Improving Robust Generalization by Direct PAC-Bayesian Bound Minimization

Zifan Wang[1]* Nan Ding[2]† Tomer Levinboim[2] Xi Chen[2] Radu Soricut[2]
Carnegie Mellon University[1] Google Research[2]
zifan@cmu.edu, {dingnan, tomerl, chillxichen, rsoricut}@google.com

Abstract

Recent research in robust optimization has shown an overfitting-like phenomenon in which models trained against adversarial attacks exhibit higher robustness on the training set compared to the test set. Although previous work provided theoretical explanations for this phenomenon using a robust PAC-Bayesian bound over the adversarial test error, related algorithmic derivations are at best only loosely connected to this bound, which implies that there is still a gap between their empirical success and our understanding of adversarial robustness theory. To close this gap, in this paper we consider a different form of the robust PAC-Bayesian bound and directly minimize it with respect to the model posterior. The derivation of the optimal solution connects PAC-Bayesian learning to the geometry of the robust loss surface through a Trace of Hessian (TrH) regularizer that measures the surface flatness. In practice, we restrict the TrH regularizer to the top layer only, which results in an analytical solution to the bound whose computational cost does not depend on the network depth. Finally, we evaluate our TrH regularization approach over CIFAR-10/100 and ImageNet using Vision Transformers (ViT) and compare against baseline adversarial robustness algorithms. Experimental results show that TrH regularization leads to improved ViT robustness that either matches or surpasses previous state-of-the-art approaches while at the same time requires less memory and computational cost.

1. Introduction

Despite their success in a wide range of fields and tasks, deep learning models still remain susceptible to manipulating their outputs by even tiny perturbations to the input [6, 7, 10, 19, 30, 32, 43, 47]. Several lines of work have focused on developing robust training techniques against such adversarial attacks [8, 20, 31, 32, 36, 41, 46, 48, 54]. Importantly, Rice et al. [38] observe a robust overfitting phenomenon, referred to as the *robust generalization gap*, in which a robustly-trained classifier shows much higher accuracy on adversarial examples from the training set, compared to lower accuracy on the test set. Indeed, several technical approaches have been developed that could alleviate this overfitting phenomenon, including ℓ_2 weight regularization, early stopping [38], label smoothing, data augmentation [51, 53], using synthetic data [21] and etc.

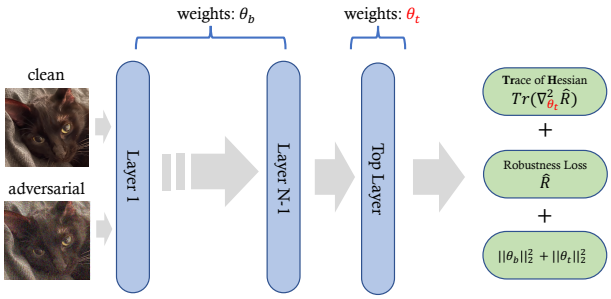

Figure 1. We propose Trace of Hessian (TrH) regularization for training adversarially robust models. In addition to an ordinary robust loss (e.g., TRADES [54]), we regularize the TrH of the loss with respect to the weights of the top layer to encourage flatness. The training objective is the result of direct PAC-Bayesian bound minimization in Theorem 3.

According to learning theory, the phenomenon of overfitting can be characterized by a PAC-Bayesian bound [4, 9, 18, 34, 42] which upper-bounds the expected performance of a random classifier over the underlying data distribution by its performance on a finite set of training points plus some additional terms. Although several prior works [21, 24, 26, 49] have built upon insights from the PAC-Bayesian bound, none attempted to directly minimize the upper bound, likely due to the fact that the minimization of their forms of the PAC-Bayesian bound do not have an analytical solution.

In this paper, we rely on a different form of the PAC-

*Work done in Google.
†Corresponding author

Bayesian bound [18], which can be readily optimized using a Gibbs distribution [16] with which we derive a second-order upper bound over the robust test loss. Interestingly, the resulting bound consists of a regularization term that involves *Trace of Hessian* (TrH) [12] of the network weights, a well-known measure of the loss-surface flatness.

For practical reasons, we limit TrH regularization to the top layer of the network only because computing a Hessian matrix and its trace for the entire network is too costly. We further derive the analytical expression of the top-layer TrH and show both theoretically and empirically that top-layer TrH regularization has a similar effect as regularizing the entire network. The resulting TrH regularization (illustrated in Figure 1) is less expensive and more memory efficient compared to other competitive methods [21, 24, 26, 49].

In summary, our contributions are as follows: (1) We provide a PAC-Bayesian upper-bound over the robust test loss and show how to directly minimize it (Theorem 3). To the best of our knowledge, this has not been done by prior work. Our bound includes a TrH term which encourages the model parameters to converge at a flat area of the loss function; (2) Taking efficiency into consideration, we restrict the TrH regularization to the top layer only (Algorithm 1) and show that it is an implicit but empirically effective regularization on the TrH of each internal layer (Theorem 4 and Example 1); and (3) Finally, we conduct experiments with our new TrH regularization and compare the results to several baselines using Vision Transformers [14]. On CIFAR-10/100, our method consistently matches or beats the best baseline. On ImageNet, we report a significant gain (+2.7%) in robust accuracy compared to the best baseline and establish a new state-of-the-art result of 48.9%.

2. Background

We begin by introducing our notation and basic definitions. Let $g_\theta(x)$ denote the output logits of a network θ on an input $x \in \mathcal{X}$. We denote the predicted label by $F_\theta(x) = \arg\max g_\theta(x) \in \mathcal{Y}$ and the softmax output of g_θ by $s(g_\theta)$. To train the network, we sample a dataset $D^m \stackrel{\text{def}}{=} \{(x_i, y_i)\}_{i=0}^{m-1}$ containing m i.i.d examples from a distribution $\mathcal{D}$, so that the test and training losses are

$$L(\theta, \mathcal{D}) \stackrel{\text{def}}{=} \mathbb{E}_{(x,y)\sim\mathcal{D}}\, l((x,y), g_\theta),$$
$$\text{and}\quad \hat{L}(\theta, D^m) \stackrel{\text{def}}{=} \frac{1}{m}\sum_{(x,y)\in D^m}^{m} l((x,y), g_\theta),$$

respectively and l denotes the loss function. Ideally, the classification error of g_θ should be represented by the 0-1 loss $l((x,y), g_\theta) = \mathbb{I}[F_\theta(x) \neq y]$. However, since the 0-1 loss is infeasible to minimize and inferior for generalization, it is common to use surrogate losses such as the Cross-Entropy loss $\text{CE}((x,y), g_\theta) = -\log\{s(g_\theta(x))\}_y$ for training instead, where $\{v\}_j$ denotes the j-th element of the vector v. Finally, we denote the KL-divergence between two distributions $\mathcal{P}, \mathcal{Q}$ as $\text{KL}(\mathcal{P}||\mathcal{Q})$. For any twice-differentiable function L, we denote its Jacobian and Hessian matrix as ∇L and $\nabla^2 L$ respectively.

2.1. Adversarial Robustness

The goal of *adversarial robustness* is to train a deep network that is robust against ℓ_p norm-bounded noises. In this case, the test and training loss functions are replaced by their respective robust counterparts – $R(\theta, \mathcal{D})$ on the test set and $\hat{R}(\theta, D^m)$ on the training set, such that

$$R(\theta, \mathcal{D}) \stackrel{\text{def}}{=} \mathbb{E}_{(x,y)\sim\mathcal{D}} \max_{||\epsilon||_p \leq \delta} l((x+\epsilon, y), g_\theta),$$
$$\text{and}\ \hat{R}(\theta, D^m) \stackrel{\text{def}}{=} \frac{1}{m}\sum_{(x,y)\in D^m}^{m} \max_{||\epsilon||_p \leq \delta} l((x+\epsilon, y), g_\theta),$$

where δ is the maximum noise budget. Various methods have been proposed to improve adversarial robustness. In this paper, we focus on the following two well-known defenses: *adversarial training* [32], denote AT (Definition 1), and TRADES [54] (Definition 2). AT is a classical method in enforcing robustness; while TRADES achieves state-of-the-art robust accuracy on CIFAR-10 and ImageNet [21].

Definition 1 (Adversarial Training (AT) [32]) *Given a network with parameter θ, a training set D^m, the bound of noise δ and norm $||\cdot||_p$, AT minimizes the following objective w.r.t θ,*

$$\hat{R}_{AT}(\theta, D^m) \stackrel{def}{=} \frac{1}{m}\sum_{(x,y)\in D^m} \max_{||\epsilon||_p \leq \delta} \text{CE}((x+\epsilon, y), g_\theta).$$

which considers the adversarial sample $x + \epsilon$ in a δ-ball around each input, instead of the input itself.

Definition 2 (TRADES [54]) *Given the same assumptions as in Definition 1, TRADES minimizes the following objective w.r.t θ:*

$$\hat{R}_T(\theta, D^m) \stackrel{def}{=} \frac{1}{m}\sum_{(x,y)\in D^m} \Big[\text{CE}((x,y), g_\theta) + \lambda_t \cdot \max_{||\epsilon||_p \leq \delta} \text{KLloss}((x, x+\epsilon), g_\theta)\Big]$$
where $\text{KLloss}((x, x+\epsilon), g_\theta) = \text{KL}(s(g_\theta(x))||s(g_\theta(x+\epsilon)))$,

λ_t is a penalty hyper-parameter, which balances the clean accuracy and the robustness.

Intuitively, TRADES minimizes the cross-entropy loss, while also regularizing for prediction smoothness in an δ-ball around each input.

2.2. Generalizing Robustness

Rice et al. [38] observe that the AT and TRADES methods may suffer from severe overfitting. Namely, the model exhibits higher robustness on the training data compared to the test data. A number of traditional strategies have been applied to alleviate the overfitting problem in robust training, such as ℓ_2 weight regularization, early stop [38], label smoothing, data augmentation [51, 53] and using synthetic data [21]. Below we describe popular and recent methods that were designed specifically for adversarial training.

Stochastic Weight Averaging (SWA). Izmailov et al. [24] introduces SWA for improved generalization performance in standard training. SWA maintains an exponential running average $\theta_{\texttt{avg}}$ of the model parameters θ at each training iteration and uses the running average $\theta_{\texttt{avg}}$ for inference. Namely,

$$\theta^{(t+1)} \leftarrow \texttt{SGD}(\theta^{(t)}); \quad \theta_{\texttt{avg}} \leftarrow \alpha\theta_{\texttt{avg}} + (1-\alpha)\theta^{(t+1)}$$

where α is commonly set to 0.995. Recent work has also shown that SWA helps generalization in the robust training [21] scenario. Note that maintaining the running average requires additional memory resources.

Adversarial Weight Perturbation (AWP). Similar to Sharpness-Aware Minimization [15], Wu et al. [49] encourages model robustness by penalizing against the most offending local weight perturbation to the model. Specifically, given a robust loss $\hat{R}_*(\theta, D^m)$, AWP minimizes the following objective w.r.t θ:

$$\max_{\psi(\xi)\leq\delta_{awp}} \hat{R}_*(\theta+\xi, D^m) \tag{1}$$

where * denotes either AT or T, ψ measures the amount of noise ξ (e.g. ℓ_2 norm) and δ_{awp} denotes the noise total budget. Note that solving the inner maximization problem requires at least one additional gradient ascent step in the parameter space.

Second-Order Statistic (S2O). Jin et al. [26] improve the robustness generalization by regularizing the statistical correlations between the weights of the network. Furthermore, they provide a second-order and data-dependent approximation A_θ of the weight correlation matrix. During training, S2O minimizes the following objective w.r.t θ:

$$\hat{R}_*(\theta, D^m) + \alpha||A_\theta||_F \tag{2}$$

where $\alpha > 0$ is a hyper-parameter. Computing A_θ involves computing the per-instance self-Kronecker product of the logit outputs for the clean and adversarial inputs.

2.3. PAC-Bayesian Theory for Robust Training

PAC-Bayesian theory [4,9,18,34,35,42] provides a foundation for deriving an upper bound of the generalization gap between the training and test error. Both AWP and S2O plug the robust loss into the classical bound [35] to obtain the following form:

Theorem 1 (Square Root-Form PAC-Bayesian Bound [49]) *Given a prior distribution $\mathcal{P}$ on the weight θ of a network, any $\tau \in (0,1]$, a robustness loss $R(\theta,\mathcal{D})$ for a data distribution $\mathcal{D}$ and its empirical version $\hat{R}(\theta, D^m)$, for any posterior distribution $\mathcal{Q}$ of θ, the following inequality holds with a probability at least $1-\tau$,*

$$\mathbb{E}_{\theta\in\mathcal{Q}} R(\theta,\mathcal{D}) \leq \mathbb{E}_{\theta\in\mathcal{Q}} \hat{R}(\theta, D^m) + 4\sqrt{\frac{1}{m}\mathrm{KL}(\mathcal{Q}||\mathcal{P}) + \log\frac{2m}{\tau}}. \tag{3}$$

Generally speaking, the goal of PAC-Bayesian learning is to optimize $\mathcal{Q}$ on the RHS of Eq. 3 so as to obtain a tight upper bound on the test error (LHS). However, directly optimizing Eq. 3 over $\mathcal{Q}$ is difficult because of the square root term. Instead of optimizing the RHS, AWP replaces it with $\max_\xi \hat{R}(\theta+\xi, D^m)$ plus a constant upper bound on the square root term, which is only loosely connected to the bound. On the other hand, S2O assumes $\mathcal{P}$ and $\mathcal{Q}$ as non-spherical Gaussian, i.e. $\mathcal{P} = \mathcal{N}(0, \Sigma_p)$, $\mathcal{Q} = \mathcal{N}(\theta, \Sigma_q)$ and show that the square root bound increases as the Frobenius norm and singular value of Σ_q increase. To overcome the complexity of the square root term, they approximate it with $\|A_\theta\|_F$ as in Eq. 2.

3. Direct PAC-Bayesian Bound Minimization

As mentioned earlier, neither AWP [49] nor S2O [26] minimize the PAC-Bayesian bound directly, likely due to the complicating square-root term in Eq.3. However, Eq.3 is only one instance out of PAC-Bayesian bounds defined in Germain et al. [18]. According to their framework, there is also a *linear-form* of the bound related to Bayesian inference which can be analytically minimized with the aid of a Gibbs distribution [12, 17, 40]. We leverage this linear-form of the PAC-Bayesian bound, and adapt it for bounding the robustness gap:

Theorem 2 (Linear-Form PAC-Bayesian Bound [17]) *Under the same assumptions as in Theorem 1, for any $\beta > 0$, with a probability at least $1-\tau$,*

$$\mathbb{E}_{\theta\in\mathcal{Q}} R(\theta,\mathcal{D}) \leq \mathbb{E}_{\theta\in\mathcal{Q}} \hat{R}(\theta, D^m) + \frac{1}{\beta}\mathrm{KL}(\mathcal{Q}||\mathcal{P}) + C(\tau,\beta,m), \tag{4}$$

where $C(\tau,\beta,m)$ is a function independent of $\mathcal{Q}$.

Here β is a hyper-parameter that balances the three terms on the RHS. A common choice is $\beta \propto m$, i.e. the size of the dataset [13, 17]. Recently, [12] proposed to set $\mathcal{P}$ and $\mathcal{Q}$ to univariate Gaussians and used a second-order approximation to estimate the PAC-Bayesian bound. Applying a similar technique, we introduce Theorem 3 (see Appendix. A for the proof) which minimizes the RHS of Eq. 4.

Theorem 3 (Minimization of PAC-Bayesian Bound) *If $\mathcal{P} = \mathcal{N}(\mathbf{0}, \sigma_0^2)$, and $\mathcal{Q}$ is also a product of univariate Gaussian distributions, then the minimum of Eq. 4 w.r.t $\mathcal{Q}$ can be bounded by*

$$\begin{aligned}
&\min_{\mathcal{Q}} \mathbb{E}_{\theta\in\mathcal{Q}} R(\theta, \mathcal{D}) \\
\leq &\min_{\mathcal{Q}} \{\mathbb{E}_{\theta\in\mathcal{Q}} \hat{R}(\theta, D^m) + \frac{1}{\beta}\mathrm{KL}(\mathcal{Q}||\mathcal{P})\} + C(\tau, \beta, m) \\
= &\min_{\theta} \{\hat{R}(\theta, D^m) + \frac{||\theta||_2^2}{2\beta\sigma_0^2} + \frac{\sigma_0^2}{2}\mathrm{Tr}(\nabla_\theta^2 \hat{R}(\theta, D^m))\} \\
&+ C(\tau, \beta, m) + O(\sigma_0^4). \qquad (5)
\end{aligned}$$

Assuming that $\sigma_0^2 = 1/d$ is the variance of the initial weight matrices where d is the input feature dimension, then $O(\sigma_0^4) = O(d^{-2})$. Theorem 3 considerably simplifies the optimization problem from one over the space of probability density functions $\mathcal{Q}$ to one over model weights θ. Moreover, the resulting bound (RHS of Eq. 5) contains the Trace of Hessian (TrH) term $\mathrm{Tr}(\nabla_\theta^2 \hat{R}(\theta, D^m))$ of the robust loss, which sums the loss surface curvatures in all directions. For a convex loss, the Hessian is positive semi-definite (PSD) in which case trace minimization leads θ to a flatter region of the loss surface. Although in general, our Hessian is not PSD, empirically we find that its largest eigenvalue has a much larger magnitude than the least ones [3] and that trace minimization correlates with minimizing the standard deviation of the eigenvalues (see Figure 5 in the appendix), so that the overall curvature is effectively decreases.

The RHS of Eq. 5 in Theorem 3 provides a training objective function that bounds the optimal test error up to a constant. However, evaluating and minimizing the bound in Eq. 5 requires computing the Trace of Hessian (TrH) term. Unfortunately, computing the Hessian directly by applying auto-differentiation twice is infeasible for a large deep network. Another solution is to apply Hutchinson's method [5] to randomly approximate the TrH, however, the variance of the resulting estimator is too high and it ends up being too noisy for training in practice.

Instead, we propose to restrict TrH regularization to the top layer of the deep network only. Although this restriction brings approximation error to the bound, it has two benefits. First, the TrH on the top layer has simple analytical expressions (Section 3.1). Second, we show that the top-layer TrH regularization effectively regularizes the TrH of the entire network as well (Section 3.2).

3.1. Top-Layer TrH regularizers for AT & TRADES

The network parameter θ can be decomposed into $\{\theta_t, \theta_b\}$ where θ_t denotes the weights of the top layer and θ_b denotes the weights of the remaining (bottom) network. In this case, the logits can be expressed as $g_\theta(x) = \theta_t^\top f_{\theta_b}(x)$, where $f_{\theta_b}(x)$ is the penultimate layer feature. Furthermore, we define $h(x, \theta) = s(g_\theta(x)) - s(g_\theta(x))^2$. We present the analytical forms of the top-layer TrH w.r.t. the AT and TRADES losses in Proposition 1 and 2 (additional examples of other adversarial losses are provided in Appendix B).

Proposition 1 (TrH for AT) *Given a training dataset D^m and the adversarial input example x' for each example x, the top-layer TrH of the AT loss (Definition 1) is equal to*

$$\mathrm{Tr}(\nabla_{\theta_t}^2 \hat{R}_{\scriptscriptstyle AT}(\theta, D^m)) = \frac{1}{m}\sum_{(x,y)\in D^m} \mathrm{TrH}_{\scriptscriptstyle AT}(x'; \theta),$$
$$\textit{where} \quad \mathrm{TrH}_{\scriptscriptstyle AT}(x'; \theta) = ||f_{\theta_b}(x')||_2^2 \cdot \mathbf{1}^\top h(x', \theta).$$

Proposition 2 (TrH for TRADES) *Under the same assumption in Proposition 1, the top-layer TrH of the TRADES loss (Definition 2) is equal to*

$$\mathrm{Tr}(\nabla_{\theta_t}^2 \hat{R}_{\scriptscriptstyle T}(\theta, D^m)) = \frac{1}{m}\sum_{(x,y)\in D^m} \mathrm{TrH}_{\scriptscriptstyle T}(x, x'; \lambda_t, \theta),$$
$$\begin{aligned}
\textit{where,} \quad \mathrm{TrH}_{\scriptscriptstyle T}(x, x'; \lambda_t, \theta) &= ||f_{\theta_b}(x)||_2^2 \cdot \mathbf{1}^\top h(x, \theta) \\
&+ \lambda_t ||f_{\theta_b}(x')||_2^2 \cdot \mathbf{1}^\top h(x', \theta). \qquad (6)
\end{aligned}$$

To obtain Eq. 6, we stopped the gradient on the KLloss in Definition 2 with respect to the clean logits $g_\theta(x)$, because we find it is more stable for training, otherwise, there would be an additional regularization term $G(x, x', \theta)$ in the $\mathrm{TrH}_{\scriptscriptstyle T}$ (see more details in the Appendix A).

With Proposition 1 and 2, we now present the complete training objective in Algorithm 1. Notice that, to simplify hyper-parameter notations, we re-parameterize $\gamma \overset{\text{def}}{=} 1/2\beta\sigma_0^2$ and $\lambda \overset{\text{def}}{=} \sigma_0^2/2$ in the algorithm.

3.2. The effect of Top-Layer TrH Regularization

Although the TrH regularization is restricted to the top layer only, the gradient back-propagates to the entire network through the penultimate layer features. This motivates the following research question:

> *What effect does top-layer TrH regularization have on the TrH of the entire network?*

To answer this question, we take a two-step approach. First, we experiment with a small 3-layer network on the synthetic Two-Moon dataset (see Appendix C for a visualization of the dataset), provided that computing the full

Algorithm 1: TrH Regularization for Training Adversarial Robust Network

Hyper-parameters: The ℓ_2-norm penalty for weights γ, the TRADES penalty λ_t and the TrH penalty λ.
Inputs: A batch B of examples, a `loss_type` ='AT' or 'TRADES', the noise budget δ, and a network $g_\theta = \theta_t^\top f_{\theta_b}(x)$.
Output: The training objective at the current iteration.

$R \leftarrow 0$
foreach $(x, y) \in B$ **do**
  $x' \leftarrow$ PGD_InnerLoop$(x, y, \delta,$ loss_type)
  $z' \leftarrow f_{\theta_b}(x'), \quad h' \leftarrow \mathrm{s}(\theta_t^\top z') - s(\theta_t^\top z')^2$; // s is the softmax function. This step computes the penultimate feature z' and the softmax gradient h' on the adversarial input x'.
  if *loss_type* == '*AT*' **then**
    $R \leftarrow R + \mathrm{CE}((x', y), g_\theta) + \lambda(||z'||_2^2 \cdot \mathbf{1}^\top h')$;
  else if *loss_type* == '*TRADES*' **then**
    $z \leftarrow f_{\theta_b}(x), \quad h \leftarrow \mathrm{s}(\theta_t^\top z) - s(\theta_t^\top z)^2$; // TRADES needs the z and h for the clean input x.
    $R \leftarrow R + \mathrm{CE}((x, y), g_\theta) + \lambda_t \mathrm{KL}(s(\theta_t^\top z)||s(\theta_t^\top z')) + \lambda(||z||_2^2 \cdot \mathbf{1}^\top h + \lambda_t||z'||_2^2 \cdot \mathbf{1}^\top h')$;
  end
end
return $\frac{1}{|B|} R + \gamma||\theta||_2^2$

network Hessian is inexpensive. Subsequently, we provide a theorem that can be used to bound the Trace of Hessian of the full network with that of the top layer only.

Example 1 (Two Moons) *We use* 500 *training examples, where* $x \in \mathbb{R}^2$ *and* $y \in \{0, 1\}$ *from the Two Moon dataset. We train a 3-layer fully-connected network with Dense(100)-ReLU-Dense(100)-ReLU-Dense(2) with the AT loss under three settings:*

- *Standard: no regularization.*
- *Top: Regularizing the TrH of the top layer only.*
- *Full: Regularizing the TrH of the entire network.*

Throughout the training process (x-axis), we evaluate the TrH of the entire *network (y-axis) with a numerical approach (twice differentiation) and plot the results in Figure 2.*

Figure 2 empirically shows that regularizing the TrH of the top layer (orange) indirectly reduces the TrH of the entire network and is nearly as effective as regularizing the entire network (green) after the 60-th epoch.

Next, we provide a theoretical justification for the impact of top-layer TrH regularization on the layers below. Intuitively, our Theorem 4, establishes an inductive relation between the TrH of the consecutive layers in a feedforward neural network with ReLU activation and CE loss, a common building block in AT and TRADES training.

Theorem 4 (Inductive Relation in TrH) *Suppose* g_θ *is a feed-forward network with ReLU activation. For the* i*-th layer, let* $W^{(i)}$ *be its weight and* $\mathcal{I}_x^{(i)}$ *be its input evaluated*

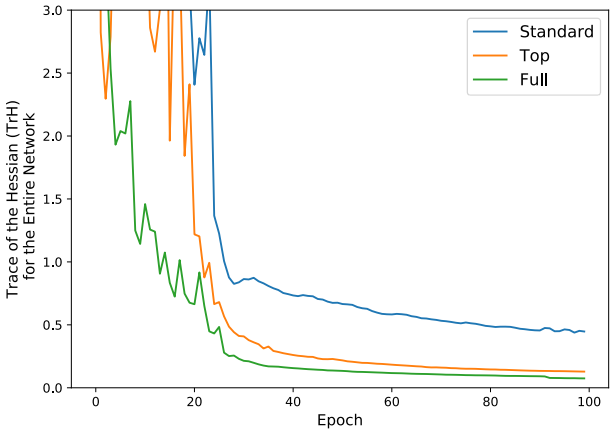

Figure 2. Trace of the Hessian (TrH) of the entire network (y-axis) over training epochs (x-axis) with (1) no regularization (Standard, in blue); (2) top layer TrH regularization (Top, in orange); and (3) regularization the TrH of all layers (Full, in green). See Example 1 for the experimental setup.

at x*. Thus,* $\mathrm{TrH}_x^{\mathrm{CE}}(W^{(i-1)})$*, i.e. TrH evaluated at* x *using a CE loss w.r.t* $W^{(i-1)}$*, is equal to*

$$\mathrm{TrH}_x^{\mathrm{CE}}(W^{(i-1)}) = ||\{\mathcal{I}_x^{(i-1)}\}||_2^2 \sum_{k,d \in P^{(i)}} \{\mathcal{H}^{(i)}\}_{k,d}$$

$$\text{where } \{\mathcal{H}^{(i)}\}_{k,d_i} \stackrel{def}{=} \left[\frac{\partial\{g_\theta(x)\}_k}{\partial\{\mathcal{I}_x^{(i)}\}_{d_i}}\right]^2 \cdot \{h(x,\theta)\}_k$$

$$P^{(i)} \stackrel{def}{=} \{d_i | \{\mathcal{I}_x^{(i)}\}_{d_i} > 0\}$$

and the following inequality holds for any $\mathcal{H}^{(i)}, \mathcal{H}^{(i+1)}$*:*

$$\max_{k,d_i}\{\mathcal{H}^{(i)}\}_{k,d_i} \leq \max_{k,d_{i+1}}\{\mathcal{H}^{(i+1)}\}_{k,d_{i+1}} \cdot ||W^{(i)}||_1^2. \quad (7)$$

Put simply, the max-norm of the Hessian of each layer forms an upper bound for the max-norm of the layer below it (times a scalar). Therefore, regularizing the TrH of the top layer alone implicitly regularizes the Hessian of the layers below it. This effect explains the phenomenon observed in Example 1.

Furthermore, the operator norm $||W^{(i)}||_1^2$ of weights in Eq. 7 makes an interesting connection between TrH minimization and robust training. Roth et al. [39] show that robust training is an implicit way of regularizing the weight operator norm. Moreover, directly minimizing the operator norm is a commonly used technique in training certifiable robust networks [23, 31], a stronger version of the robustness guarantee targeted in this paper. Thus, the combination of robustness training and top-layer regularization results in an implicit regularization on TrH for internal layers.

4. Evaluation

In this section, we evaluate TrH regularization (Algorithm 1) over three image classification benchmarks including CIFAR-10/100 and ImageNet and compare against the robust training baselines described in Section 2.2.

4.1. Experiment Setup

Datasets and Threat Model. For CIFAR-10/100, we choose the standard $\epsilon = 0.031 (\approx 8/255)$ of the ℓ_∞ ball to bound the adversarial noise. In addition, Gowal et al. [21] has reported that using synthetic images from a DDPM [22] generator can improve the robustness accuracy for the original test set. Thus, we also add 1M DDPM images to CIFAR-10/100 (these images are publicly availabe [1]). For ImageNet, we choose $\epsilon = 3.0$ for the ℓ_2 and $\epsilon = 0.0157 (\approx 4/255)$ for the ℓ_∞ ball, respectively. We do not use extra data for ImageNet. The chosen ϵs follow the common choices in the literature [21, 26, 32, 49]. We report the test accuracy evaluated with benign images and adversarial images generated by AutoAttack [11] (AutoPGD+AutoDLR), which is widely used in the literature [21, 26, 49].

Network Architectures. We use Vision Transformer (ViT) [14] through all experiments because of its success on multiple tasks over existing architectures. Specifically, we report the results of ViT-L16, Hybrid-L16 for CIFAR-10/100 and ViT-B16, ViT-L16 for ImageNet (additional results and architecture details can be found in Appendix G). We initialize the weights from a pre-trained checkpoint on ImageNet21K [2]. Notice that the resolution of the CIFAR images (32×32) are too small to correctly align with the pre-trained kernel of the first layer. To address this issue, we down-sample the kernel of the input layer following the receipt proposed by Mahmood et al. [33].

Methods. We train the models using AT and TRADES losses, together with the following defenses: (1) base: no regularization; (2) AWP [49]; (3) SWA [24]; (4) S2O [26]; and (5) our Algorithm 1 (TrH). We are interested in the baselines defenses because of their connections to the PAC-Bayesian bound or the loss flatness. During training, the model was partitioned over four Google Cloud TPUv4 chips [27] for the base and TrH methods, and eight chips for AWP, S2O, and SWA as they consume more HBM memory.

Hyperparameter Selection. Training robust ViTs can be challenging due to a large number of hyper-parameters. We use two steps for hyperparameter selection. For the choice of common hyper-parameters, e.g. batch size, patch size, training iterations, and etc. (a full list is included in Appendix D), we first do a large grid search and select values that produce the best results on TRADES(`base`). This step is referred to as pre-tuning and is done per dataset.

After the first step, where the common hyper-parameters are locked, we tune and report the best results after trying different sets of method-specific hyper-parameters, e.g., the norm bound of weight noises in AWP and λ for TrH. Appendix E provides a full list of hyper-parameters we use to produce our main results in Table 1.

4.2. Main Results

In Table 1, we report the robust test accuracy using different types of defenses. The top results are highlighted in bold font. To measure the significance of an improvement, we calculate the maximum standard error (SE) [44] = $\sqrt{0.5 * (1 - 0.5)/m}$ (where m denotes the number of test examples) for each dataset. Thus, the accuracy of a certain model is regarded a *silver* result a_{silver}, if its SE interval overlaps with the that of the top result a_{top}. Namely, $a_{top} - SE \leq a_{silver} + SE$.

CIFAR-10/100. In CIFAR-10/100, a significant improvement is at least 1% according to SE. Therefore, based on Table 1, the performance differences among the methods are relatively small. Nevertheless, TrH attains either the top result (in 3 instances) or the silver result (in 5 instances) across all eight instances. In comparison, AWP is the second best with 3 top and 3 silver; S2O has 1 top and 4 silver; SWA has 3 silver; and the base has 1 top and 3 silver. Notably, Hybird-L16+TRADES(`TrH`) achieves the highest robust accuracy (34.1%) on CIFAR-100 which beats all other baselines by at least 2%.

$\ell_\infty(\delta = 8/255)$	ViT-L16				Hybrid-L16			
SE= ±0.5%	CIFAR-10		CIFAR-100		CIFAR-10		CIFAR-100	
Defense	Clean(%)	AA(%)	Clean (%)	AA(%)	Clean(%)	AA(%)	Clean (%)	AA(%)
AT(base)	87.4	60.8	64.3	31.7	88.0	**61.7**	64.2	31.8
AT(SWA)	88.0	61.7	62.5	31.7	86.5	57.2	63.6	30.8
AT(S2O)	87.1	60.2	64.9	31.7	88.7	61.6	64.3	31.8
AT(AWP)	88.5	**62.4**	63.3	32.6	88.5	61.5	63.9	**32.5**
AT(TrH)	87.0	61.7	62.5	**32.8**	88.0	61.0	63.8	32.4
TRADES(base)	85.2	60.3	62.0	31.4	85.9	60.8	61.3	30.6
TRADES(SWA)	85.9	60.7	61.5	32.0	84.3	59.4	62.3	29.9
TRADES(S2O)	87.4	**61.6**	62.4	31.6	87.2	61.5	65.0	31.9
TRADES(AWP)	85.3	60.8	63.0	**32.2**	84.5	59.8	61.4	32.1
TRADES(TrH)	86.4	61.4	62.0	32.1	87.4	**61.7**	66.4	**34.1**

ImageNet	ViT-B16				ViT-L16			
SE= ±0.2%	$\ell_\infty(\delta = 4/255)$		$\ell_2(\delta = 3.0)$		$\ell_\infty(\delta = 4/255)$		$\ell_2(\delta = 3.0)$	
Defense	Clean(%)	AA(%)	Clean (%)	AA(%)	Clean(%)	AA(%)	Clean (%)	AA(%)
AT(base)	72.2	39.0	71.0	39.3	75.4	44.1	73.9	43.5
AT(SWA)	72.3	40.0	71.5	39.8	75.2	44.3	74.3	43.8
AT(S2O)	72.0	39.0	71.4	39.3	75.4	46.2	76.3	46.4
AT(AWP)	71.3	39.5	70.5	39.3	74.5	44.0	73.8	44.0
AT(TrH)	71.3	**42.2**	71.6	**42.4**	75.8	**48.8**	74.2	**47.0**
TRADES(base)	69.2	39.3	67.8	39.7	77.8	40.9	77.1	41.5
TRADES(SWA)	69.5	39.7	68.1	40.0	72.7	44.5	71.1	44.2
TRADES(S2O)	70.6	39.3	69.8	39.7	73.7	43.6	72.5	44.0
TRADES(AWP)	65.8	38.6	64.5	38.3	68.5	43.2	67.2	42.7
TRADES(TrH)	65.3	**41.9**	66.4	**41.6**	70.1	**48.9**	68.9	**47.3**

Table 1. Clean: % of Top-1 correct predictions. AA: % of Top-1 correct predictions under AutoAttack. A max Standard Error (SE) [44] = $\sqrt{0.5 * (1 - 0.5)/m}$ (m as the number of test examples) is computed for each dataset. The best results appear in bold. Underlined results are those that fall within the SE range of the result and are regarded roughly equal to the best result.

ImageNet. On ImageNet, a significant improvement is at least 0.4% according to SE. We observe that TrH regularization outperforms the other baselines across the board. Using AT(TrH), we set a new *state-of-the-art* robust accuracy of 48.8%, at $\ell_\infty(4/255)$ and 47.0% at $\ell_2(3.0)$ with a ViT-L16 model. Specifically, in the case of ℓ_∞, 48.8% is an improvement of 4.7% compared to the basic setup AT(base). In TRADES(TrH), although the robust accuracy is even slightly higher than AT, the clean accuracy is lower. This is due to the choice of $\lambda_t = 6$, which strongly favors robustness over clean accuracy. As we reduce λ_t to 1, we obtain a better balance of (clean, AA) accuracy of TRADES(TrH) at (78.7, 46.7) for l_∞ and (77.1, 45.2) for l_2.

Summary The results in Table 1 demonstrate that training with TrH regularization either matches or outperforms the robust accuracy of the existing methods. In fact, the gains of TrH regularization are wider on the larger ImageNet dataset compared to on CIFAR-10/100, where several methods show equivalent performance (per Standard Error analysis). This suggests that future work in robust adversarial training should move beyond the CIFAR benchmarks to larger-scale tasks.

4.3. Sensitivity and Efficiency

Sensitivity to λ. To study the sensitivity of λ to the training in TrH regularization, we train a ViT-L16 model over the ImageNet dataset and vary the choice of λ. Figure 3. plots the AA accuracy (y-axis) against the different choices of λ (x-axis) with all other setups fixed. The plot shows that $\lambda = 5 \times 10^{-4}$ attains the highest AA accuracy for both AT and TRADES losses with little degradation in the range of $[10^{-4}, 10^{-3}]$.

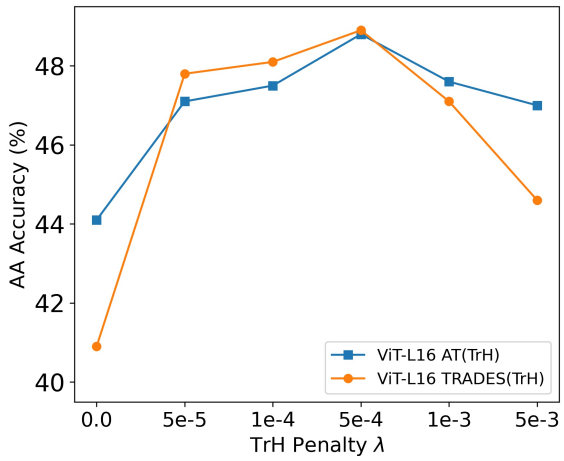

Figure 3. A plot of the Autoattack accuracy (AA %) against λ (TrH Penalty coefficient) when training on ImageNet with TrH Regularization in the ℓ_∞ case.

Defense	Mem.(MB/chip)	Img/sec/chip
AT(base)	5.0×10^3	5.5
AT(TrH) (ours)	5.0×10^3	5.2
AT(SWA)	5.6×10^3	4.4
AT(S2O)	8.0×10^3	5.2
AT(AWP)	6.8×10^3	3.7

Table 2. Peak memory usage and run-time reports measured when training CIFAR-10 using a ViT-L16 with a batch size of 32. Training is paralleled on two NVIDIA RTX chips. Lower memory usage and higher img/sec/chip are more efficient.

Training Efficiency. To compare the computational efficiency of the various methods, we report the peak memory usage and runtime performance (i.e., images processed per second; the higher the better) in Table 2. As the cloud TPU chips are dynamically allocated and preemptable, we instead measure these costs on two local NIVIDA RTX chips with 24G memory per chip. Because by default JAX preallocates all GPUs, we set `XLA_PYTHON_CLIENT_ALLOCATOR=platform` before measuring the memory allocation and run-time. In Table 2, we show that adding TrH regularization brings almost no additional memory usage and runtime slowdown, compared to training without any regularization.

5. Related Work and Discussion

Perhaps the closest approach to ours is the PACTran-Gaussian metric [12], which uses a 2nd-order approximation of the PAC-Bayesian bound as the metric for evaluating the transferability of pretrained checkpoints. Similarly to our Eq. 5, the PACTran metric is composed of two terms: an ℓ_2 regularized empirical risk (RER) and a flatness regularizer (FR). Our work was inspired by the PACTran metric, but also makes several improvements. Firstly, in PACTran a single posterior variance σ_q^2 was shared for all parameters. On the contrary, we used different σ_q^2's for different parameters. Secondly, in PACTran the optimization of the posterior mean θ_q is over the RER term only, while our optimization is over both RER and FR. These two changes potentially make our bound tighter than the one in the PACTran paper. In addition, we also study and explicitly include the high-order terms in $O(\sigma_0^4)$, which was neglected in PACTran.

The main difference between our method and other sharpness-aware and PAC-Bayesian motivated method, such as SAM [15] and AWP [49], lies on how sharpness and PAC-Bayesian theory are positioned. SAM/AWP is motivated by empirically reducing the sharpness of the training loss landscape, whereas our work is theoretically motivated and directly minimizes the PAC-Baysian bound of the (adversarial) loss w.r.t the Gaussian posterior so as to reduce the generalization error. Specifically, the sharpness measure TrH emerges as a result of that optimization. Although a PAC-Bayesian bound was also considered as a theoretical motivation for SAM/AWP, it is unclear whether its optimization indeed reduces the PAC-Bayesian bound.

Beyond the PAC-Bayesian inspired approaches for robust training, a different line of work focuses on adapting PAC-Bayesian bounds for an ensemble of models, to improve either adversarial robustness [45] or out-of-distribution (OOD) robustness data [52]. Several other works have focused on methods for finding flat local minima of the loss surface in order to improve (standard) model performance [12, 15, 25]. Also outside the scope of adversarial robustness, Ju et al. [28] find that trace of Hessian regularization leads to robustness against noise in the *labels*. Their Hessian regularization is based on randomized estimation and is done for all layers, whereas we effectively apply TrH regularization on the top layer only and with Theorem 4 as theoretical justification.

6. Conclusion

In summary, we alleviate overfitting in adversarial training with PAC-Bayesian theory. Using a linear-form PAC-Bayesian bound overlooked by the prior work, we derive a 2nd-order estimation that includes the Trace of Hessian (TrH) of the model. We focus on the TrH of the top layer only, showing both empirically and theoretically (for ReLU networks) that it leads to an effective minimization of TrH of the full network, thereby providing a sound flatness regularization technique for adversarial training. Experiments show that TrH regularization method is consistently among the top performers on CIFAR-10/100 benchmark as well as achieves a significant improvement on robust test accuracy on ImageNet compared to other baselines.

References

[1] Github - deepmind-research/adversarial_robustness. 6

[2] Github - google-research/vision_transformer. 6

[3] Guillaume Alain, Nicolas Le Roux, and Pierre-Antoine Manzagol. Negative eigenvalues of the hessian in deep neural networks, 2018. 4

[4] Pierre Alquier. User-friendly introduction to pac-bayes bounds, 2021. 1, 3

[5] Haim Avron and Sivan Toledo. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *J. ACM*, 58(2), apr 2011. 4

[6] Nicholas Carlini and Hany Farid. Evading deepfake-image detectors with white- and black-box attacks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2804–2813, 2020. 1

[7] 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. 1

[8] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. 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. 1

[9] Olivier Catoni. A pac-bayesian approach to adaptive classification. 2004. 1, 3

[10] Thomas Cilloni, Charles Walter, and Charles Fleming. Focused adversarial attacks. *ArXiv*, abs/2205.09624, 2022. 1

[11] 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

[12] Nan Ding, Xi Chen, Tomer Levinboim, Beer Changpinyo, and Radu Soricut. Pactran: Pac-bayesian metrics for estimating the transferability of pretrained models to classification tasks. In *ECCV*, 2022. 2, 3, 4, 8

[13] Nan Ding, Xi Chen, Tomer Levinboim, Sebastian Goodman, and Radu Soricut. Bridging the gap between practice and pac-bayes theory in few-shot meta-learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 29506–29516. Curran Associates, Inc., 2021. 4

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 2, 6, 32

[15] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021. 3, 8

[16] Pascal Germain, Francis Bach, Alexandre Lacoste, and Simon Lacoste-Julien. Pac-bayesian theory meets bayesian inference. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. 2

[17] Pascal Germain, Francis Bach, Alexandre Lacoste, and Simon Lacoste-Julien. Pac-bayesian theory meets bayesian inference. volume 29, 2016. 3, 4

[18] 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*, ICML '09, page 353–360, New York, NY, USA, 2009. Association for Computing Machinery. 1, 2, 3

[19] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2015. 1

[20] Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples, 2020. 1

[21] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy Mann. Improving robustness using generated data. In *NeurIPS*, 2021. 1, 2, 3, 6, 29

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 6

[23] Yujia Huang, Huan Zhang, Yuanyuan Shi, J Zico Kolter, and Anima Anandkumar. Training certifiably robust neural networks with efficient local lipschitz bounds. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. 6

[24] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, 34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018, pages 876–885. Association For Uncertainty in Artificial Intelligence (AUAI), 2018. 1, 2, 3, 6

[25] Zhiwei Jia and Hao Su. Information-theoretic local minima characterization and regularization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4773–4783. PMLR, 2020. 8

[26] Gaojie Jin, Xinping Yi, Wei Huang, Sven Schewe, and Xiaowei Huang. Enhancing adversarial training with second-order statistics of weights. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15273–15283, June 2022. 1, 2, 3, 6

[27] Norman P. Jouppi, Cliff Young, Nishant Patil, David A. Patterson, Gaurav Agrawal, Raminder Singh Bajwa, Sarah Bates, Suresh Bhatia, Nanette J. Boden, Al Borchers, Rick Boyle, Pierre luc Cantin, Clifford Chao, Chris Clark, Jeremy Coriell, Mike Daley, Matt Dau, Jeffrey Dean, Ben Gelb, Tara Vazir Ghaemmaghami, Rajendra Gottipati, William Gulland, Robert B. Hagmann, C. Richard Ho, Doug Hogberg, John Hu, Robert Hundt, Daniel Hurt, Julian Ibarz,

Aaron Jaffey, Alek Jaworski, Alexander Kaplan, Harshit Khaitan, Daniel Killebrew, Andy Koch, Naveen Kumar, Steve Lacy, James Laudon, James Law, Diemthu Le, Chris Leary, Zhuyuan Liu, Kyle A. Lucke, Alan Lundin, Gordon MacKean, Adriana Maggiore, Maire Mahony, Kieran Miller, Rahul Nagarajan, Ravi Narayanaswami, Ray Ni, Kathy Nix, Thomas Norrie, Mark Omernick, Narayana Penukonda, Andy Phelps, Jonathan Ross, Matt Ross, Amir Salek, Emad Samadiani, Chris Severn, Gregory Sizikov, Matthew Snelham, Jed Souter, Dan Steinberg, Andy Swing, Mercedes Tan, Gregory Thorson, Bo Tian, Horia Toma, Erick Tuttle, Vijay Vasudevan, Richard Walter, Walter Wang, Eric Wilcox, and Doe Hyun Yoon. In-datacenter performance analysis of a tensor processing unit. *2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*, pages 1–12, 2017. 6

[28] Haotian Ju, Dongyue Li, and Hongyang R Zhang. Robust fine-tuning of deep neural networks with hessian-based generalization guarantees. In *Proceedings of the 39th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 10431–10461. PMLR, 17–23 Jul 2022. 8

[29] Harini Kannan, Alexey Kurakin, and Ian J. Goodfellow. Adversarial logit pairing. *ArXiv*, abs/1803.06373, 2018. 23

[30] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *ArXiv*, abs/1607.02533, 2017. 1

[31] Klas Leino, Zifan Wang, and Matt Fredrikson. Globally-robust neural networks. In *International Conference on Machine Learning*, pages 6212–6222. PMLR, 2021. 1, 6

[32] 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, 2, 6

[33] Kaleel Mahmood, Rigel Mahmood, and Marten van Dijk. On the robustness of vision transformers to adversarial examples. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7838–7847, October 2021. 6, 27, 32

[34] David A. McAllester. Pac-bayesian model averaging. In *COLT '99*, 1999. 1, 3

[35] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. *Advances in neural information processing systems*, 30, 2017. 3

[36] Tianyu Pang, Min Lin, Xiao Yang, Junyi Zhu, and Shuicheng Yan. Robustness and accuracy could be reconcilable by (proper) definition. In *ICML*, 2022. 1

[37] Siyuan Qiao, Huiyu Wang, Chenxi Liu, Wei Shen, and Alan Yuille. Micro-batch training with batch-channel normalization and weight standardization. *arXiv preprint arXiv:1903.10520*, 2019. 32

[38] Leslie Rice, Eric Wong, and J. Zico Kolter. Overfitting in adversarially robust deep learning. In *ICML*, 2020. 1, 3

[39] Kevin Roth, Yannic Kilcher, and Thomas Hofmann. Adversarial training is a form of data-dependent operator norm regularization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14973–14985. Curran Associates, Inc., 2020. 6

[40] Jonas Rothfuss, Vincent Fortuin, Martin Josifoski, and Andreas Krause. Pacoh: Bayes-optimal meta-learning with pac-guarantees. In *International Conference on Machine Learning*, pages 9116–9126. PMLR, 2021. 3

[41] Vikash Sehwag, Saeed Mahloujifar, Tinashe Handina, Sihui Dai, Chong Xiang, Mung Chiang, and Prateek Mittal. Robust learning meets generative models: Can proxy distributions improve adversarial robustness? In *ICLR*, 2022. 1

[42] John Shawe-Taylor and Robert C. Williamson. A pac analysis of a bayesian estimator. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, COLT '97, page 2–9, New York, NY, USA, 1997. Association for Computing Machinery. 1, 3

[43] Dawn Song, Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramèr, Atul Prakash, and Tadayoshi Kohno. Physical adversarial examples for object detectors. In *12th USENIX Workshop on Offensive Technologies (WOOT 18)*, Baltimore, MD, Aug. 2018. USENIX Association. 1

[44] P.B. Stark and Berkeley. Department of Statistics University of California. *SticiGui: Statistics Tools for Internet and Classroom Instruction*. Department of Statistics. University of California, Berkeley, 2005. 6, 7, 31, 32

[45] Paul Viallard, Eric Guillaume VIDOT, Amaury Habrard, and Emilie Morvant. A pac-bayes analysis of adversarial robustness. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 14421–14433. Curran Associates, Inc., 2021. 8

[46] 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*, 2020. 1, 24

[47] Xingxing Wei, Siyuan Liang, Xiaochun Cao, and Jun Zhu. Transferable adversarial attacks for image and video object detection. In *IJCAI*, 2019. 1

[48] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 1

[49] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *NeurIPS*, 2020. 1, 2, 3, 6, 8, 29

[50] Yuxin Wu and Kaiming He. Group normalization. *International Journal of Computer Vision*, 128:742–755, 2019. 32

[51] 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 *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. 1, 3, 28

[52] Matteo Zecchin, Sangwoo Park, Osvaldo Simeone, Marios Kountouris, and David Gesbert. Robust pacm: Training en-

semble models under model misspecification and outliers. *ArXiv*, abs/2203.01859, 2022. 8

[53] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 1, 3

[54] 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. 1, 2