Enhancing Adversarial Training with Second-Order Statistics of Weights

Gaojie Jin[1], Xinping Yi[2], Wei Huang[1], Sven Schewe[1], Xiaowei Huang *[1]

[1] Department of Computer Science, University of Liverpool, Liverpool, UK
[2]Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, UK
{g.jin3, xinping.yi, w.huang23, svens, xiaowei.huang}@liverpool.ac.uk

Abstract

Adversarial training has been shown to be one of the most effective approaches to improve the robustness of deep neural networks. It is formalized as a min-max optimization over model weights and adversarial perturbations, where the weights can be optimized through gradient descent methods like SGD. In this paper, we show that treating model weights as random variables allows for enhancing adversarial training through ***S****econd-Order* ***S****tatistics* ***O****ptimization (S^2O) with respect to the weights. By relaxing a common (but unrealistic) assumption of previous PAC-Bayesian frameworks that all weights are statistically independent, we derive an improved PAC-Bayesian adversarial generalization bound, which suggests that optimizing second-order statistics of weights can effectively tighten the bound. In addition to this theoretical insight, we conduct an extensive set of experiments, which show that S^2O not only improves the robustness and generalization of the trained neural networks when used in isolation, but also integrates easily in state-of-the-art adversarial training techniques like TRADES, AWP, MART, and AVMixup, leading to a measurable improvement of these techniques. The code is available at* https://github.com/Alexkael/S2O.

1. Introduction

It is well known that it is simple to fool convolutional neural networks – to whom we refer as neural networks in this paper – to make incorrect predictions with high confidence by adding human-imperceptible perturbations to their input [27, 50, 72, 80]. Among many different approaches [4, 44, 57, 75, 82] to detect or reduce such adversarial examples, adversarial training [45, 57] is known to be the most effective [3].

Adversarial training is formulated as a min-max optimization problem, where the inner maximization is to find the worst-case adversarial perturbations for the training instances, while the outer minimization is to reduce the loss induced by these adversarial perturbations. While finding the optimal solution to this min-max optimization is challenging, the current wisdom is to first decompose the min-max problem into a master minimization problem and a slave maximization problem, and to then solve them via alternating optimization. Both, the minimization and the maximization, are typically solved by utilizing the gradients of the loss function $\mathcal{L}$. In particular, the master minimization updates the weights according to the gradient over the weight, i.e., $\nabla_{\mathbf{W}}\mathcal{L}$.

This paper takes a drastically different view: we believe that adversarial training can benefit from also considering a second-order statistics over weight. Our study covers both theoretical and empirical perspectives.

Our theoretical argument is obtained through updating the PAC-Bayesian framework [22, 48], which only deals with the model generalization in its original format, by considering adversarial robustness and a second order statistics over weight. Under Bayesian regime, weights are random variables, and a model is a sample drawn from an *a posteriori* distribution. Our update of the framework draws from two aspects as described in Sec. 3. First, by relaxing the unreasonable assumption that all weights are statistically independent, we introduce a second-order statistics of weights, i.e., a weight correlation matrix (or normalized covariance matrix). Second, as in [25], we consider the adversarial robustness in addition to the model generalization.

This updated framework provides a theoretical indication that we may control an adversarial generalization bound during training – and thus improve both robustness and generalization of the resulting model – by monitoring some norms (*e.g.,* singular value, spectral norm, determinant) of the weight correlation matrix. To enable such a control, we need methods to estimate the weight correlation matrix and conduct training, respectively. As described in Sec. 4 we employ two methods for the former: one is

*Corresponding author

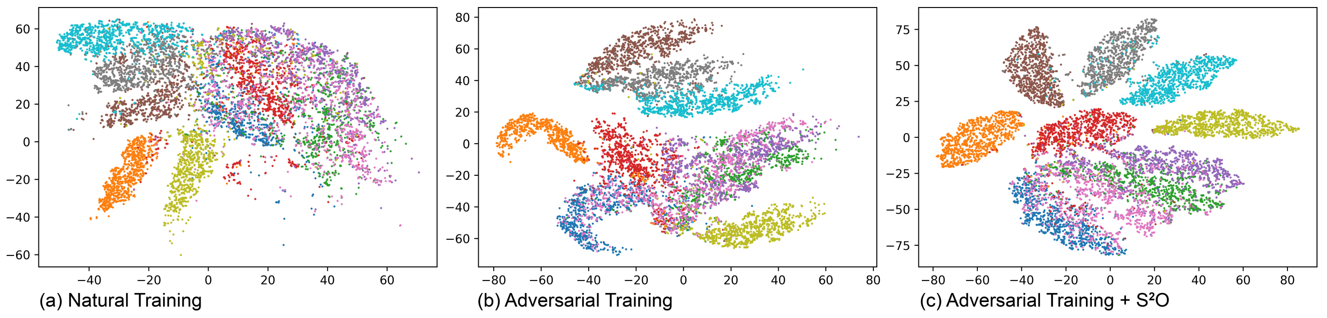


Figure 1. Visualization of the (PGD-20) distributions of penultimate latent representations for each label through t-SNE with trained CNN models on Fashion-MNIST. (**a**) natural training; (**b**) adversarial training (PGD-10); and (**c**) adversarial training (PGD-10) with S^2O.

a sampling method and the other a Laplace approximation method inspired by [9, 61]. For the latter, we propose a novel **S**econd-Order **S**tatistics **O**ptimization (S^2O) method.

To intuitively understand why S^2O can improve the robustness, Fig. 1 shows that adversarial training based on S^2O (abbreviated AT+S^2O) leads to a visually improved separation between points of different classes under PGD-20 attack -– a clear sign of increased robustness.

Through extensive experiments and comparison with the strate-of-the-art in Sec. 5, we show that S^2O can not only significantly improve the robustness and generalization of the trained models by itself, but also enhance the existing adversarial training techniques further. Notably, S^2O can be used in a plug-and-play manner with other adversarial training techniques, including TRADES [85], MART [78], AWP [81], and AVMixup [41], four state-of-the-art adversarial training techniques that represent different directions of the effort to improve the min-max optimization scheme. Importantly, we note that the enhancement from S^2O only leads to a marginal increase of the training time (Sec. 5.2).

We remark that S^2O is different from other second-order adversarial training methods: *e.g.,* [43] proposes an adversarial regularization through approximating the loss function as a second-order Taylor series expansion, [77] improves robustness via Hessian matrix of the input, and [68] studies adversarial attack through Hessian of the weight matrix (while we focus on the weight correlation matrix). It is also different from the "*weight orthogonality*" in [6,49,65]: while the weight correlation matrix is a statistical property of weight matrices treated as random variables, orthogonality treats weight matrices as deterministic variables.

2. Preliminaries

Basic Notation. Let $\mathcal{S} = \{\mathbf{s}_1, ..., \mathbf{s}_m\}$ be a training set with m samples drawn from the input distribution $\mathcal{D}$. As an adversarial sample $\mathbf{s}'$ is slightly different from a clean sample $\mathbf{s}$, we let $\mathcal{S}'$ and $\mathcal{D}'$ be an adversarial set and distribution for a specific model, respectively, such that $||\mathbf{s}' - \mathbf{s}|| \leq \epsilon$ (using by default the ℓ_2-norm). We omit the label y of sample $\mathbf{s}$, as it is clear from the context. Let $\mathbf{W}$, $\mathbf{W}_l$ be the weight matrix, the weight matrix of the l-th layer, respectively. The loss function $\mathcal{L}(\cdot)$ and learning function $f_{\mathbf{W}}(\cdot)$ are parameterized over $\mathbf{W}$. We consider $f_{\mathbf{W}}(\cdot)$ as the n-layer neural networks with h hidden units per layer and activation functions $\mathbf{act}(\cdot)$. We can now express each $f_{\mathbf{W}}(\mathbf{s})$ as $f_{\mathbf{W}}(\mathbf{s}) = \mathbf{W}_n\mathbf{act}(\mathbf{W}_{n-1}...\mathbf{act}(\mathbf{W}_1\mathbf{s})...)$. Note that we omit bias for convenience. At the l-th layer ($l = 1, \dots, n$), the latent representation before and after the activation function is denoted as $\mathbf{h}_l = \mathbf{W}_l\mathbf{a}_{l-1}$ and $\mathbf{a}_l = \mathbf{act}_l(\mathbf{h}_l)$, respectively. We use $||\mathbf{W}_l||_2$ to denote the spectral norm of $\mathbf{W}_l$, defined as the largest singular value of $\mathbf{W}_l$, and $||\mathbf{W}_l||_F$, to denote the Frobenius norm of $\mathbf{W}_l$. We denote the Kronecker product by $\otimes$ and the Hadamard product by $\odot$.

Margin Loss. For any margin $\gamma > 0$, we define the expected margin loss as

$$\mathcal{L}_{\gamma,\mathcal{D}}(f_{\mathbf{W}}) = \mathbb{P}_{\mathbf{s}\sim\mathcal{D}}\Big[f_{\mathbf{W}}(\mathbf{s})[y] \leq \gamma + \max_{j\neq y} f_{\mathbf{W}}(\mathbf{s})[j]\Big],$$

and let $\mathcal{L}_{\gamma,\mathcal{S}}(f_{\mathbf{W}})$ be the empirical margin loss. Note that, setting $\gamma = 0$ corresponds to the normal theoretical classification loss or empirical classification loss, which will be written as $\mathcal{L}_{\mathcal{D}}(f_{\mathbf{W}})$ or $\mathcal{L}_{\mathcal{S}}(f_{\mathbf{W}})$.

3. Adversarial Generalization Bound

Adversarial training is supposed to improve the robustness against adversarial attacks not just on training samples, but also on unseen samples. Thus, the generalization performance of adversarial training (namely adversarial generalization, also known as robust generalization) may differ from that of non-adversarial natural training.

For natural training, PAC-Bayes [22,48] provides an upper bound on the generalization error with respect to the Kullback-Leibler divergence (KL) between the posterior distribution Q and the prior distribution P of weights. Let $f_{\mathbf{W}}$ be any predictor learned from the training data and parametrized by $\mathbf{W}$. We consider the distribution Q in the form of $f_{\mathbf{W}+\mathbf{U}}$ over learned weights $\mathbf{W}$, and denote by $\mathbf{u} = \text{vec}(\mathbf{U})$ (vectorization of $\mathbf{U}$) the multivariate random variable whose distribution may also depend on the training data. Then, for any $\delta, \gamma > 0$, the following bound holds for the margin loss of any $f_{\mathbf{W}}$ with probability $1-\delta$ [53, 54],

$$\mathcal{L}_{\mathcal{D}}(f_{\mathbf{W}}) \leq \mathcal{L}_{\gamma,\mathcal{S}}(f_{\mathbf{W}}) + 4\sqrt{\frac{\mathrm{KL}(Q_{\mathrm{vec}(\mathbf{W})+\mathbf{u}}||P)+\ln\frac{6m}{\delta}}{m-1}}. \tag{1}$$

In the above expression, the KL term is evaluated, for a fixed $\mathbf{W}$, with respect to the only random variable $\mathbf{u}$. That is, the distribution of $\mathrm{vec}(\mathbf{W})+\mathbf{u}$ can be obtained from $\mathbf{u}$ with the mean shifted by $\mathrm{vec}(\mathbf{W})$ and the same covariance matrix (and therefore the same weight correlation matrix).

Notably, such a bound is so general that the inequality holds for any potential prior P and posterior Q. Thus, specifying particular priors and posteriors does not violate the bound but only affects the tightness. Furthermore, [48, 53] present a general framework to construct the posterior for a wide variety of models, including deterministic ones, to calculate the PAC-Bayesian bound.

Given a learning setting, it is a common practice in the literature, *e.g.,* [25, 53, 54], to *assume that the prior P to be a spherical Gaussian $\mathcal{N}(\mathbf{0},\sigma^2\mathbf{I})$, and that the random variable $\mathbf{u}$ also follows $\mathcal{N}(\mathbf{0},\sigma^2\mathbf{I})$ (there is a slight difference between [25] and [54], Appendix A)*. Under such assumption in the adversarial setting with attack methods (*e.g.,* FGM [27], PGM [37], WRM [69]), by letting the clean input domain be norm-bounded as $||\mathbf{s}|| \leq B, \forall \mathbf{s}\in\mathcal{D}$, where $B>0$ is a constant perturbation budget, and for any $\gamma,\delta>0$, any $\mathbf{u}$ *s.t.* $\mathbb{P}_{\mathbf{U}}(\max_{\mathbf{s}}||f_{\mathbf{W}+\mathbf{U}}(\mathbf{s})-f_{\mathbf{W}}(\mathbf{s})||<\frac{\gamma}{4})\geq\frac{1}{2}$ [25, 54], [25] gives the following margin-based PAC-Bayesian adversarial generalization bound,

$$\mathcal{L}_{\mathcal{D}'}(f_{\mathbf{W}}) \leq \mathcal{L}_{\gamma,\mathcal{S}'}(f_{\mathbf{W}}) + \mathcal{O}\left(\sqrt{\frac{(B+\epsilon)^2n^2h\ln(nh)\Phi^{\mathrm{adv}}+\ln\frac{m}{\delta}}{\gamma^2 m}}\right), \tag{2}$$

where Φ^{adv} depends on the attack method (and we omit the coefficient for $\ln\frac{m}{\delta}$ in Eq. (2)). *E.g.*, for an FGM attack, let $\kappa\leq||\nabla_{\mathbf{s}''}\mathcal{L}(f_{\mathbf{W}}(\mathbf{s}''))||$ hold for every $\mathbf{s}''\in\{\mathcal{D}\cup\mathcal{D}'\}$ with constant $\kappa>0$, we have

$$\Phi^{\mathrm{adv}} = \prod_{l=1}^{n}||\mathbf{W}_l||_2^2\Big\{1+\frac{\epsilon}{\kappa}(\prod_{l=1}^{n}||\mathbf{W}_l||_2) \cdot\sum_{l=1}^{n}\prod_{j=1}^{l}||\mathbf{W}_j||_2\Big\}^2\sum_{l=1}^{n}\frac{||\mathbf{W}_l||_F^2}{||\mathbf{W}_l||_2^2}. \tag{3}$$

Details of other Φ^{adv} are given in Appendix A, and a proof is given in [25]. Note that we simplify Eq. (3) to Φ^{adv}, because our new terms of second-order statistics of weights in our bound are unrelated with it.

3.1. Second-Order Statistics in Adversarial Bound

The assumption of a spherical Gaussian distributed $\mathbf{u}$ has greatly simplified the theoretical derivation of the bound in [25, 53, 54]. Based on the same assumption, [25] develops an adversarial PAC-Bayesian bound by considering the impact of attack methods. However, given the complexity of neural networks, this assumption is unrealistic.

In this work, we relax this assumption by letting $\mathbf{u}$ be a non-spherical Gaussian with the correlation matrix $\mathbf{R}$, where $\mathbf{R}\neq\mathbf{I}$ in general, and consider the impact of $\mathbf{R}$ on the above adversarial bound. Specifically, we assume *the correlations of weights from the same layer are not 0, whereas those from different layers are 0.* Therefore, we develop the adversarial bound with the consideration of second-order statistics of weights. Before turning to the theoretical part, we first give the definition of $\mathbf{R}$.

Definition 3.1 *Given the clean sample $\mathbf{s}$ and the adversarial sample $\mathbf{s}'$, let $\mathbf{u}_{\mathbf{s}}$ and $\mathbf{u}_{\mathbf{s}'}$ be Gaussian distributed random vectors with each element being identically distributed as $\mathcal{N}(0,\sigma^2)$ but not independent one another. Then over the entire datasets $\mathcal{S}$ and $\mathcal{S}'$, $\mathbf{u}_{\mathcal{S}}\triangleq\mathbb{E}_{\mathbf{s}}(\mathbf{u}_{\mathbf{s}})$ and $\mathbf{u}_{\mathcal{S}'}\triangleq\mathbb{E}_{\mathbf{s}'}(\mathbf{u}_{\mathbf{s}'})$ obey multivariate Gaussian mixture distributions, respectively, with corresponding correlation matrices as follows:*

$$\mathbf{R}_{\mathcal{S}} \triangleq \frac{1}{\sigma^2}\mathbb{E}_{\mathbf{s}}[\mathbb{E}_{\mathbf{u}}(\mathbf{u}_{\mathbf{s}}\mathbf{u}_{\mathbf{s}}^{\mathsf{T}})], \tag{4}$$

$$\mathbf{R}_{\mathcal{S}'} \triangleq \frac{1}{\sigma^2}\mathbb{E}_{\mathbf{s}'}[\mathbb{E}_{\mathbf{u}}(\mathbf{u}_{\mathbf{s}'}\mathbf{u}_{\mathbf{s}'}^{\mathsf{T}})]. \tag{5}$$

To get a more general adversarial bound, it is reasonable to consider that the true correlation matrix $\mathbf{R}$ under adversarial training is over both $\mathcal{S}$ and $\mathcal{S}'$, rather than only over $\mathcal{S}'$, as most adversarial training methods may utilize both clean data and adversarial data [16, 18, 24, 33, 41, 70, 78, 85–87]. Therefore, we assume that the true $\mathbf{u}$ of an adversarial trained model is a combination of two random variables, $\mathbf{u}=q\mathbf{u}_{\mathcal{S}}+(1-q)\mathbf{u}_{\mathcal{S}'}$ and $\mathbf{R}=q\mathbf{R}_{\mathcal{S}}+(1-q)\mathbf{R}_{\mathcal{S}'}$, where $q\in[0,1]$ is an unknown parameter (as we are not clear how much clean data or adversarial data affects the model).

Denote by $\mathbf{R}_{l,\mathcal{S}}$ and $\mathbf{R}_{l,\mathcal{S}'}$ the weight correlation matrices of l-th layer for clean and adversarial datasets, respectively, following Definition 3.1. In the following, for notational convenience, we let

$$\begin{aligned}\Lambda_{l,\max} &= \max\left(\lambda_{\max}(\mathbf{R}_{l,\mathcal{S}}),\lambda_{\max}(\mathbf{R}_{l,\mathcal{S}'})\right),\\ \Lambda_{l,\min} &= \min\left(\lambda_{\min}(\mathbf{R}_{l,\mathcal{S}}),\lambda_{\min}(\mathbf{R}_{l,\mathcal{S}'})\right),\end{aligned} \tag{6}$$

where $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ are the largest and the smallest singular value of the matrix, respectively. Note that $\mathbf{R}_{l,\mathcal{S}}$ and $\mathbf{R}_{l,\mathcal{S}'}$ are symmetric positive semi-definite matrices, thus their eigenvalues and singular values coincide.

Next, by relaxing the spherical Gaussian assumption in Eq. (2) to consider the non-spherical Gaussian distributed $\mathbf{u}$ with the correlation matrix $\mathbf{R}$, we have the following lemma, as non-spherical Gaussian $\mathbf{u}$ makes an obvious difference during the derivation of the KL term in Eq. (1) (Appendix B).

Lemma 3.2 *Let the posteriori Q be over the predictors of the form $f_{\mathbf{W}+\mathbf{U}}$, where $\mathbf{u}$ is a non-spherical Gaussian with the correlation matrix $\mathbf{R}$. We can get*

$$\mathcal{L}_{\mathcal{D}'}(f_{\mathbf{W}}) \le \mathcal{L}_{\gamma,\mathcal{S}'}(f_{\mathbf{W}}) + \mathcal{O}\Bigg(\Bigg(\frac{-\sum_l \ln\det\mathbf{R}_l + \ln\frac{m}{\delta}}{\gamma^2 m} + \frac{\Psi^{\text{adv}}\big(\sum_l \big(c_1||\mathbf{R}'_l||_2^{\frac{1}{2}} + c_2||\mathbf{R}''_l||_2^{\frac{1}{2}}\big)\big)^2}{\gamma^2 m}\Bigg)^{\frac{1}{2}}\Bigg),$$

where $c_1, c_2 > 0$ are universal constants and

$$\begin{aligned}&\Psi^{\text{adv}} = (B+\epsilon)^2\Phi^{\text{adv}},\\ &\mathbf{R}'_l = \mathbb{E}(\mathbf{U}_l^\intercal\mathbf{U}_l)/\sigma_l^2\\ &= (\mathbf{I}^{h\times h}\otimes\mathbf{1}^{1\times h})(\mathbf{R}_l\odot(\mathbf{1}^{h\times h}\otimes\mathbf{I}^{h\times h}))(\mathbf{I}^{h\times h}\otimes\mathbf{1}^{1\times h})^\intercal,\\ &\mathbf{R}''_l = \mathbb{E}(\mathbf{U}_l\mathbf{U}_l^\intercal)/\sigma_l^2\\ &= (\mathbf{I}^{h\times h}\otimes\mathbf{1}^{1\times h})(\mathbf{R}_l\odot(\mathbf{I}^{h\times h}\otimes\mathbf{1}^{h\times h}))(\mathbf{I}^{h\times h}\otimes\mathbf{1}^{1\times h})^\intercal.\end{aligned}$$

We defer the *proof* to Appendix B. As mentioned in Def. 3.1, $\mathbf{R}_l$ is a combination of $\mathbf{R}_{l,\mathcal{S}}$ and $\mathbf{R}_{l,\mathcal{S}'}$ with an unknown coefficient q. We can use the following two lemmas to refine the bound in Lem. 3.2 through the terms of $\mathbf{R}_{l,\mathcal{S}}$ and $\mathbf{R}_{l,\mathcal{S}'}$.

Lemma 3.3 $||\mathbf{R}'_l||_2^{\frac{1}{2}}$ *and* $||\mathbf{R}''_l||_2^{\frac{1}{2}}$ *can be upper bounded by* $\Lambda'_{l,\max}$ *and* $\Lambda''_{l,\max}$, *i.e.,*

$$\begin{aligned}&||\mathbf{R}'_l||_2^{\frac{1}{2}} \le \Lambda'_{l,\max},\ ||\mathbf{R}''_l||_2^{\frac{1}{2}} \le \Lambda''_{l,\max},\\ &\Lambda'_{l,\max} = \max\big(||\mathbf{R}'_{l,\mathcal{S}}||_2^{\frac{1}{2}}, ||\mathbf{R}'_{l,\mathcal{S}'}||_2^{\frac{1}{2}}\big),\\ &\Lambda''_{l,\max} = \max\big(||\mathbf{R}''_{l,\mathcal{S}}||_2^{\frac{1}{2}}, ||\mathbf{R}''_{l,\mathcal{S}'}||_2^{\frac{1}{2}}\big).\end{aligned} \tag{7}$$

Proof 3.3 *According to [35], we have*

$$\lambda_{\max}(\mathbf{R}'_l) \le q\lambda_{\max}(\mathbf{R}'_{l,\mathcal{S}}) + (1-q)\lambda_{\max}(\mathbf{R}'_{l,\mathcal{S}'}) \le (\Lambda'_{l,\max})^2,$$

and similarly $\lambda_{\max}(\mathbf{R}''_l) \le (\Lambda''_{l,\max})^2$.

Lemma 3.4 *The determinant of* $\mathbf{R}_l$ *can be lower bounded by the term of* $\Lambda_{l,\min}$ *and* $\Lambda_{l,\max}$, *i.e.,*

$$\det\mathbf{R}_l \ge \Lambda_{l,\min}^{k_l}\Lambda_{l,\max}^{h^2-k_l}, \tag{8}$$

where $k_l = (h^2\Lambda_{l,\max} - h^2)/(\Lambda_{l,\max} - \Lambda_{l,\min})$.

Proof 3.4 *For any vector* $\mathbf{x}$, *we have*

$$\begin{aligned}\langle\mathbf{x}, \mathbf{R}_l\mathbf{x}\rangle &= \langle\mathbf{x}, (q\mathbf{R}_{l,\mathcal{S}} + (1-q)\mathbf{R}_{l,\mathcal{S}'})\mathbf{x}\rangle\\ &\ge (q\lambda_{\min}(\mathbf{R}_{l,\mathcal{S}}) + (1-q)\lambda_{\min}(\mathbf{R}_{l,\mathcal{S}'}))||\mathbf{x}||^2\\ &\ge \Lambda_{l,\min}||\mathbf{x}||^2.\end{aligned} \tag{9}$$

Hence, we can get $\lambda_{\min}(\mathbf{R}_l) \ge \Lambda_{l,\min}$. *According to [35], we have* $\lambda_{\max}(\mathbf{R}_l) \le \Lambda_{l,\max}$. *Finally, with the determinant lower bound in [32], we can get Lem. 3.4 directly.*

Lems. 3.2, 3.3 and 3.4 lead to the following corollary.

Corollary 3.5 *Let* $\mathbf{u}$ *be a non-spherical Gaussian with the correlation matrix* $\mathbf{R}$ *over* $\mathcal{S}$ *and* $\mathcal{S}'$. *Then we get*

$$\mathcal{L}_{\mathcal{D}'}(f_{\mathbf{W}}) \le \mathcal{L}_{\mathcal{S}',\gamma}(f_{\mathbf{W}}) + \mathcal{O}\Bigg(\Bigg(\frac{\Psi^{\text{adv}}\big(\sum_l(c_1\Lambda'_{l,\max} + c_2\Lambda''_{l,\max})\big)^2}{\gamma^2 m} + \frac{+\ln\frac{m}{\delta} - \sum_l \ln(\Lambda_{l,\min}^{k_l}\Lambda_{l,\max}^{h^2-k_l})}{\gamma^2 m}\Bigg)^{\frac{1}{2}}\Bigg). \tag{10}$$

Remark 1 *Cor. 3.5 demonstrates an updated adversarial generalization bound with consideration of non-spherical Gaussian* $\mathbf{u}$ *over clean and adversarial data. It also indicates that, assume other coefficients are constant, minimizing* $\Lambda'_{l,\max}$, $\Lambda''_{l,\max}$ *and maximizing* $\Lambda_{l,\min}^{k_l}\Lambda_{l,\max}^{h^2-k_l}$ *can effectively tighten the adversarial generalization bound.*

4. Estimation and Optimization

According to Cor. 3.5, we need to monitor and control the weight correlation matrix and some of its norms – such as the singular value, the spectral norm, and the determinant – during training. To this end, we need to be able to efficiently estimate weight correlation matrix and have a corresponding effective optimization scheme for training.

4.1. Estimation of the Weight Correlation Matrix

We employ two different methods to estimate the weight correlation matrix and, through an inter-comparison between their estimations, to ensure that our empirical conclusions are not compromised by the estimation errors. One is a sampling method, and the other is Laplace approximation of neural networks [9, 61]. Note that, though we only use Laplace approximation to optimize the second-order statistics terms (S^{2}O) during training due to time complexity of sampling (Sec. 4.2), our empirical results in Sec. 5.1 indicate that S^{2}O is applicable with both methods.

Sampling method obtains a set of weight samples $(\mathbf{W}+\eta)$ by a sharpness-like method [29, 34] *s.t.* $|\mathcal{L}(f_{\mathbf{W}+\eta}) - \mathcal{L}(f_{\mathbf{W}})| \le \epsilon'$ (*e.g.*, $\epsilon' = 0.05$ for CIFAR-10 and $\epsilon' = 0.1$ for CIFAR-100), where $\text{vec}(\eta)$ is a $\mathbf{0}$ mean Gaussian noise. These samples are then used to estimate the correlation matrix of $\mathbf{u}_{\mathcal{S}}$ and $\mathbf{u}_{\mathcal{S}'}$. More details are given in Appendix C.

Laplace approximation is an estimation method widely used in Bayesian framework to approximate posterior densities or posterior moments [61, 63, 74]. Technically, it approximates the posterior (*e.g.*, $\text{vec}(\mathbf{W}) + \mathbf{u}$) by a Gaussian distribution with the second-order Taylor expansion of the ln posterior around its MAP estimate. Specifically, given weights for layer l with an MAP estimate $\mathbf{W}_l^*$ on $\mathcal{S}$ (we omit the estimation on $\mathcal{S}'$ as it is the same with $\mathcal{S}$), we have

$$\ln p(\text{vec}(\mathbf{W}_l) + \mathbf{u}_l|\mathcal{S}) \approx \ln p\Big(\text{vec}(\mathbf{W}_l^*)|\mathcal{S}\Big) - \frac{1}{2}\Big(\text{vec}(\mathbf{W}_l - \mathbf{W}_l^*) + \mathbf{u}_l\Big)^\intercal\mathbb{E}_{\mathbf{s}}[\mathbf{H}_l]\Big(\text{vec}(\mathbf{W}_l - \mathbf{W}_l^*) + \mathbf{u}_l\Big), \tag{11}$$

where $\mathbb{E}_{\mathbf{s}}[\mathbf{H}_l]$ is the expectation of the Hessian matrix over input data sample $\mathbf{s}$, and the Hessian matrix $\mathbf{H}_l$ is given by $\mathbf{H}_l = \frac{\partial^2 \mathcal{L}(f_{\mathbf{W}}(\mathbf{s}))}{\partial \mathrm{vec}(\mathbf{W}_l)\partial \mathrm{vec}(\mathbf{W}_l)}$.

It should be noted that, in Eq. (11), the first-order Taylor polynomial has been dropped because the gradient around the MAP estimate $\mathbf{W}_l^*$ is zero. Then, taking a closer look at Eq. (11), we find that its second line is exactly the logarithm of the probability density function of a Gaussian distributed multivariate random variable with mean $\mathbf{W}_l^*$ and covariance $\Sigma_l = \mathbb{E}_{\mathbf{s}}[\mathbf{H}_l]^{-1}$, *i.e.*, $\mathrm{vec}(\mathbf{W}_l) + \mathbf{u}_l \sim \mathcal{N}(\mathrm{vec}(\mathbf{W}_l^*), \Sigma_l)$, where Σ_l can be viewed as the covariance matrix of $\mathbf{u}_l$ and learned weights $\mathbf{W}_l$ can be seen as the MAP estimate $\mathbf{W}_l^*$.

Laplace approximation indicates that efficiently estimating Σ_l is achievable through the inverse of the Hessian matrix, because $\Sigma_l^{-1} = \mathbb{E}_{\mathbf{s}}[\mathbf{H}_l]$. *Note that we omit* $\Sigma_{l,\mathcal{S}'}^{-1} = \mathbb{E}_{s'}[\mathbf{H}_l]$ *as it is similar to* $\Sigma_{l,\mathcal{S}}^{-1} = \mathbb{E}_s[\mathbf{H}_l]$. Moreover, [9, 61] developed a Kronecker factored Laplace approximation based on insights from second-order optimization of neural networks. That is, in contrast to the classical second-order methods [7, 66] with high computational costs for deep neural networks, they suggest that Hessian matrices of l-th layer can be Kronecker factored, i.e.,

$$\mathbf{H}_l = \underbrace{\mathbf{a}_{l-1}\mathbf{a}_{l-1}^\intercal}_{\mathcal{A}_{l-1}} \otimes \underbrace{\frac{\partial^2 \ell(f_{\mathbf{W}}(\mathbf{s}))}{\partial \mathbf{h}_l \partial \mathbf{h}_l}}_{\mathcal{H}_l} = \mathcal{A}_{l-1} \otimes \mathcal{H}_l, \tag{12}$$

where $\mathcal{A}_{l-1} \in \mathbb{R}^{h\times h}$ is the covariance of the post-activation of the previous layer, and $\mathcal{H}_l \in \mathbb{R}^{h\times h}$ is the Hessian matrix of the loss with respect to the pre-activation of the current layer, and h is the number of neurons for each layer. With the assumption that $\mathcal{A}_{l-1}$ and $\mathcal{H}_l$ are independent [9, 61], we can approximate $\mathbb{E}_{\mathbf{s}}[\mathbf{H}_l]$ with

$$\mathbb{E}_{\mathbf{s}}[\mathbf{H}_l] = \mathbb{E}_{\mathbf{s}}[\mathcal{A}_{l-1} \otimes \mathcal{H}_l] \approx \mathbb{E}_{\mathbf{s}}[\mathcal{A}_{l-1}] \otimes \mathbb{E}_{\mathbf{s}}[\mathcal{H}_l]. \tag{13}$$

4.2. A Novel Optimization Scheme S^{2}O for Training

The adversarial training of a neural network is seen as a process of optimizing over an adversarial objective function J_{adv}. To tighten the adversarial bound in Cor. 3.5, we add the second-order statistics penalty terms $\Lambda'_{l,\max}$, $\Lambda''_{l,\max}$ and $-\ln \Lambda_{l,\min}^{k_l}\Lambda_{l,\max}^{h^2-k_l}$ to the objective function J_{adv}, and denote the new objective function as $\tilde{J}_{\mathrm{adv}}$. To reduce the complexity, we approximate $\nabla_{\mathbf{w}_l}(\Lambda'_{l,\max} + \Lambda''_{l,\max} - \ln \Lambda_{l,\min}^{k_l}\Lambda_{l,\max}^{h^2-k_l})$ through

$$\nabla_{\mathbf{w}_l}(\mathbf{g}(\mathbf{R}_l)) = \frac{\partial\big(||\mathbf{R}_{l,\mathcal{S}}||_F^2 + ||\mathbf{R}_{l,\mathcal{S}'}||_F^2\big)}{\partial \mathrm{vec}(\mathbf{W}_l)}. \tag{14}$$

Although it is impractical to find the exact relation between $\mathbf{R}_l$ and above penalty terms (*e.g.*, perfect positive or negative), they are clearly related. In particular, when $\mathbf{R}_{l,\mathcal{S}}$ and $\mathbf{R}_{l,\mathcal{S}'}$ have the same off-diagonal elements as $r_{\mathbf{s}}$ and $r_{\mathbf{s}'}$, respectively, and $r_{\mathbf{s}}r_{\mathbf{s}'} \geq 0$, we have the following lemmas.

Lemma 4.1 *Decreasing* $||\mathbf{R}_{l,\mathcal{S}}||_F^2$ *and* $||\mathbf{R}_{l,\mathcal{S}'}||_F^2$ *leads to a decline in* $|r_{\mathbf{s}}|$ *and* $|r_{\mathbf{s}'}|$*, and further causes reduced* $\Lambda'_{l,\max}$ *and* $\Lambda''_{l,\max}$.

Lemma 4.2 *Decreasing* $||\mathbf{R}_{l,\mathcal{S}}||_F^2$ *and* $||\mathbf{R}_{l,\mathcal{S}'}||_F^2$ *leads to an increase in* $\Lambda_{l,\min}^{k_l}\Lambda_{l,\max}^{h^2-k_l}$.

Proofs are given in Appendix D. We also provide more general case simulations of the relationship between $||\mathbf{R}_{l,\mathcal{S}}||_F^2$, $||\mathbf{R}_{l,\mathcal{S}'}||_F^2$ and the above penalty terms in Appendix E.

Remark 2 *Lems. 4.1, 4.2 and simulations in Appendix E indicate that we can decrease* $\Lambda'_{l,\max}$*,* $\Lambda''_{l,\max}$ *and increase* $\Lambda_{l,\min}^{k_l}\Lambda_{l,\max}^{h^2-k_l}$ *by decreasing* $||\mathbf{R}_{l,\mathcal{S}}||_F^2$ *and* $||\mathbf{R}_{l,\mathcal{S}'}||_F^2$.

It is noted that a direct optimization over Eq. (14) is also computationally prohibitive. Fortunately, Laplace approximation can greatly reduce the complexity of Eq. (14). Specifically, according to Eq. (13) and $\Sigma_l = \mathbb{E}[\mathbf{H}_l]^{-1}$, the following term

$$\nabla_{\mathbf{w}_{l-1}}(\mathbf{g}(\mathbf{A}_{l-1})) = \frac{\partial\big(||\mathbf{A}_{l-1,\mathcal{S}}||_F^2 + ||\mathbf{A}_{l-1,\mathcal{S}'}||_F^2\big)}{\partial \mathrm{vec}(\mathbf{W}_{l-1})}, \tag{15}$$

can be used to approximate $\nabla_{\mathbf{w}_l}(\mathbf{g}(\mathbf{R}_l))$, where $\mathbf{A}_{l-1,\mathcal{S}}$ is the normalization of $\mathbb{E}_{\mathbf{s}}[\mathcal{A}_{l-1}]^{-1}$, *i.e.*, $\forall\ 0 < i, j \leq h$ and for $i, j \in \mathbb{N}$,

$$(\mathbf{A}_{l-1,\mathcal{S}})_{[ij]} = \frac{(\mathbb{E}_{\mathbf{s}}[\mathcal{A}_{l-1}]^{-1})_{[ij]}}{\sqrt{(\mathbb{E}_{\mathbf{s}}[\mathcal{A}_{l-1}]^{-1})_{[ii]}(\mathbb{E}_{\mathbf{s}}[\mathcal{A}_{l-1}]^{-1})_{[jj]}}}. \tag{16}$$

Finally, we add the regularizer $\mathbf{g}(\mathbf{A})$ to the adversarial training objective function J_{adv}, obtaining the new objective function $\tilde{J}_{\mathrm{adv}}$ with

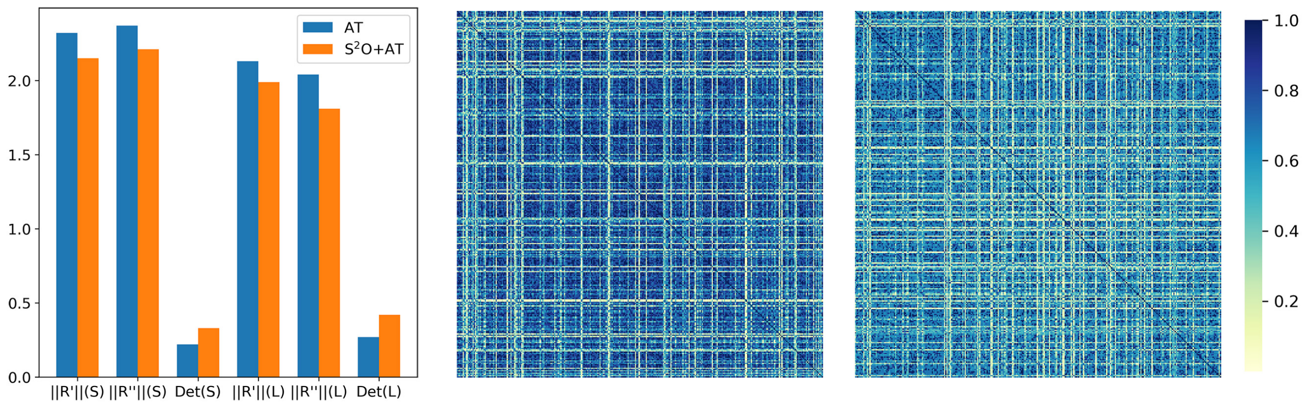
$$\nabla_{\mathbf{W}}\tilde{J}_{\mathrm{adv}} = \nabla_{\mathbf{W}}J_{\mathrm{adv}} + \alpha\nabla_{\mathbf{W}}\mathbf{g}(\mathbf{A}). \tag{17}$$

Here, $\alpha \in [0, \infty)$ is a hyper-parameter to balance the relative contributions of the second-order statistics penalty term $\mathbf{g}(\mathbf{A})$ and the original objective function J_{adv} (Appendix F).

5. Empirical Results

In this section, we first provide a comprehensive understanding of our S^{2}O training method, and then evaluate its robustness on benchmark data sets against various white-box and black-box attacks.

Experimental Setup. We train PreAct ResNet-18 [28] for ℓ_∞ and ℓ_2 threat models on CIFAR-10/100 [36] and SVHN [52] (Tab. 1). In addition, we also train WideResNet-34-10 [83] for CIFAR-10 with an ℓ_∞ threat model (Tabs. 2 and 3). We follow the settings of [60]: for the ℓ_∞ threat model, $\epsilon = 8/255$ and step size $2/255$; for the ℓ_2 threat model, $\epsilon = 128/255$ and step size $15/255$ for all data



(a) Some normalized norms of AT and AT+S²O

(b) Absolute (partial) weight correlation matrix of AT

(c) Absolute (partial) weight correlation matrix of AT+S²O

Figure 2. We train PreAct ResNet18 with AT and AT+S^{2}O, and show the results of partial weights. **(a)** shows the normalized spectral norm of $\mathbf{R}'_{S'}$, $\mathbf{R}''_{S'}$, and the determinant of $\mathbf{R}_{S'}$, with sampling estimation (S) and Laplace approximation (L) respectively. **(b)** and **(c)** demonstrate the absolute correlation matrix of partial weights, for AT and AT+S^{2}O respectively.

Table 1. Adversarial training across data sets on PreAct ResNet18 (%).

Threat Model	Method	CIFAR-10			CIFAR-100			SVHN		
		Clean	PGD-20	Time/epoch	Clean	PGD-20	Time/epoch	Clean	PGD-20	Time/epoch
ℓ_∞	AT	82.41	52.77	309s	58.02	28.02	307s	93.17	60.91	509s
	AT+S^{2}O	**83.65**	**55.11**	368s	58.45	**30.58**	371s	93.39	**64.83**	595s
ℓ_2	AT	88.83	68.83	292s	64.21	42.20	290s	94.02	66.76	477s
	AT+S^{2}O	**89.57**	**69.42**	364s	**65.32**	**44.07**	366s	**94.93**	**76.19**	586s

Table 2. TRADES ($1/\lambda = 6$) and AWP on CIFAR-10 with ℓ_∞ threat model (%).

Method	ResNet18						WideResNet					
	Clean	FGSM	PGD-20	PGD-100	CW-20	AA	Clean	FGSM	PGD-20	PGD-100	CW-20	AA
TRADES	82.89	58.72	53.81	53.69	51.83	48.6	83.98	61.08	56.82	56.53	54.54	52.7
TRADES+AWP	82.30	59.48	56.18	55.90	53.12	51.7	84.99	63.11	59.67	59.42	57.41	**56.2**
TRADES+S^{2}O	**84.15**	60.19	55.20	54.73	52.47	49.5	85.67	62.73	58.34	57.69	55.36	54.1
TRADES+AWP+S^{2}O	83.79	**60.27**	**57.29**	**56.51**	**53.84**	**52.4**	**86.01**	**64.16**	**61.12**	**60.46**	**57.93**	55.9

sets. In all experiments, the training/test attacks are PGD-10/(PGD-20 and others) respectively. All models (except SVHN) are trained for 200 epochs using SGD with momentum 0.9, batch size 128, weight decay 5×10^{-4}, and an initial learning rate of 0.1 that is divided by 10 at the 100th and 150th epochs. For SVHN, we use the same parameters except for setting the starting learning rate to 0.01. Simple data augmentations, such as 32×32 random crop with 4-pixel padding and random horizontal flip, are applied. We implement each PreAct ResNet18 on single GTX 1080 Ti and each WideResnet on single NVIDIA A100.

White-box attack. We conduct white-box attacks, including FGSM [27], PGD-20/100 [45], and CW-20 [10] (the ℓ_∞ version of CW loss optimized by PGD-20), on the models trained with baseline methods and our S^{2}O-enhanced variants.

Black-box attack. Black-box attacks are created from the clean test data by attacking a surrogate model with an architecture that is either a copy of the defense model or a more complex model [56]. After constructing adversarial examples from each of the trained models, we apply these adversarial examples to the other models and evaluate the performances. The attacking methods we have used are FGSM and PGD-20.

Table 3. AVMixup and MART on CIFAR-10 with ℓ_∞ threat model for WideResNet (%).

Method	Clean	FGSM	PGD-20	PGD-100	CW-20	AA
AVMixup	92.56	80.46	59.75	49.51	54.53	39.7
AVMixup+S^{2}O	**93.72**	**84.57**	**60.43**	50.49	**56.16**	39.3
MART	83.51	61.53	58.31	57.55	54.33	51.2
MART+S^{2}O	83.91	**62.56**	**59.29**	**58.33**	**55.14**	**54.1**

Auto Attack. We consider Auto Attack (AA) [13], a powerful and reliable attack, which attacks through an ensemble of different parameter-free attacks that include three white-box attacks (APGD-CE [13], APGD-DLR [13], and FAB [12]) and a black-box attack (Square Attack [2]).

By default, we use the setting $\alpha = 0.3$ for S^{2}O, with the exception of $\alpha = 0.1$ for AT+S^{2}O (CIFAR-10) in Tab. 1. Following [85], we set epsilon=0.031and step size=0.003 for PGD and CW evaluation. And we use standard version auto attack evaluation.

Table 4. TRADES ($1/\lambda = 6$) and AWP on CIFAR-100 with ℓ_∞ threat model for WideResNet (%).

Method	Clean	FGSM	PGD-20	CW-20	AA
TRADES	60.38	35.01	32.11	28.93	26.9
TRADES+LBGAT	60.43	-	35.50	**31.50**	29.3
TRADES+AWP	60.27	36.12	34.04	30.64	28.5
TRADES+S^{2}O	63.40	35.96	33.06	29.57	27.6
TRADES+AWP+S^{2}O	**64.17**	**37.98**	**35.95**	31.26	**29.9**

Table 5. VGG16 and MobileNetV2 on CIFAR-10 with ℓ_∞ threat model (%).

Method	Clean	FGSM	PGD-20	CW-20	AA
VGG16 AT	81.63	53.23	49.21	48.01	43.1
VGG16 AT+S^{2}O	**82.57**	**54.03**	**50.53**	48.15	**44.6**
MNV2 AT	81.97	55.52	50.76	49.53	44.9
MNV2 AT+S^{2}O	**82.48**	**57.51**	**52.93**	49.92	**45.7**

5.1. Empirical Understanding of S^{2}O

In this part, we explore how the second-order statistics of weights (*e.g.*, weight correlation matrix) change when we apply S^{2}O on it. The results in Fig. 2 indicate that S^{2}O effectively decreases the spectral norm of $\mathbf{R}'_{S'}$, $\mathbf{R}''_{S'}$ and increases the determinant of $\mathbf{R}_{S'}$. We also provide the results of clean data in Appendix E.

5.2. Robustness Under White-Box Attacks and Auto Attack

Applying S^{2}O on vanilla adversarial training (Tab. 1). We employ PreAct ResNet-18 to explore the power of our proposed S^{2}O method embedded with normal PGD-10 training (with ℓ_∞ and ℓ_2 threat models), across a number of data sets including CIFAR-10, CIFAR-100, and SVHN.

Tab. 1 suggests that S^{2}O-enhanced variants can improve both, the accuracy (over clean data) and the robust accuracy (over PGD-20 attack), across the three datasets. For example, the accuracy on PGD-20 of the AT+S^{2}O model is 2%-3% higher than that of the standard adversarial training model with an ℓ_∞ threat model on CIFAR-10. There is also an increase of 1%-1.5% in accuracy on clean data when compared to the standard adversarial training model. Generally, the improvement is very consistent across data sets and attacks.

Applying S^{2}O on TRADES and AWP (Tab. 2). We employ PreAct ResNet-18 and WideResNet to explore the performance of our S^{2}O method when it works with two state-of-the-art methods, TRADES and TRADES+AWP, on the CIFAR-10 (under an ℓ_∞ threat model). The robustness of all defense models are tested against white-box FGSM, PGD-20, PGD-100, CW-20 attacks, and auto attack.

Tab. 2 shows that S^{2}O-enhanced variants perform consistently and significantly better than the existing ones (with only one exception). While AWP improves over TRADES, S^{2}O can further enhance it. For example, the accuracy on PGD-20 of the S^{2}O-enhanced TRADES+AWP model is 1.45% higher than that of the TRADES+AWP model on WideResNet, and the accuracy on PGD-20 of the S^{2}O-

Table 6. Sensitivity analysis on CIFAR-10 with ℓ_∞ threat model for ResNet18 (%).

Method	Clean	AA	PGD-20$_{train}$	PGD-20$_{test}$	Gap
AT	82.41	47.1	62.33	52.77	9.56
AT+S^{2}O (0.05)	83.22	**48.5**	61.99	53.82	8.17
AT+S^{2}O (0.1)	**83.65**	48.3	61.50	**55.11**	6.39
AT+S^{2}O (0.2)	83.43	47.8	60.27	54.59	5.68
AT+S^{2}O (0.3)	82.89	46.5	59.36	54.24	5.12
AT+S^{2}O (0.4)	82.54	46.7	58.41	52.92	5.49

Table 7. Adversarial training across data sets on PreAct ResNet18 with black-box attacks under ℓ_∞ threat model (%).

Data	Method	FGSM	PGD-20
CIFAR-10	AT	64.32	62.63
	AT+S^{2}O	**65.63**	**63.87**
CIFAR-100	AT	38.55	37.36
	AT+S^{2}O	**39.68**	**38.60**
SVHN	AT	71.77	63.76
	AT+S^{2}O	72.20	**64.31**

enhanced TRADES model is 1.39% higher than that of the TRADES model on PreAct ResNet-18. For additional experiments on CIFAR-100 in Tab. 4, compared with TRADES+LBGAT [14], S^{2}O can also improve robustness under most attacks.

Applying S^{2}O on AVMixup and MART (Tab. 3). We employ WideResNet to study the performance of our S^{2}O method when it works with some other state-of-the-art methods, such as AVMixup and MART, on the CIFAR-10 data set (under ℓ_∞ threat model). The robustness of all defense models is tested against white-box FGSM, PGD-20, PGD-100, CW-20 attacks, and auto attack.

Tab. 3 also shows that S^{2}O enhanced models perform better than normal AVMixup and MART under most attacks (and clean data). Note that we mark the results of AVMixup with the color cyan under PGD-100 attack and auto attack (AA), as AVMixup (including AVMixup+S^{2}O) is not a robust method across all attacks – it performs not so well under PGD-100 and Auto attacks.

Remark on our Baselines in Tabs. 1 to 4. We have checked that our baselines are close to, or slightly better than, the baselines in the recent paper [81], with which we have very similar experimental settings. We omit the standard deviations of 3 runs as they are very small ($< 0.40\%$).

Supplement. We provide hyper-parameter (α) sensitivity analysis in Tab. 6 with PreAct ResNet-18 and CIFAR-10. We also apply S^{2}O to other structures (VGG16 [67] and MobileNetV2 [64]) with normal adversarial training; Tab. 5 shows that S^{2}O also works on these two structures. In addition, we notice that SOAR [43] can get 56.06% accuracy on ResNet-10, CIFAR-10 under PGD-20 attack; it is interesting to combine S^{2}O with SOAR in the future work.

5.3. Robustness Under Black-Box Attacks

We also employ PreAct ResNet-18 to explore the power of our proposed S^{2}O method embedded with normal PGD-10 training under black-box attacks (with an ℓ_∞ threat model), across a number of data sets, including CIFAR-10/100, and SVHN. For the same data set, all black-box attacks are generated by the same adversarial training model.

Tab. 7 suggests that S^{2}O enhanced models also can get some improvements under black box attacks.

6. Related Work

Adversarial training updates the minimization objective of the training scheme from the usual one to

$$J_{\text{adv}} = \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{D}} \left[\max_{||\mathbf{s}'-\mathbf{s}||\leq\epsilon} \ell(f_{\mathbf{W}}(\mathbf{s}')) \right], \tag{18}$$

where $\mathbf{s}'$ is an adversarial example causing the greatest loss within an ϵ-ball centered at a clean example $\mathbf{s}$ with respect to a norm distance (by default ℓ_2). Adversarial training methods roughly fall within three categories, and in the following, we highlight four state-of-the-art methods, which are combined with our S^{2}O method in the experiments.

The first category is to reduce Eq. (18) to an equivalent, or approximate, expression, which includes measuring the distance between $\mathbf{s}$ and $\mathbf{s}'$. For example, ALP [24, 33] enforces the similarity between $f_{\mathbf{W}}(\mathbf{s})$ and $f_{\mathbf{W}}(\mathbf{s}')$, the logits activations on unperturbed and adversarial versions of the same image $\mathbf{s}$. MMA [16] encourages every correctly classified instance $\mathbf{s}$ to leave sufficiently large margin, i.e., the distance to the boundary, by maximizing the size of the shortest successful perturbation. **MART** [78] observes the difference of misclassified and correctly classified examples in adversarial training, and suggests different loss functions for them. **TRADES** [85] analyzes the robustness error and the clean error, and shows an upper bound and lower bound on the gap between robust error and clean error, which motivates adversarial training networks to optimize J_{tr} with

$$\mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{D}} \left[\ell(f_{\mathbf{W}}(\mathbf{s})) + \max_{||\mathbf{s}'-\mathbf{s}||\leq\epsilon} \text{KL}\big(f_{\mathbf{W}}(\mathbf{s})||f_{\mathbf{W}}(\mathbf{s}')\big)/\lambda \right], \tag{19}$$

where λ is the hyper-parameter to control the trade-off between clean accuracy and robust accuracy. It considers the KL-divergence of the activations of the output layer, i.e., $\text{KL}(f_{\mathbf{W}}(\mathbf{s})||f_{\mathbf{W}}(\mathbf{s}'))$, for every instance $\mathbf{s}$. The measurement over $\mathbf{s}$ and $\mathbf{s}'$ can be extended to consider a *local* distributional distance, i.e., the distance between the distributions within a norm ball of $\mathbf{s}$ and within a norm ball of $\mathbf{s}'$. For example, [87] forces the similarity between local distributions of an image and its adversarial example, [70] uses Wasserstein distance to measure the similarity of local distributions, and [17–19, 46, 55] optimize over distributions over a set of adversarial perturbations for a single image.

The second category is to pre-process the generated adversarial examples before training instead of directly using the adversarial examples generated by attack algorithms. Notable examples include label smoothing [11, 71], which, instead of considering the adversarial instances $(\mathbf{s}', y)$ for the "hard" label y, it considers $(\mathbf{s}', \tilde{y})$, where $\tilde{y}$ is a "soft" label represented as a weighted sum of the hard label and the uniform distribution. This idea is further exploited in [51], which empirically studies how label smoothing works. Based on these, **AVMixup** [41, 86] defines a virtual sample in the adversarial direction and extends the training distribution with soft labels via linear interpolation of the virtual sample and the clean sample. Specifically, it optimizes

$$J_{\text{avm}} = \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{D}} \left[\ell(f_\theta(\hat{\mathbf{s}}), \hat{\mathbf{y}}) \right], \tag{20}$$

where $\hat{\mathbf{s}} = \beta\mathbf{s} + (1-\beta)\gamma(\mathbf{s}'-\mathbf{s})$, $\hat{\mathbf{y}} = \beta\phi(\mathbf{y}, \lambda_1) + (1-\beta)\phi(\mathbf{y}, \lambda_2)$, β is drawn from the Beta distribution for each single $\mathbf{s}_i$, γ is the hyper-parameter to control the scale of adversarial virtual vector, $\mathbf{y}$ is the one-hot vector of y, $\phi(\cdot)$ is the label smoothing function [71], and λ_1 and λ_2 are hyper-parameters to control the smoothing degree. Other than label smoothing, [84] generates adversarial examples by perturbing the local neighborhoods in an unsupervised fashion.

These two categories follow the min-max formalism and only adapt its components. **AWP** [81] adapts the inner maximization to take one additional maximization to find a weight perturbation based on the generated adversarial examples. The outer minimization is then based on the perturbed weights [15] to minimize the loss induced by the adversarial examples. Specifically, it is to optimize the double-perturbation adversarial training problem

$$J_{\text{awp}} = \max_{\mathbf{V}\in\mathcal{V}} \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{D}} \max_{||\mathbf{s}'-\mathbf{s}||\leq\epsilon} \ell(f_{\mathbf{W}+\mathbf{V}}(\mathbf{s}')), \tag{21}$$

where $\mathcal{V}$ is a feasible region for the parameter perturbation $\mathbf{V}$.

Another thread of related work is on the PAC-Bayesian framework, a well-known theoretical tool to bound the generalization error of machine learning models [26, 38–40, 47, 48, 58, 79]. Recently, it is also widely developed in various aspects for both traditional machine learning models and deep neural networks [1, 8, 20, 21, 23, 30, 31, 42, 59, 62, 73].

7. Conclusion

This work addresses an oversight in the adversarial training literature by arguing that the second-order statistics of weights need to be systematically considered. Through theoretical study (updating the PAC-Bayesian framework), algorithmic development (efficient estimation of weight correlation matrix, effective training with S^{2}O), and extensive experiments, we show that the consideration of second-order statistics of weights can improve the robustness and generalization over not only the vanilla adversarial training but also the state-of-the-art adversarial training methods.

Acknowledgment. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 956123, and from UK Dstl under project SOLITUDE. GJ and XH are also partially supported by the UK EPSRC under projects [EP/R026173/1, EP/T026995/1].

References

[1] Pierre Alquier, James Ridgway, and Nicolas Chopin. On the properties of variational approximations of gibbs posteriors. *The Journal of Machine Learning Research*, 17(1):8374–8414, 2016. 8

[2] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision*, pages 484–501. Springer, 2020. 6

[3] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283. PMLR, 2018. 1

[4] Yang Bai, Yan Feng, Yisen Wang, Tao Dai, Shu-Tao Xia, and Yong Jiang. Hilbert-based generative defense for adversarial examples. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4784–4793, 2019. 1

[5] Afonso S Bandeira and March T Boedihardjo. The spectral norm of gaussian matrices with correlated entries. *arXiv preprint arXiv:2104.02662*, 2021. 12

[6] Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. Can we gain more from orthogonality regularizations in training deep networks? *Advances in Neural Information Processing Systems*, 31:4261–4271, 2018. 2

[7] Roberto Battiti. First-and second-order methods for learning: between steepest descent and newton's method. *Neural computation*, 4(2):141–166, 1992. 5

[8] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622. PMLR, 2015. 8

[9] Aleksandar Botev, Hippolyt Ritter, and David Barber. Practical gauss-newton optimisation for deep learning. *International Conference on Machine Learning*, 2017. 2, 4, 5

[10] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017. 6

[11] Tianlong Chen, Zhenyu Zhang, Sijia Liu, Shiyu Chang, and Zhangyang Wang. Robust overfitting may be mitigated by properly learned smoothening. In *International Conference on Learning Representations*, 2020. 8

[12] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pages 2196–2205. PMLR, 2020. 6

[13] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, pages 2206–2216. PMLR, 2020. 6

[14] Jiequan Cui, Shu Liu, Liwei Wang, and Jiaya Jia. Learnable boundary guided adversarial training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15721–15730, 2021. 7

[15] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 8

[16] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. *arXiv preprint arXiv:1812.02637*, 2018. 3, 8

[17] Yinpeng Dong, Zhijie Deng, Tianyu Pang, Hang Su, and Jun Zhu. Adversarial distributional training for robust deep learning. *arXiv preprint arXiv:2002.05999*, 2020. 8

[18] Yinpeng Dong, Zhijie Deng, Tianyu Pang, Jun Zhu, and Hang Su. Adversarial distributional training for robust deep learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8270–8283. Curran Associates, Inc., 2020. 3, 8

[19] Yinpeng Dong, Qi-An Fu, Xiao Yang, Tianyu Pang, Hang Su, Zihao Xiao, and Jun Zhu. Benchmarking adversarial robustness on image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 321–331, 2020. 8

[20] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. *arXiv preprint arXiv:1506.02629*, 2015. 8

[21] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Leon Roth. Preserving statistical validity in adaptive data analysis. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 117–126, 2015. 8

[22] Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017. 1, 2

[23] Gintare Karolina Dziugaite and Daniel M Roy. Data-dependent pac-bayes priors via differential privacy. *arXiv preprint arXiv:1802.09583*, 2018. 8

[24] Logan Engstrom, Andrew Ilyas, and Anish Athalye. Evaluating and understanding the robustness of adversarial logit pairing. *arXiv preprint arXiv:1807.10272*, 2018. 3, 8

[25] Farzan Farnia, Jesse M Zhang, and David Tse. Generalizable adversarial training via spectral normalization. In *ICLR*, 2019. 1, 3, 12, 13

[26] Pascal Germain, Alexandre Lacasse, François Laviolette, and Mario Marchand. Pac-bayesian learning of linear classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 353–360, 2009. 8

[27] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1, 3, 6

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[29] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. *International Conference on Learning Representations*, 2020. 4

[30] Gaojie Jin, Xinping Yi, Pengfei Yang, Lijun Zhang, Sven Schewe, and Xiaowei Huang. Weight expansion: A new perspective on dropout and generalization. *arXiv preprint arXiv:2201.09209*, 2022. 8

[31] Gaojie Jin, Xinping Yi, Liang Zhang, Lijun Zhang, Sven Schewe, and Xiaowei Huang. How does weight correlation affect generalisation ability of deep neural networks? *Advances in Neural Information Processing Systems*, 33, 2020. 8

[32] Bahman Kalantari and Thomas H Pate. A determinantal lower bound. *Linear Algebra and Its Applications*, 326(1-3):151–159, 2001. 4

[33] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018. 3, 8

[34] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *International Conference on Learning Representations*, 2017. 4, 13

[35] Allen Knutson and Terence Tao. Honeycombs and sums of hermitian matrices. *Notices Amer. Math. Soc*, 48(2), 2001. 4

[36] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[37] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2016. 3

[38] John Langford and Rich Caruana. (not) bounding the true error. *Advances in Neural Information Processing Systems*, 2:809–816, 2002. 8

[39] John Langford and Matthias Seeger. Bounds for averaging classifiers. 2001. 8

[40] John Langford and John Shawe-Taylor. Pac-bayes & margins. *Advances in neural information processing systems*, pages 439–446, 2003. 8

[41] Saehyung Lee, Hyungyu Lee, and Sungroh Yoon. Adversarial vertex mixup: Toward better adversarially robust generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 272–281, 2020. 2, 3, 8

[42] Gaël Letarte, Pascal Germain, Benjamin Guedj, and François Laviolette. Dichotomize and generalize: Pac-bayesian binary activated deep neural networks. *arXiv preprint arXiv:1905.10259*, 2019. 8

[43] Avery Ma, Fartash Faghri, Nicolas Papernot, and Amir-massoud Farahmand. Soar: Second-order adversarial regularization. *arXiv preprint arXiv:2004.01832*, 2020. 2, 7

[44] Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 2018. 1

[45] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 1, 6

[46] Chengzhi Mao, Ziyuan Zhong, Junfeng Yang, Carl Vondrick, and Baishakhi Ray. Metric learning for adversarial robustness. *arXiv preprint arXiv:1909.00900*, 2019. 8

[47] Andreas Maurer. A note on the pac bayesian theorem. *arXiv preprint cs/0411099*, 2004. 8

[48] David A McAllester. Pac-bayesian model averaging. In *Proceedings of the twelfth annual conference on Computational learning theory*, pages 164–170, 1999. 1, 2, 3, 8

[49] Dmytro Mishkin and Jiri Matas. All you need is a good init. *arXiv preprint arXiv:1511.06422*, 2015. 2

[50] Ronghui Mu, Wenjie Ruan, Leandro Soriano Marcolino, and Qiang Ni. Sparse adversarial video attacks with spatial transformations. *arXiv preprint arXiv:2111.05468*, 2021. 1

[51] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? *arXiv preprint arXiv:1906.02629*, 2019. 8

[52] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 5

[53] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. *arXiv preprint arXiv:1706.08947*, 2017. 3

[54] Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1707.09564*, 2017. 3, 12

[55] Tianyu Pang, Xiao Yang, Yinpeng Dong, Hang Su, and Jun Zhu. Bag of tricks for adversarial training. *arXiv preprint arXiv:2010.00467*, 2020. 8

[56] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017. 6

[57] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*, pages 582–597. IEEE, 2016. 1

[58] Emilio Parrado-Hernández, Amiran Ambroladze, John Shawe-Taylor, and Shiliang Sun. Pac-bayes bounds with data dependent priors. *The Journal of Machine Learning Research*, 13(1):3507–3531, 2012. 8

[59] Marıa Pérez-Ortiz, Omar Rivasplata, John Shawe-Taylor, and Csaba Szepesvári. Tighter risk certificates for neural networks. *Journal of Machine Learning Research*, 22, 2021. 8

[60] Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pages 8093–8104. PMLR, 2020. 5

[61] Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. *International Conference on Learning Representations*, 2018. 2, 4, 5

[62] Omar Rivasplata, Vikram M Tankasali, and Csaba Szepesvari. Pac-bayes with backprop. *arXiv preprint arXiv:1908.07380*, 2019. 8

[63] Håvard Rue, Sara Martino, and Nicolas Chopin. Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the*

royal statistical society: Series b (statistical methodology), 71(2):319–392, 2009. 4

[64] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 7

[65] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013. 2

[66] Adrian J Shepherd. *Second-order methods for neural networks: Fast and reliable training methods for multi-layer perceptrons*. Springer Science & Business Media, 2012. 5

[67] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 7

[68] Sahil Singla and Soheil Feizi. Second-order provable defenses against adversarial attacks. In *International Conference on Machine Learning*, pages 8981–8991. PMLR, 2020. 2

[69] Aman Sinha, Hongseok Namkoong, Riccardo Volpi, and John Duchi. Certifying some distributional robustness with principled adversarial training, 2020. 3

[70] Matthew Staib and Stefanie Jegelka. Distributionally robust deep learning as a generalization of adversarial training. *NIPS workshop on Machine Learning and Computer Security*, 2017. 3, 8

[71] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 8

[72] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1

[73] Niklas Thiemann, Christian Igel, Olivier Wintenberger, and Yevgeny Seldin. A strongly quasiconvex pac-bayesian bound. In *International Conference on Algorithmic Learning Theory*, pages 466–492. PMLR, 2017. 8

[74] Luke Tierney and Joseph B Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the american statistical association*, 81(393):82–86, 1986. 4

[75] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017. 1

[76] Joel A Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434, 2012. 12

[77] Theodoros Tsiligkaridis and Jay Roberts. Second order optimization for adversarial robustness and interpretability. *arXiv preprint arXiv:2009.04923*, 2020. 2

[78] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*, 2019. 2, 3, 8

[79] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011. 8

[80] Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, and Xingjun Ma. Skip connections matter: On the transferability of adversarial examples generated with resnets. *arXiv preprint arXiv:2002.05990*, 2020. 1

[81] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33, 2020. 2, 7, 8

[82] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017. 1

[83] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 5

[84] Haichao Zhang and Jianyu Wang. Defense against adversarial attacks using feature scattering-based adversarial training. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 8

[85] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482. PMLR, 2019. 2, 3, 6, 8

[86] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? *arXiv preprint arXiv:2010.04819*, 2020. 3, 8

[87] Tianhang Zheng, Changyou Chen, and Kui Ren. Distributionally adversarial attack. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):2253–2260, Jul. 2019. 3, 8