features (shortcuts) instead of semantic ones.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 308–318, 2016.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (S&P)*, pp. 39–57. IEEE, 2017.

Sizhe Chen, Zhengbao He, Chengjin Sun, and Xiaolin Huang. Universal adversarial attack on attention and the resulting dataset DAmageNet. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pp. 2188–2197, 2022.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9185–9193, 2018.

Minghong Fang, Neil Zhenqiang Gong, and Jia Liu. Influence function based data poisoning attacks to top-n recommender systems. In *Proceedings of The Web Conference (WWW)*, pp. 3019–3025, 2020.

Ji Feng, Qi-Zhi Cai, and Zhi-Hua Zhou. Learning to confuse: generating training time adversarial data with auto-encoder. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 11971–11981, 2019.

Liam Fowl, Ping-yeh Chiang, Micah Goldblum, Jonas Geiping, Arpit Bansal, Wojtek Czaja, and Tom Goldstein. Preventing unauthorized use of proprietary data: Poisoning for secure dataset release. *arXiv preprint arXiv:2103.02683*, 2021a.

Liam H Fowl, Micah Goldblum, Ping-yeh Chiang, Jonas Geiping, Wojciech Czaja, and Tom Goldstein. Adversarial examples make strong poisons. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 30339–30351, 2021b.

Shaopeng Fu, Fengxiang He, Yang Liu, Li Shen, and Dacheng Tao. Robust unlearnable examples: Protecting data privacy against adversarial learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Jonas Geiping, Liam H Fowl, W Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. Witches' brew: Industrial scale data poisoning via gradient matching. In *International Conference on Learning Representations (ICLR)*, 2020.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7132–7141, 2018.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4700–4708, 2017.

Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. In *International Conference on Learning Representations (ICLR)*, 2020a.

W Ronny Huang, Jonas Geiping, Liam Fowl, Gavin Taylor, and Tom Goldstein. Metapoison: Practical general-purpose clean-label data poisoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 12080–12091, 2020b.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 315–323, 2013.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, pp. 1885–1894, 2017.

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. *Communications of the ACM (CACM)*, pp. 84–90, 2017.

Tao Li, Lei Tan, Zhehao Huang, Qinghua Tao, Yipeng Liu, and Xiaolin Huang. Low dimensional trajectory hypothesis is true: Dnns can be trained in tiny subspaces. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pp. 1–1, 2022.

Jiabin Liu, Zhiquan Qi, Bo Wang, YingJie Tian, and Yong Shi. Self-llp: Self-supervised learning from label proportions with self-ensemble. *Pattern Recognition*, pp. 108767, 2022.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *International Conference on Learning Representations (ICLR)*, 2017.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *European Conference on Computer Vision (ECCV)*, pp. 181–196, 2018.

Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning (ICML)*, pp. 4970–4979, 2019.

Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. In *Proceedings of the National Academy of Sciences (PNAS)*, pp. 24652–24663, 2020.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8024–8035, 2019.

Pedro Sandoval-Segura, Vasu Singla, Jonas Geiping, Micah Goldblum, Tom Goldstein, and David W Jacobs. Autoregressive perturbations for data poisoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Juncheng Shen, Xiaolei Zhu, and De Ma. Tensorclog: An imperceptible poisoning attack on deep neural network applications. In *IEEE Access*, pp. 41498–41506, 2019.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

David Stutz, Matthias Hein, and Bernt Schiele. Confidence-calibrated adversarial training: Generalizing to unseen attacks. In *International Conference on Machine Learning (ICML)*, pp. 9155–9166, 2020.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 843–852, 2017.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations (ICLR)*, 2018.

Hongjun Wang and Yisen Wang. Self-ensemble adversarial training for improved robustness. In *International Conference on Learning Representations (ICLR)*, 2021.

Shutong Wu, Sizhe Chen, Cihang Xie, and Xiaolin Huang. One-pixel shortcut: on the learning preference of deep neural networks. *arXiv preprint arXiv:2205.12141*, 2022.

Yifeng Xiong, Jiadong Lin, Min Zhang, John E Hopcroft, and Kun He. Stochastic variance reduced ensemble adversarial attack for boosting the adversarial transferability. In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14983–14992, 2022.

Huanrui Yang, Jingyang Zhang, Hongliang Dong, Nathan Inkawhich, Andrew Gardner, Andrew Touchet, Wesley Wilkes, Heath Berry, and Hai Li. Dverge: diversifying vulnerabilities for enhanced robust generation of ensembles. pp. 5505–5515, 2020.

Zhuolin Yang, Linyi Li, Xiaojun Xu, Shiliang Zuo, Qian Chen, Pan Zhou, Benjamin Rubinstein, Ce Zhang, and Bo Li. TRS: Transferability reduced ensemble via promoting gradient diversity and model smoothness. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 17642–17655, 2021.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6023–6032, 2019.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Painless adversarial training using maximal principle. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 227–238, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

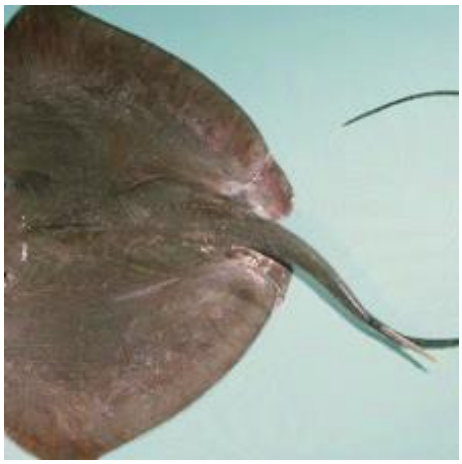
Yuan-Xing Zhao, Yan-Ming Zhang, Ming Song, and Cheng-Lin Liu. Multi-view semi-supervised 3d whole brain segmentation with a self-ensemble network. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 256–265, 2019.

A PROTECTED IMAGENET SAMPLES

African chameleon

black grouse

electric ray



hammerhead

hen

house finch

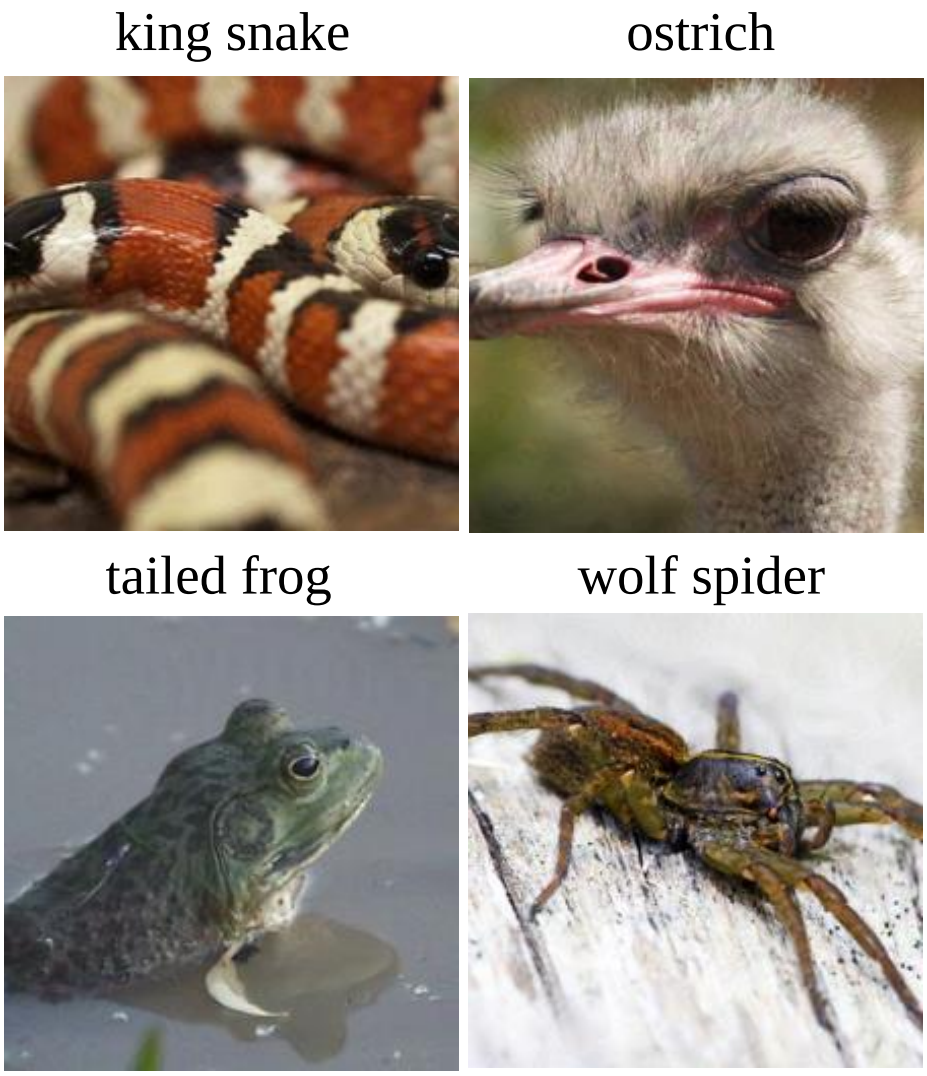


Figure 6: 10-class protected ImageNet samples by our method with $\varepsilon = 2/255$. We select the images with clean backgrounds, but the perturbations are still invisible in such cases. However, it would significantly decrease the model's accuracy to below 30%.

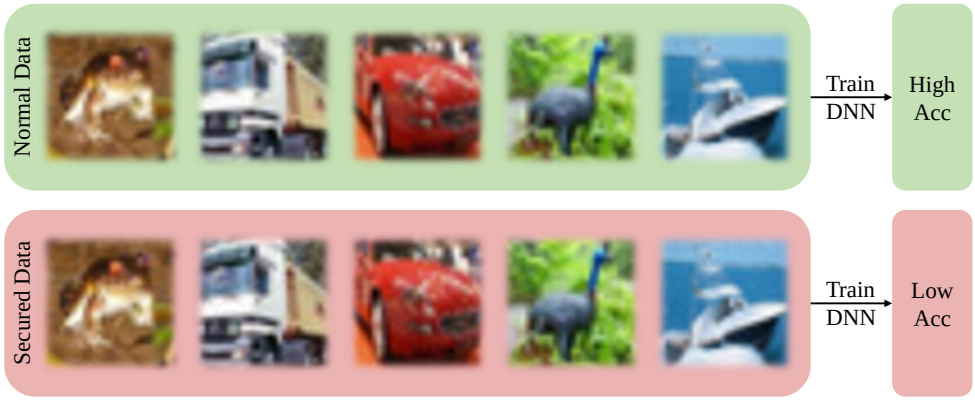


Figure 7: Our protected CIFAR-10 images with $\ell_\infty = 2/255$.

B GRADIENT ANALYSIS OF CIFAR-100 AND IMAGENET CHECKPOINTS

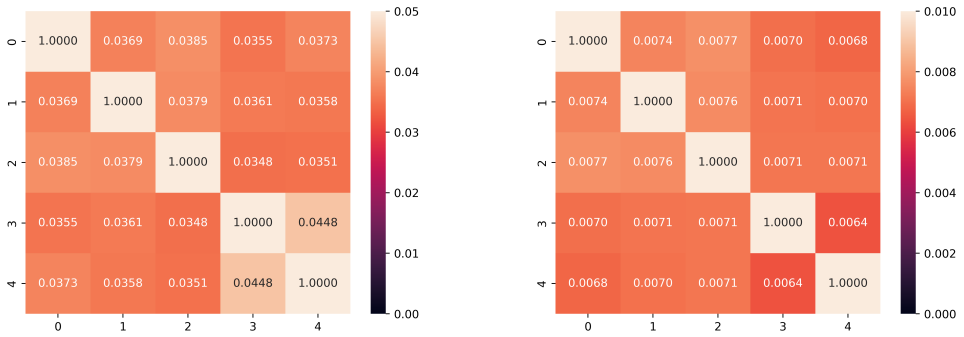


Figure 8: The average absolute cosine value of cross-model gradients for CIFAR-100 (left) and ImageNet (right) checkpoints calculated by 1K samples. The gradients stay nearly orthogonal, indicating good diversity of sub-models in self-ensemble.

C CONFUSION MATRIX OF THE APPROPRIATOR DNN

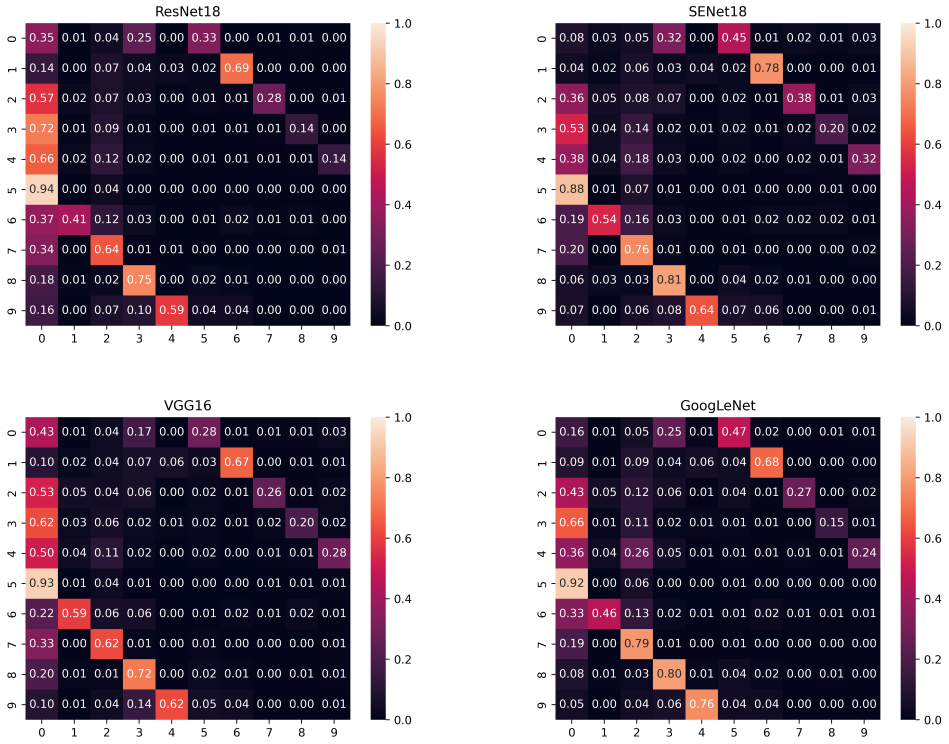


Figure 9: The confusion matrices of DNNs trained on $\varepsilon = 8/255$ CIFAR-10 perturbations by our method. As also shown in (Fowl et al., 2021b), the model tends to predict clean samples to the target class $g(y) = (y + 5)\%10$. We also discover that DNNs prefer to classify them to a fixed class, which is class 0 here, but it may vary during the training. We normalize by the number of samples here.

D OTHER BASELINES AND DEFENSES

[R3.3] We present comparisons with weaker baselines and the overall experimental setups here. Random Noise uses a variance of 8/255. We use the default settings of the baseline papers. TensorClog is with regularization strength 0.01, attack optimization rate 1 and maximum attack iterations 100. Gradient Alignment is with restarts 8 and optimization steps 240. DeepConfuse is with number of trials 500, learning rate of the classification model 0.01, batch size 64 and learning rate for noise generator with 1e-4. Unlearnable Examples is with PGD step 20, step size 8/2550, stop condition error rate 0.01 and attack iterations 10. Robust Unlearnable Examples is with adversarial perturbation radius of REM noise 4/255, sampling number for expectation estimation 5. Adversarial Poison is with PGD step 250, and the step size 1/255. Autoregressive Poison is crafted by padding the image to 36 with a crop of 4 and using 10 autoregressive processes.

Table 5: ResNet18 testing accuracy trained on CIFAR-10 secured by different methods.

Method ($\epsilon = 8/255$)	Accuracy (↓)
None	94.56
Random Noise	90.52
TensorClog (Shen et al., 2019)	84.24
Gradient Alignment (Fowl et al., 2021a)	53.67
DeepConfuse (Feng et al., 2019)	31.10
Ours	**4.73**

Besides adversarial training, we also investigate whether other training strategies can hurt our data protection method. Strategies include image processing (Gaussian smoothing with kernel size 5, adding random noise with variance $8/255$), data augmentations (mixup, cutmix, and cutout with an alpha 1.0), and privacy protection optimization (DPSGD with a clipping parameter 1.0 and noise parameter 0.005). As in Table 6, the above strategies cannot train a good DNN from protective examples, and we also surpass AdvPoison in this setting.

Table 6: Effectiveness of data protection methods under difference training strategies.

Training Strategy ($\epsilon = 8/255$)	AdvPoison	Ours
None	6.25	**4.73** (-1.52)
Gaussian Smoothing	11.94	**9.95** (-1.99)
Adding Random Noise	6.55	**2.89** (-3.66)
Mixup (Zhang et al., 2018)	15.86	**10.06** (-5.80)
Cutmix (Yun et al., 2019)	10.09	**4.47** (-5.62)
Cutout (DeVries & Taylor, 2017)	8.11	**5.51** (-2.60)
DPSGD (Abadi et al., 2016)	24.61	**4.60** (-20.01)