Enhance Diffusion to Improve Robust Generalization

Jianhui Sun
University of Virginia,
js9gu@virginia.edu

Sanchit Sinha
University of Virginia,
ss7mu@virginia.edu

Aidong Zhang
University of Virginia,
aidong@virginia.edu

ABSTRACT

Deep neural networks are susceptible to human imperceptible adversarial perturbations. One of the strongest defense mechanisms is *Adversarial Training* (AT). In this paper, we aim to address two predominant problems in AT. First, there is still little consensus on how to set hyperparameters with a performance guarantee for AT research, and customized settings impede a fair comparison between different model designs in AT research. Second, the robustly trained neural networks struggle to generalize well and suffer from tremendous overfitting. This paper focuses on the primary AT framework - Projected Gradient Descent Adversarial Training (PGD-AT). We approximate the dynamic of PGD-AT by a continuous-time Stochastic Differential Equation (SDE), and show that the diffusion term of this SDE determines the robust generalization. An immediate implication of this theoretical finding is that robust generalization is positively correlated with the ratio between learning rate and batch size. We further propose a novel approach, *Diffusion Enhanced Adversarial Training* (DEAT), to manipulate the diffusion term to improve robust generalization with virtually no extra computational burden. We theoretically show that DEAT obtains a tighter generalization bound than PGD-AT. Our empirical investigation is extensive and firmly attests that DEAT universally outperforms PGD-AT by a significant margin.

CCS CONCEPTS

• **Theory of computation** → **Sample complexity and generalization bounds**; • **Computing methodologies** → **Adversarial learning**; • **Mathematics of computing** → *Stochastic differential equations.*

KEYWORDS

Adversarial Training (AT), Projected Gradient Descent Adversarial Training (PGD-AT), Robust Generalization, Stochastic Differential Equation (SDE), Diffusion Enhanced Adversarial Training (DEAT)

ACM Reference Format:
Jianhui Sun, Sanchit Sinha, and Aidong Zhang. 2023. Enhance Diffusion to Improve Robust Generalization. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3580305.3599333

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08.
https://doi.org/10.1145/3580305.3599333

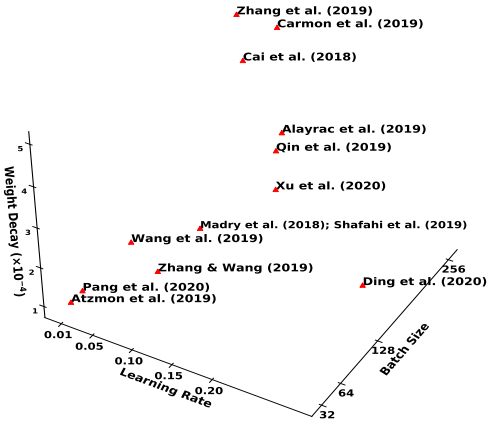


Figure 1: We summarize three key hyperparameters (learning rate, batch size, weight decay) used in a list of recent papers [2, 4, 6, 12, 36, 41, 44, 47, 57, 59, 67, 69, 70]. The hyperparameter specifications are highly inconsistent and a fair comparison is difficult in such condition as we will demonstrate in our empirical experiments, these hyperparameters make a huge difference in robust generalization.

1 INTRODUCTION

Despite achieving surprisingly good performance in a wide range of tasks, deep learning models have been shown to be vulnerable to adversarial attacks which add human imperceptible perturbations that could significantly jeopardize the model performance [15]. Adversarial training (AT), which trains deep neural networks on adversarially perturbed inputs instead of on clean data, is one of the strongest defense strategies against such adversarial attacks.

This paper mainly focuses on the primary AT framework - Projected Gradient Descent Adversarial Training (PGD-AT) [36]. Though many new improvements [1] have been proposed on top of PGD-AT to be better tailored to the needs of different domains, or to alleviate the heavy computational burden [6, 22, 38, 47, 56, 58, 61, 69], PGD-AT at its vanilla version is still the default choice in most scenarios due to its compelling performances in several adversarial competitions [3, 31].

1.1 Motivation

This paper aims to address the following problems in AT:

I. **Inconsistent hyperparameter specifications impede a fair comparison between different model designs.**

[1] e.g., training tricks including early stopping w.r.t. the training epoch [45], and label smoothing [47] prove to be useful to improve robustness.

Though the configuration of hyperparameters is known to play an essential role in the performance of AT, there is little consensus on how to set hyperparameters with a performance guarantee. For example in Figure 1, we plot a list of recent AT papers on the (learning rate, batch size, weight decay) space according to each paper's specification and we could observe that the hyperparameters of each paper are relatively different from each other with little consensus. Moreover, the completely customized settings make it extremely difficult to understand which approach really works, as the misspecification of hyperparameters would potentially cancel out the improvements from the methods themselves. Most importantly, the lack of theoretical understanding also exhausts practitioners with time-consuming tuning efforts.

II. **The robust generalization gap in AT is surprisingly large.** Overfitting is a dominant problem in adversarially trained deep networks [45]. To demonstrate that, we run both standard training (non-adversarial) and adversarial training on CIFAR10 with VGG [49] and SENet [23]. Training curves are reported in Figure 2. We could observe the robust test accuracy is much lower than the standard test accuracy. Further training will continue to improve the robust training loss of the classifier, to the extent that robust training loss could closely track standard training loss [46], but fail to further improve robust testing loss. Early stopping is advocated to partially alleviate overfitting [45, 71], but there is still huge room for improvement.

1.2 Contribution

In this paper, to address the aforementioned problems, we consider PGD-AT as an alternating stochastic gradient descent [2]. Motivated by the theoretical attempts which approximate the discrete-time dynamic of stochastic gradient algorithm with continuous-time Stochastic Differential Equation (SDE) [19, 33, 37, 72], we derive the continuous-time SDE dynamic for PGD-AT. The SDE contains a drift term and a diffusion term, and we further prove the diffusion term determines the robust generalization performance.

As the diffusion term is determined by (A) ratio of learning rate α and batch size b and (B) gradient noise, an immediate implication of our theorem is that the robust generalization has a positive correlation with the size of both (A) and (B). In other words, we could improve robust generalization via scaling up (A) and (B). Although it is fairly simple to scale up (A) by increasing α and decreasing b, adjusting α and b could be a double-edged sword. One reason is that small batch improves generalization while significantly increases training time. Considering the computational cost of adversarial training is already extremely expensive (e.g., the PGD-10 training of ResNet on CIFAR-10 takes several days on a single GPU), large batch training is apparently more desirable. α is allowed to increase only within a very small range to ensure convergence of AT algorithm.

[2]It is natural to view PGD-AT as an alternating stochastic gradient descent as PGD-AT is a min-max game, where in the inner maximization, an adversary generates adversarial examples against the neural network via multi-step projected gradient descent, while in the outer minimization, the neural network is updated via a single-step gradient descent based on the generated perturbed data. Please refer to Section 2 for formal definition.

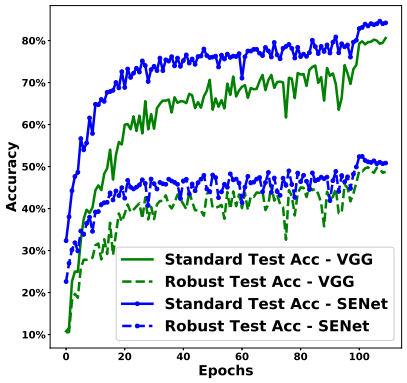


Figure 2: Classification accuracy for standard training (non-adversarial) and adversarial training on CIFAR10 with VGG and SENet. Table 1 summarizes the experimental setting. The adversarial test accuracy (dashed line) is far from the non-adversarial test accuracy (solid line) in both architectures. The generalization gap for the robust accuracy is significant and much larger than normal training.

To overcome the aforementioned limitations, we propose a novel algorithm, DEAT (*Diffusion Enhanced Adversarial Training*), to instead adjust (B) to improve robust generalization (see Algorithm 2). Our approach adds virtually no extra computational burden, and universally achieves better robust testing accuracy over vanilla PGD-AT by a large margin. We theoretically prove DEAT achieves a tighter robust generalization gap. Our extensive experimental investigation strongly supports our theoretical findings and attests the effectiveness of DEAT.

We summarize our contributions as follows:

I. Theoretically, we approximate PGD-AT with a continuous-time SDE, and prove the diffusion term of this SDE determines the robust generalization. The theorem guides us how to tune α and b in PGD-AT. To our best knowledge, this is the first study that rigorously proves the role of hyperparameters in AT.

II. Algorithmically, we propose a novel approach, DEAT (Diffusion Enhanced Adversarial Training), to manipulate the diffusion term with virtually no additional computational cost, and manage to universally improve over vanilla PGD-AT by a significant margin. We also theoretically show DEAT is guaranteed to generalize better than PGD-AT. Interestingly, DEAT also improves the generalization performance in non-adversarial tasks, which further verifies our theoretical findings.

Organization In Section 2, we formally introduce adversarial training and PGD-AT, which are pertinent to this work. In Section 3, we present our main theorem that derives the robust generalization bound of PGD-AT. In Section 4, motivated by the theoretical findings and in recognition of the drawbacks in adjusting α and b, we present our novel DEAT (Diffusion Enhanced Adversarial Training). We theoretically show DEAT has a tighter generalization bound. In Section 5, we conduct extensive experiments to verify our

theoretical findings and the effectiveness of DEAT. Related works are discussed in Section 6. Proofs of all our theorems and corollaries are presented in Appendix.

2 BACKGROUND: PGD-AT

In this section, we formally introduce PGD-AT which is the main focus of this work.

Notation: This paragraph summarizes the notation used throughout the paper. Let θ, $\mathcal{D}$, and $l_\theta(x_i, y_i)$ be the trainable model parameter, data distribution, and loss function, respectively. Let $\{z_i = (x_i, y_i)\}_{i=1}^N$ denote the training set, and $\{x_i\}_{i=1}^N \subset \mathbb{R}^d$. Expected risk function is defined as $\mathcal{R}(\theta) \triangleq \mathbb{E}_{z\sim\mathcal{D}} l_\theta(z)$. Empirical risk $\mathcal{R}_\zeta(\theta)$ is an unbiased estimator of the expected risk function, and is defined as $\mathcal{R}_\zeta(\theta) \triangleq \frac{1}{b}\sum_{j\in\zeta}\mathcal{R}_j(\theta)$, where $\mathcal{R}_j(\theta) \triangleq l_\theta(z_j)$ is the contribution to risk from j-th data point. ζ represents a mini-batch of random samples and $b \triangleq |\zeta|$ represents the batch size. Similarly, we define $\nabla_\theta\mathcal{R}$, $\nabla_\theta\mathcal{R}_j$, and $\nabla_\theta\mathcal{R}_\zeta$ as their gradients, respectively. We denote the empirical gradient as $\hat{g}(\theta) \triangleq \nabla_\theta\mathcal{R}_\zeta$ and exact gradient as $g(\theta) \triangleq \nabla_\theta\mathcal{R}$ for the simplicity of notation.

In standard training, most learning tasks could be formulated as the following optimization problem:

$$\min_\theta \mathcal{R}(\theta) = \min_\theta \mathbb{E}_{(x_i,y_i)\sim\mathcal{D}} l_\theta(x_i, y_i), \tag{1}$$

Stochastic Gradient Descent (SGD) and its variants are most widely used to optimize (1). SGD updates with the following rule:

$$\theta_{t+1} = \theta_t - \alpha_t s_t, \tag{2}$$

where α_t and s_t are the learning rate and search direction at t-th step, respectively. SGD uses $\hat{g}_t \triangleq \hat{g}(\theta_t)$ as s_t.

The performance of learning models, depends heavily on whether SGD is able to reliably find a solution of (1) that could generalize well to unseen test instances.

An adversarial attacker aims to add a human imperceptible perturbation to each sample, i.e., transform $\{z_i = (x_i, y_i)\}_{i=1}^N$ to $\{\tilde{z}_i = (\tilde{x}_i = x_i + \delta_i, y_i)\}_{i=1}^N$, where perturbations $\{\delta_i\}_{i=1}^N$ are constrained by a pre-specified budget Δ ($\delta_i \in \Delta$), such that the loss $l_\theta(\tilde{x}_i, y_i)$ is large. The choice of budget is flexible. A typical formulation is $\{\delta \in \mathbb{R}^d : \|\delta\|_p \le \epsilon\}$ for $p = 1, 2, \infty$. In order to defend such attack, we resort to solving the following objective function:

$$\min_{\theta\in\mathbb{R}_\theta} \rho(\theta), \text{ where } \rho(\theta) = \mathbb{E}_{(x_i,y_i)\sim\mathcal{D}}[\max_{\delta_i\in\Delta} l_\theta(x_i + \delta_i, y_i)] \tag{3}$$

Objective function (3) is a composition of an inner maximization problem and an outer minimization problem. The inner maximization problem simulates an attacker who aims to find an adversarial version of a given data point x_i that achieves a high loss, while the outer minimization problem is to find model parameters so that the "adversarial loss" given by the inner attacker is minimized. Projected Gradient Descent Adversarial Training (PGD-AT) [36] solves this min-max game by gradient ascent on the perturbation parameter δ before applying gradient descent on the model parameter θ.

The detailed pseudocode of PGD-AT is in Algorithm 1. Basically, projected gradient descent (PGD) is applied K steps on the negative loss function to produce strong adversarial examples in the inner loop, which can be viewed as a multi-step variant of Fast Gradient Sign Method (FGSM) [15], while every training example is replaced with its PGD-perturbed counterpart in the outer loop to produce a model that an adversary could not find adversarial examples easily.

Algorithm 1: PGD-AT (Projected Gradient Descent Adversarial Training) [36]

Input: Loss function $l_\theta(z_i)$, initialization θ_0, total training steps T, PGD steps K, inner/outer learning rates α_I/α_O, batch size b, perturbation budget set Δ;

for $t \in \{1, 2, ..., T\}$ **do**
  Sample a mini-batch of random examples $\zeta = \{(x_{i_j}, y_{i_j})\}_{j=1}^b$;
  Set $\delta_0 = 0, \hat{x}_j = x_{i_j}$;
  for $k \in \{1, ..., K\}$ **do**
    $\delta_k = \Pi_\Delta(\delta_{k-1} + \frac{\alpha_I}{b}\sum_{j=1}^b \nabla_x l_{\theta_{t-1}}(\hat{x}_j + \delta_{k-1}, y_{i_j}))$;
  end
  $\theta_t = \theta_{t-1} - \frac{\alpha_O}{b}\sum_{j=1}^b \nabla_\theta l_{\theta_{t-1}}(\hat{x}_j + \delta_K, y_{i_j}))$;
end
return θ_T

3 THEORY: ROBUST GENERALIZATION BOUND OF PGD-AT

In this section, we describe our logical framework of deriving the robust generalization gap of PGD-AT, and then identify the main factors that determine the generalization.

To summarize the entire section before we dive into details, we consider PGD-AT as an alternating stochastic gradient descent and approximate the discrete-time dynamic of PGD-AT with continuous-time Stochastic Differential Equation (SDE), which contains a drift term and a diffusion term, and we would show the diffusion term determines the robust generalization. Our theorem immediately points out the robust generalization has a positive correlation with the ratio between learning rate α and batch size b.

Let us first introduce our logical framework in Section 3.1 before we present main theorem in Section 3.2.

3.1 Roadmap to robust generalization bound

Continuous-time dynamics of gradient based methods

A powerful analysis tool for stochastic gradient based methods is to model the continuous-time dynamics with stochastic differential equations and then study its limit behavior [19, 24, 33, 37, 54, 66]. [37] characterizes the continuous-time dynamics of using a constant step size SGD (2) to optimize normal training task (1).

Lemma 1 ([37]). *Assume the risk function* [3] *is locally quadratic, and gradient noise is Gaussian with mean 0 and covariance* $\frac{1}{b}H$*, and* $H = BB^T$ *for some B. The following two statements hold,*

I. Constant-step size SGD (2) *could be recast as a discretization of the following continuous-time dynamics:*

[3] Without loss of generality, we assume the minimum of the risk function is at $\theta = 0$, as we could always translate the minimum to 0.

$$d\theta = -\alpha g(\theta)dt + \frac{\alpha}{\sqrt{b}}BdW_t \tag{4}$$

where $dW_t = \mathcal{N}(0, Idt)$ is a Wiener process.

II. *The stationary distribution of stochastic process* (4) *is Gaussian and its covariance matrix Q is explicit.*

$\alpha g(\theta)$ and $\frac{\alpha}{\sqrt{b}}B$ are referred to as drift and diffusion, respectively. Many variants of SGD (e.g. heavy ball momentum [43] and Nesterov's accelerated gradient [40]) can also be cast as a modified version of (4), and we could explicitly write out its stationary distribution as well [14].

Next we will discuss PAC-Bayesian generalization bound and how it connects to the SDE approximation.

PAC-Bayesian generalization bound

Bayesian learning paradigm studies a distribution of every possible setting of model parameter θ instead of betting on one single setting of parameters to manage model uncertainty, and has proven increasingly powerful in many applications. In Bayesian framework, θ is assumed to follow some prior distribution P (reflects the prior knowledge of model parameters), and at each iteration of SGD (2) (or any other stochastic gradient based algorithm), the θ distribution shifts to $\{Q_t\}_{t\geq 0}$, and converges to posterior distribution Q (reflects knowledge of model parameters after learning with $\mathcal{D}$).

Bayesian risk function is defined as $\mathcal{R}(Q) \triangleq \mathbb{E}_{\theta\sim Q}\mathbb{E}_{(x,y)\sim\mathcal{D}}l(f_\theta(x), y)$, and $\hat{\mathcal{R}}(Q) \triangleq \mathbb{E}_{\theta\sim Q}\frac{1}{N}\sum_{j=1}^{N} l(f_\theta(x_j), y_j)$. $\mathcal{R}(Q)$ is the population risk, while $\hat{\mathcal{R}}(Q)$ is the risk evaluated on the training set and $N \triangleq |\mathcal{D}|$ is the sample size. The generalization bound could therefore be defined as follows:

$$\mathcal{E} \triangleq |\mathcal{R}(Q) - \hat{\mathcal{R}}(Q)|. \tag{5}$$

The following lemma connects generalization bound to the stationary distribution of a stochastic gradient algorithm.

Lemma 2 (PAC-Bayesian Generalization Bound [39, 48]). *Let $KL(Q||P)$ be the Kullback-Leibler divergence between distributions Q and P. For any positive real $\varepsilon \in (0, 1)$, with probability at least $1-\varepsilon$ over a sample of size N, we have the following inequality for all distributions Q:*

$$|\mathcal{R}(Q) - \hat{\mathcal{R}}(Q)| \leq \sqrt{\frac{KL(Q||P) + \log\frac{1}{\varepsilon} + \log N + 2}{2N-1}} \tag{6}$$

Let $\mathcal{G}$ denote $\sqrt{\frac{\mathrm{KL}(Q||P)+\log\frac{1}{\varepsilon}+\log N+2}{2N-1}}$, which is an upper bound of generalization error, i.e. $\mathcal{G}$ is an upper bound of $\mathcal{E}$ in (5). The prior P is typically assumed to be a Gaussian prior $\mathcal{N}(\theta_0, \lambda_0 I_d)$, reflecting the common practice of Gaussian initialization [13] and L_2 regularization, and posterior distribution Q is the stationary distribution of the stochastic gradient algorithm under study.

The importance of Lemma 2 is that, we could easily get an upper bound of generalization bound if we could explicitly represent KL$(Q||P)$. Recall Lemma 1 gives the exact form of Q for SGD, and therefore, naturally results in a generalization bound.

3.2 Robust generalization of PGD-AT

SGD (2) can be viewed as a special example (by setting the total steps of PGD attack $K = 0$) of PGD-AT (Algorithm 3) [4]. PGD-AT is also a stochastic gradient based iterative updating process. Therefore, a natural question arises:

Can we approximate the continuous-time dynamics of PGD-AT by a stochastic differential equation?

We provide a positive answer to this question in Theorem 1. However, general SDEs do not possess closed-form stationary distributions, which makes the downstream tasks extremely difficult to proceed. The following question requires answering:

Can we explicitly represent the stationary distribution of this SDE and subsequently calculate $KL(Q||P)$ required in Lemma 2?

The answer also has a positive answer with mild assumptions. With the stationary distribution, we will leverage Lemma 2 to derive a generalization bound of PGD-AT, which would be a powerful analytic tool to identify the main factors that determine the robust generalization.

We are now ready to give our main theorem.

Theorem 1. *Assume the risk function is locally quadratic, and gradient noise is Gaussian* [5]*. Suppose inner learning rate equals outer learning rate, and they are both fixed, i.e., $\alpha = \alpha_I = \alpha_O$. Let the Hessian matrix of risk function be A, and covariance matrix of Gaussian noise be $H = BB^T$. Let b denote the batch size and $\mathcal{G}$ be the upper bound of generalization error.*

The following statements hold,

1. The continuous-time dynamics of PGD-AT can be described by the following stochastic process:

$$d\theta = f dt + \sigma dW_t, \quad \text{where} \quad dW_t = \mathcal{N}(0, Idt) \quad \text{is Wiener process,}$$
$$f = -(A + (K + \frac{1}{2})\alpha A^2)\theta \qquad \text{and} \quad \sigma = \sqrt{\frac{\alpha}{b}}AB \tag{7}$$

f and σ are referred to as drift term and diffusion term, respectively.

2. This stochastic process (7) *is known as an Ornstein-Uhlenbeck process. The stationary distribution of this stochastic process is a Gaussian distribution with explicit covariance Σ. The norm of Σ is positively correlated with $\frac{\alpha}{b}$ and norm of B.*

3. Larger $\frac{\alpha}{b}$ and/or norm of B results in smaller $\mathcal{G}$, i.e., induces tighter robust generalization bound.

Proof. Please refer to Appendix for proof. □

Theorem 1 implies the following important statements,

(A) Diffusion term $\sqrt{\frac{\alpha}{b}}AB$ is impactful in the robust generalization, and we could manipulate diffusion to improve robust generalization.

(B) We could effectively boost robust generalization via increasing α and decreasing b. We provide extensive empirical evidence to support this claim (See e.g. Figure 4 and Table 2).

[4] As a matter of fact, all our theoretical findings in this paper also apply for non-AT settings, i.e., ordinary learning tasks without adversarial attacks. Interestingly, our proposed approach DEAT is not only capable of improving robust generalization in AT, but also improving generalization performance in ordinary learning tasks by setting $K = 0$ compared to SGD. Please refer to Section **??** for more details.

[5] Please refer to Section 3.3 for more details

3.3 Analysis of Assumptions

We first present the following standard assumptions from existing studies that are used in Theorem 1 and discuss why these assumptions stand.

Assumption 1 ([19, 37, 66] The second-order Taylor approximation). *Suppose the risk function is approximately convex and 2-order differentiable, in the region close to minimum, i.e., there exists a $\delta_0 > 0$, such that $\mathcal{R}(\theta) = \frac{1}{2}(\theta - \theta^*)^T A(\theta - \theta^*)$ if $\|\theta - \theta^*\| \leq \delta_0$, where θ^* is a minimizer of $\mathcal{R}(\theta)$. Here A is the Hessian matrix $\nabla^2_\theta \mathcal{R}$ around minimizer and is positive definite. Without loss of generality, we assume a minimizer of the risk is zero, i.e., $\theta^* = 0$.*

Though here we assume locally quadratic form of risk function, all our results from this study apply to locally smooth and strongly convex objectives. Note that the assumption on locally quadratic structure of loss function, even for extremely nonconvex objectives, could be justified empirically. [32] visualized the loss surfaces for deep structures like ResNet [20] and DenseNet [27], observing quadratic geometry around local minimum in both cases. And certain network architecture designs (e.g., skip connections) could further make neural loss geometry show no noticeable nonconvexity, see e.g. Figure 3.

Assumption 2 ([19, 37, 66] Unbiased Gradients with Bounded Noise Variance). *Suppose at each step t, gradient noise is Gaussian with mean 0 and covariance $\frac{1}{b}\Sigma(\theta_t)$, i.e.,*

$$\hat{g}(\theta_t) \approx g(\theta_t) + \frac{1}{\sqrt{b}}\Delta g(\theta_t), \quad \Delta g(\theta_t) \sim \mathcal{N}(0, \Sigma(\theta_t))$$

We further assume that the noise covariance matrix $\Sigma(\theta_t)$ is approximately constant with respect to θ, i.e., $\Sigma(\theta_t) \approx \Sigma = CC^T$. And noises from different iterates $\{\Delta g(\theta_t)\}_{t\geq 1}$ are mutually statistically independent.

Gaussian gradient noise is natural to assume as the stochastic gradient is a sum of b independent, uniformly sampled contributions. Invoking the central limit theorem, the noise structure could be approximately Gaussian. Assumption 2 is standard when approximating a stochastic algorithm with a continuous-time stochastic process (see e.g. [37]) and is justified when the iterates are confined to a restricted region around the minimizer.

4 ALGORITHM: DEAT - A 'FREE' BOOSTER TO PGD-AT

Theorem 1 indicates that the key factor which impacts the robust generalization is diffusion $\sqrt{\frac{\alpha}{b}}AB$. And the definitive relationship is, large diffusion level positively benefits the generalization performance of PGD-AT.

Though increasing $\frac{\alpha}{b}$ is straightforward, there are two main drawbacks. First, decreasing batch size is impractical as it significantly lengthens training time. Adversarial training already takes notoriously lengthy time compared to standard supervised learning (as the inner maximization is essentially several steps of gradient ascent). Thus, small batch size is simply not an economical option, Second, the room to increase α is very limited as α has to be relatively small to ensure convergence.

Furthermore, we also desire an approach that could universally improve the robust generalization independent of specifications of α and b, as they could potentially complement each other to achieve a even better performance. Thus, we propose to manipulate the remaining factor in the diffusion,

Can we manipulate the gradient noise level B in PGD-AT dynamic to improve its generalization?

Our proposed Diffusion Enhanced AT (DEAT) (i.e. Algorithm 2) provides a positive answer to this question. The basic idea of DEAT is simple. Inspired by the idea from [66], instead of using one single gradient estimator $\hat{g}$, Algorithm 2 maintains two gradient estimators h_t and h_{t-1} at each iteration. A linear interpolation of these two gradient estimators is still a legitimate gradient estimator, while the noise (variance) of this new estimator is larger than any one of the base estimators. k_1 and k_2 are two hyperparameters.

We would like to emphasize when h_t and h_{t-1} are two unbiased and independent gradient estimators, the linear interpolation is apparently unbiased (due to linearity of expectation) and the noise of this new estimator increases. However, DEAT (and the following Theorem 2) does not require h_t and h_{t-1} to be unbiased or independent. In fact, DEAT showcases a general idea of linear combination of two estimators which goes far beyond our current design. We could certainly devise other formulation of h_t or h_{t-1}, which may be unbiased or biased as in our current design.

It may be natural to think that why not directly inject some random noise to the gradient to improve generalization. However, existing works point out random noise does not have such appealing effect, only noise with carefully designed covariance structure and distribution class works [62]. For example, [72] and [11] point out, if noise covariance aligns with the Hessian of the loss surface to some extent, the noise would help generalize. Thus, [60] proposes to inject noise using the (scaled) Fisher as covariance and [72] proposes to inject noise employing the gradient covariance of SGD as covariance, both requiring access and storage of second order Hessian which is very computationally and memory expensive.

DEAT, compared with existing literature, is the first algorithm on adversarial training, we inject noise that does not require second order information and is "free" in memory and computation.

Theorem 2 provides a theoretical guarantee that DEAT obtains a tighter generalization bound than PGD-AT.

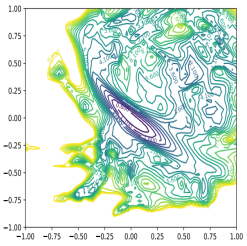
Theorem 2. *Let H_1 and H_2 be the covariance matrix of gradient noise from PGD-AT and DEAT, respectively. Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be the upper bounds of generalization error of PGD-AT (Algorithm 3) and DEAT (Algorithm 2), respectively. The following statement holds,*

$$\begin{gathered} H_2 = kH_1, \quad \text{where} \quad k > 1, \\ \mathcal{G}_1 \geq \mathcal{G}_2 \end{gathered} \tag{8}$$

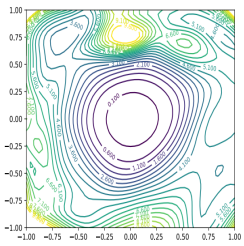
i.e., Algorithm 2 generates larger gradient noise than Algorithm 1, and such gradient noise boosts robust generalization.

Proof. We only keep primary proof procedures and omit most of the algebraic transformations. Recall the updating rule for conventional heavy ball momentum,

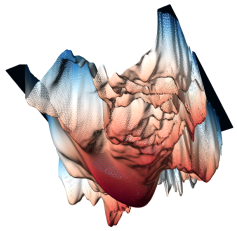
$$\begin{aligned} d_{t+1} &= (1-\beta)\hat{g}_t + \beta d_t \\ \theta_{t+1} &= \theta_t - \alpha d_t \end{aligned} \tag{9}$$



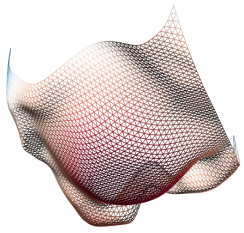
(a) WRN56-no-shortcut-connection



(b) WRN56-shortcut-connection



(c) Resnet-56-no-shortcut



(d) Resnet-56-short

Figure 3: 2D visualization of the loss surface of Wide-ResNet-56 on CIFAR-10 both without shortcut connections in Figure 3a and with shortcut connections in Figure 3b (Figure 6 in [32]). 3D visualization of the loss surface of ResNet-56 on CIFAR-10 both with shortcut connections in Figure 3d and without shortcut connections in Figure 3c (from http://www.telesens.co/loss-landscape-viz/viewer.html).

Algorithm 2: Diffusion Enhanced AT (DEAT)

Input: Loss function $J(\theta, x, \delta) = l(\theta, x + \delta, y) - \lambda R(\delta)$, initialization θ_0, total training steps T, PGD steps K, inner/outer learning rates α_I/α_O, batch size b;

1 **for** $t \in \{1, 2, ..., T\}$ **do**

2 Sample a mini-batch of random examples $\zeta = \{(x_{i_j}, y_{i_j})\}_{j=1}^{b}$;

3 Set $\delta_0 = 0, \hat{x}_j = x_{i_j}$;

4 **for** $k \in \{1, ..., K\}$ **do**

5 $\delta_k = \delta_{k-1} + \frac{\alpha_I}{b} \sum_{j=1}^{b} \nabla_\delta J(\theta_{t-1}, \hat{x}_j, \delta_{k-1})$;

6 **end**

7 $h_t = k_2 h_{t-2} + (1 - k_2)\hat{g}_t$, $\theta_{t+1} = \theta_t - \alpha'_O[(1 + k_1)h_t - k_1 h_{t-1}]$,

8 where $\alpha'_O = \frac{\alpha_O}{\sqrt{(1+k_1)^2 + k_1^2}}$;

9 **end**

10 return θ_T

where β is the momentum factor.

By some straightforward algebraic transformations, we know the momentum can be written as $d_t = (1 - \beta) \sum_{i=1}^{t} \beta^{k-i} \hat{g}_i$.

Suppose H is the noise covariance of $\hat{g}_i$ and v^2 is the scale of H, i.e., $\|H\| \leq v^2$. The noise level of d_t is $\approx (1 - \beta)\frac{1-\beta^{t+1}}{1-\beta} v^2 \approx v^2$.

Momentum d does not alter the gradient noise level. We would resort to maintain two momentum terms $d^{(1)}$ $d^{(2)}$, and use the linear interpolation $(1 + p)d^{(1)} - pd^{(2)}$ as our iterate.

The advantage is though the noise levels of $d^{(1)}$ and $d^{(2)}$ are both v^2, the noise level of $(1 + p)d^{(1)} - pd^{(2)}$ is $\approx ((1 + p)^2 + p^2)v^2$ [66].

Thus, if we could show our proposed DEAT is indeed maintaining two momentum terms, we complete the proof of the statement $H_2 = kH_1$ and $k > 1$ in Theorem 2.

Recall line 7-8 in Algorithm 2,

$$\begin{aligned} h_t &= k_2 h_{t-2} + (1 - k_2)\hat{g}_t, \\ \theta_{t+1} &= \theta_t - \alpha'_O[(1 + k_1)h_t - k_1 h_{t-1}], \end{aligned} \tag{10}$$

We could transform it into,

$$\begin{aligned} h_t &= k_2 h_{t-2} + (1 - k_2)\hat{g}_t, \\ (\theta_{t+1} - \alpha'_O k_1 h_t) &= (\theta_t - \alpha'_O k_1 h_{t-1}) - \alpha'_O h_t, \end{aligned} \tag{11}$$

We could further write it into,

$$\begin{aligned} x_t &= \theta_t - k_1 h_{t-1}, \\ x_{t+1} &= x_t - \xi\hat{g}_t + k_2(x_{t-1} - x_{t-2}), \end{aligned} \tag{12}$$

where $\xi = \alpha'_O(1 - k_2)$. We know a conventional momentum can be written as,

$$\theta_{t+1} = \theta_t - \alpha\hat{g}_t + \beta(\theta_t - \theta_{t-1}) \tag{13}$$

where α and β are learning rate and momentum factor, respectively. Note in Equation (12), the second line is exactly same as in Equation (13), indicating x_t has the same behavior as momentum. Further note $x_{t+1} - x_t = \xi h_t$, i.e., we maintain two momentum terms by alternatively using odd-number-step and even-number-step gradients. Combining everything together, we complete the proof of Theorem 2. □

One advantage of DEAT is that it adds virtually no extra parameters or computation. Though it introduces two more hyper-parameters k_1 and k_2, they are highly insensitive according to our experimental investigation.

Our experimental results in Figure 4 and Table 2 firmly attest that DEAT outperforms PGD-AT by a significant 1.5% to 2.0% margin with nearly no extra burden. We would like to emphasize that 1.5% to 2.0% improvement with virtually no extra cost is non trivial in robust accuracy. To put 1.5% to 2.0% in perspective, the difference among the robust accuracy of all popular architectures is only about 2.5% (see [42]). Our approach is nearly "free" in cost while modifying architectures includes tremendous extra parameters and model design. 2.0% is also on par with some other techniques, e.g., label smoothing, weight decay, that are already overwhelmingly used to improve robust generalization.

Training curves in Figure 5 reveal that DEAT can beat PGD-AT in adversarial testing accuracy even when PGD-AT has better adversarial training accuracy, which shows DEAT does alleviate overfitting.

5 EXPERIMENTS

We conduct extensive experiments to verify our theoretical findings and proposed approach. We include different architectures, and sweep across a wide range of hyperparameters, to ensure the robustness of our findings. All experiments are run on 4 NVIDIA Quadro RTX 8000 GPUs, and the total computation time for the experiments exceeds 10K GPU hours. Our code is available at https://github.com/jsycsjh/DEAT.

We aim to answer the following two questions:

(1) *Do hyperparameters impact robust generalization in the same pattern as Theorem 1 indicates?*
(2) *Does DEAT provide a 'free' booster to robust generalization?*

Setup We test on CIFAR-10 under the l_∞ threat model of perturbation budget $\frac{8}{255}$, without additional data. Both the vanilla PGD-AT framework and DEAT is used to produce adversarially robust model. The model is evaluated under 10-steps PGD attack (PGD-10) [36]. Note that this paper mainly focuses on PGD attack instead of other attacks like AutoAttack [10] / RayS [9] for consistency with our theorem. The architectures we test with include VGG-19 [49], SENet-18 [23], and Preact-ResNet-18 [21]. Every single data point is an average of 3 independent and repeated runs under exactly same settings (i.e., every single robust accuracy in Table 2 is an average of 3 runs to avoid stochasticity). The following Table 1 summarizes the default settings in our experiments.

Table 1: Experimental Settings

Batch Size: 128	Label Smoothing: False
Weight Decay: 5×10^{-4}	BN Mode: eval
Activation: ReLu	Total Epoch: 110
LR Decay Factor: 0.1	LR Decay Epochs: 100, 105
Attack: PGD-10	Maximal Perturbation: $\epsilon = 8/255$
Attack Step Size: 2/255	Threat Model: l_∞
k_1: 1.0	k_2: 0.8

Note that most of our experimental results are reported in terms of robust test accuracy, instead of the robust generalization gap. On one hand, test accuracy is the metric that we really aim to optimize in practice. On the other hand, robust test accuracy, though is not the whole picture of generalization gap, actually reflects the gap very well, especially in overparameterized regime, due to the minimization of empirical risk is relatively simple with deep models [6], even in an adversarial environment [45]. Therefore, we report only robust test accuracy following [19, 63] by default. To ensure our proposed approach actually closes the generalization gap, we report the actual generalization gap in Fig 5, and observe DEAT can beat vanilla PGD-AT by a non-trivial margin in testing performances even with sub-optimal training performances.

5.1 Hyperparameter is Impactful in Robust Generalization

Our theorem indicates learning rate and batch size can impact robust generalization via affecting diffusion. Specifically, Theorem 1 expects larger learning rate/batch size ratio would improve robust generalization. We sweep through a wide range of learning rates $0.01, 0.12, 0.014, \cdots, 0.50$, and report the adversarial testing accuracy of both vanilla PGD-AT and DEAT for a selection of learning rates in Table 2 and Figure 4. Considering the computational time for AT is already very long, decreasing batch size to improve robust generalization is simply economically prohibitive. Thus, we mainly focus on α.

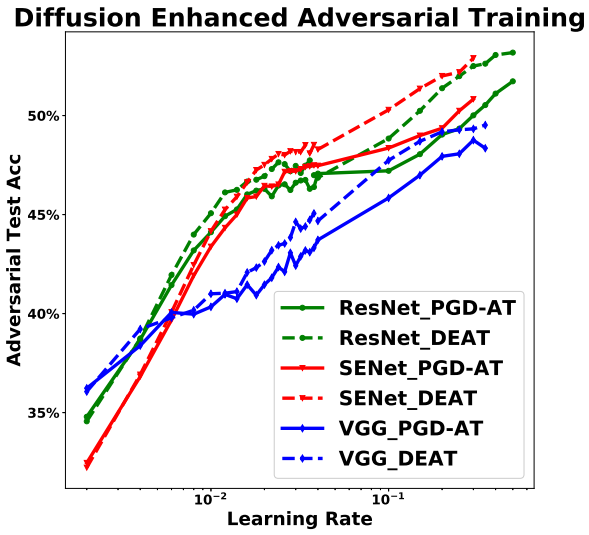


Figure 4: Adversarial testing accuracy on CIFAR10 for vanilla PGD-AT and our proposed DEAT across a wide spectrum of learning rates. The figure demonstrates a strongly positive correlation between robust generalization and learning rate. We could also observe DEAT obtains a significant improvement over PGD-AT.

Table 2 exhibits a strong positive correlation between robust generalization and learning rate. The pattern is consistent with all three architectures. Figure 4 provides a better visualization of the positive correlation.

We further do some testing on whether such correlation is statistically significant or not. We calculate the Pearson's r, Spearman's ρ, and Kendall's τ rank-order correlation coefficients [7], and the corresponding p values to investigate the statistically significance of the correlations. The procedure to calculate p values is as follows, when calculating p-value in Tables 3 and 4, we regard the data point in Table 2 as the accuracy for each α and calculate the RCC between accuracy and α and its p-value, following same procedure in [19].

We report the test result in Table 3. The closer correlation coefficient is to +1 (or −1), the stronger positive (or negative) correlation exists. If $p < 0.005$, the correlation is statistically significant [8].

Our theorem indicates ratio of learning rate and batch size (instead of batch size itself) determines generalization, which justifies the linear scaling rule in [42], i.e., scaling the learning rate up when

[6] In the setting of over-parametrized learning, there is a large set of global minima, all of which have zero training error but the test error can be very different [63, 68].

[7] They measure the statistical dependence between the rankings of two variables, and how well the relationship between two variables can be described using a monotonic function.

[8] The criterion of 'statistically significant' has various versions, such as $p < 0.05$ or $p < 0.01$. We use a more rigorous $p < 0.005$.

Table 2: Adversarial testing accuracy for both vanilla PGD-AT and DEAT. Acc_d represents the accuracy difference between diffusion enhanced adversarial training and vanilla PGD-AT, i.e., $\text{Acc}_{\text{DEAT}} - \text{Acc}_{\text{PGD-AT}}$.

Preact-ResNet [20]				SENet [23]				VGG [49]			
α	PGD-AT	DEAT	Acc_d	α	PGD-AT	DEAT	Acc_d	α	PGD-AT	DEAT	Acc_d
0.010	44.11%	45.07%	0.96%	0.010	43.38%	44.16%	0.78%	0.010	40.34%	41.00%	0.66%
0.012	44.92%	46.12%	1.20%	0.012	44.33%	45.25%	0.92%	0.012	40.97%	41.03%	0.06%
0.014	45.26%	46.25%	0.99%	0.014	45.00%	45.90%	0.90%	0.014	40.75%	41.11%	0.36%
0.018	46.21%	46.76%	0.55%	0.018	45.91%	47.25%	1.34%	0.018	40.93%	42.32%	1.39%
0.020	46.30%	46.94%	0.64%	0.020	46.45%	47.51%	1.06%	0.020	41.46%	42.08%	0.62%
0.022	45.92%	47.30%	1.38%	0.022	46.42%	47.81%	1.39%	0.022	41.81%	43.20%	1.39%
0.024	46.47%	47.64%	1.17%	0.024	46.52%	48.06%	1.54%	0.024	42.35%	43.45%	1.10%
0.028	46.24%	47.19%	0.95%	0.028	47.19%	48.20%	1.01%	0.028	43.07%	43.84%	0.77%
0.030	46.61%	47.46%	0.85%	0.030	47.19%	48.16%	0.97%	0.030	42.42%	44.63%	2.21%
0.100	47.21%	48.84%	1.63%	0.100	48.36%	50.29%	1.93%	0.100	45.84%	47.74%	1.90%
0.150	48.05%	50.24%	2.19%	0.150	48.99%	51.36%	2.37%	0.150	46.99%	48.70%	1.71%
0.200	49.04%	51.38%	2.34%	0.200	49.36%	52.00%	2.64%	0.200	47.94%	49.18%	1.24%
0.250	49.34%	51.99%	2.65%	0.250	50.24%	52.19%	1.95%	0.250	48.07%	49.28%	1.21%
0.300	50.01%	52.50%	2.49%	0.300	50.83%	52.90%	2.07%	0.300	48.76%	49.33%	0.57%
Improvement is significant especially when model is performing well (large α). **DEAT improves 1.5% on VGG, and over 2.0% on SENet and Preact-ResNet.**											

Table 3: Rank correlation coefficients (corresponding significance level) between robust generalization and learning rate. All correlation coefficient indicates a strong positive relationship (close to +1). The p values are all highly statistically significant.

Rank Correlation Coefficient	Preact-ResNet		SENet		VGG	
	PGD-AT	DEAT	PGD-AT	DEAT	PGD-AT	DEAT
Pearson's r (p-value)	0.889 (**5.5e-10**)	0.896 (**2.8e-10**)	0.711 (**1.4e-04**)	0.762 (**2.4e-05**)	0.916 (**3.3e-10**)	0.862 (**5.8e-08**)
Spearman's ρ (p-value)	0.965 (**3.4e-16**)	0.922 (**7.5e-07**)	0.998 (**<2.2e-16**)	0.982 (**<2.2e-16**)	0.988 (**<2.2e-16**)	0.992 (**8.9e-07**)
Kendall's τ (p-value)	0.907 (**3.3e-11**)	0.818 (**1.9e-12**)	0.982 (**5.7e-11**)	0.927 (**6.3e-10**)	0.932 (**1.8e-10**)	0.956 (**<2.2e-16**)

using larger batch, and maintaining the ratio between learning rate and batch size, would effectively preserve the robust generalization.

The side effect of adjusting batch size also demonstrates the necessity of our proposed approach, which could manipulate diffusion to boost generalization without extra computational burden.

5.2 DEAT Effectively Improves Robust Generalization

We compare the robust generalization of vanilla PGD-AT and DEAT in Figure 4 and Table 2.

The improvement is consistent across all different learning rates/model architectures. The improvement is even more significant when learning rate is fairly large, i.e. when the baseline is working well, in both Table 2 and Figure 4. Our proposed DEAT improves 1.5% on VGG, and over 2.0% on SENet and Preact-ResNet.

Note 1.5% to 2.0% improvement is very significant in robust generalization. It actually surpasses the performance gap between different model architectures. In Figure 4, the boosted VGG can obtain similar robust generalization compared to SENet and ResNet. [42] measures the robust generalization of virtually all popular architectures, and the range is only approximately 3%. Considering adjusting architectures would potentially include millions of more parameters and carefully hand-crafted design, our proposed approach is nearly "free" in cost.

We plot the adversarial training and adversarial testing curves (using one specific learning rate) for all three architectures in Figure 5. It is very interesting to observe that our proposed approach may not be better in terms of training performances (e.g. in ResNet and SENet), but it beats vanilla PGD-AT by a non-trivial margin in testing performances. It is safe to say that DEAT effectively control the level of overfitting in adversarial training.

We further do a t-test to check the statistical significance of the improvement and report the result in Table 4. Note the mean improvement in the table (e.g. 1.22%) is averaged across all learning rates, and does not completely reflect the extent of improvement (as we pay more attention to the improvement with larger learning rates, where the improvement is larger than 1.5%). The p-values clearly indicate a statistical significant improvement across models.

Table 4: Statistical test of significance of improvement. The p-values indicate a strongly significant improvement across all architectures.

Architecture	Statistical Significance of Improvement
Preact-ResNet	1.22% (**2.10e-09**)
SENet	1.21% (**2.11e-09**)
VGG	1.11% (**7.462e-10**)

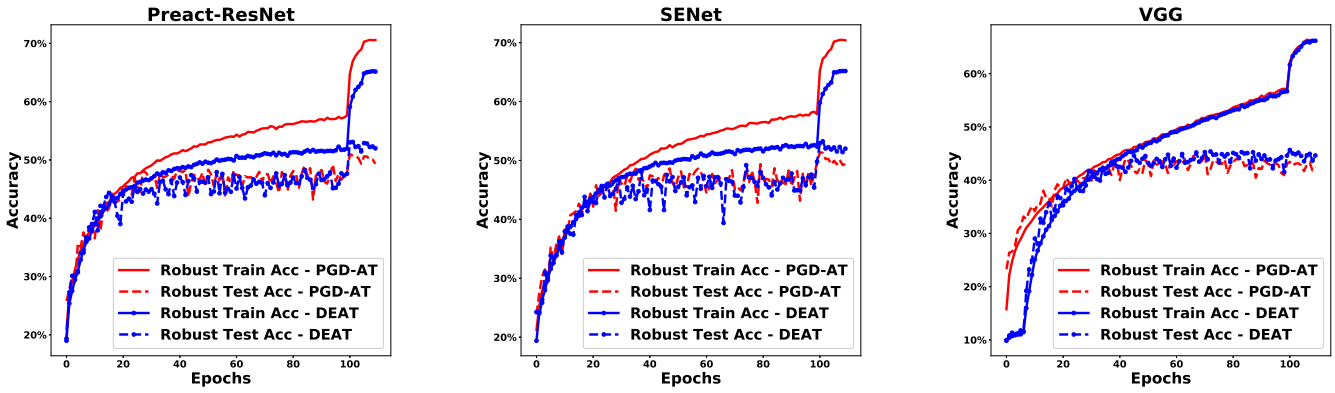


Figure 5: Adversarial training and adversarial testing curves for vanilla PGD-AT and DEAT. DEAT performs worse in training stage, but outperforms vanilla PGD-AT in testing stage. This pattern strongly attests to the effectiveness of DEAT in alleviating overfitting.

6 RELATED WORK

We summarize the related works in (A) Adversarial Training; (B) SDE Modeling of Stochastic Algorithms; and (C) Generalization and Hyperparameters in this section.

6.1 Adversarial Training

We refer readers to a comprehensive overview of adversarial attacks and defenses and references therein [7]. An incomplete list of recent advances would include [6, 22, 25, 26, 38, 44, 47, 50, 56, 58, 59, 61, 64, 65, 69, 71]. This study focuses on PGD-AT [36], the most commonly used adversarial training strategy.

6.2 SDE Modeling of Stochastic Algorithms

[33, 37] are the first works that approximate discrete-time stochastic gradient descent by continuous-time SDE. [1, 5, 14, 30] extended SDE modeling to accelerated mirror descent, asynchronous SGD, momentum SGD, and generative adversarial networks, respectively. [34] studied the SDE approximation of SGD with a moderately large learning rate, while approximation in [37] works best with infinitesimal step size. [8, 66] designed an entropy regularization and a noise injection method, respectively, motivated by the SDE characterization of SGD. [17] attempted to model adversarial training dynamics via SDE, while did not recognize the connection between dynamics and generalization error.

6.3 Generalization and Stochastic Noise

One of the goals of this paper is to theoretically and empirically study how stochastic noise impacts generalization in adversarial training. The research is mainly divided into two lines, the impact of hyperparameters on noise and directly injection of external noise.

Existing works on hyperparameters are mainly on non-adversarial training, e.g., many recent works empirically report the influence of hyperparameters in SGD, largely on b and α, and provide practical tuning guidelines. [29] empirically showed that the large-batch training leads to sharp local minima which has poor generalization, while small-batches lead to flat minima which makes SGD generalize well. [16, 28] proposed the Linear Scaling Rule for adjusting α as a function of b to maintain the generalization ability of SGD. [51, 52] suggested that increasing b during the training can achieve similar result of decaying the learning rate α.

Our generalization analysis relies on PAC-Bayesian inequalities [18, 39, 48]. [19, 35, 53–55] proved a PAC-Bayesian bound for vanilla SGD, SGD with momentum, asynchronous SGD, all in a benign environment.

The first and only systematic study on hyperparameters of adversarial training is [42], to our best knowledge. The authors carefully evaluated a wide range of training tricks, including early stopping, learning rate schedule, activation function, model architecture, optimizer and many others. However, their findings do not provide theoretical insights why certain tricks work or fail. Our study aims to bridge this gap and motivate our novel training algorithm through theoretical findings.

7 CONCLUSIONS

To our best knowledge, this paper is the first study that rigorously connects the dynamics of adversarial training to the robust generalization. Specifically, we derive a generalization bound of PGD-AT, and based on this bound, point out the role of learning rate and batch size. We further propose a novel training approach Diffusion Enhanced Adversarial Training. Our extensive experiments demonstrate DEAT universally outperforms PGD-AT by a large margin with little cost, and could potentially serve as a new strong baseline in AT research.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and helpful suggestions. This work is supported in part by the US National Science Foundation under grants 2213700, 2217071, 2106913, 2008208, 1955151.

REFERENCES

[1] Jing An, J. Lu, and Lexing Ying. 2018. Stochastic modified equations for the asynchronous stochastic gradient descent. *ArXiv* abs/1805.08244 (2018).

[2] Matan Atzmon, Niv Haim, Lior Yariv, Ofer Israelov, Haggai Maron, and Yaron Lipman. 2019. Controlling neural level sets. In *Advances in Neural Information Processing Systems*. 2032–2041.

[3] Wieland Brendel, Jonas Rauber, Alexey Kurakin, Nicolas Papernot, Behar Veliqi, Marcel Salathé, Sharada P Mohanty, and Matthias Bethge. 2018. Adversarial Vision Challenge. In *32nd Conference on Neural Information Processing Systems (NIPS 2018) Competition Track*. https://arxiv.org/abs/1808.01976

[4] Qi-Zhi Cai, Chang Liu, and Dawn Song. 2018. Curriculum Adversarial Training. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence* (Stockholm, Sweden) *(IJCAI'18)*. AAAI Press, 3740–3747.

[5] Haoyang Cao and Xin Guo. 2020. Approximation and convergence of GANs training: an SDE approach. *ArXiv* abs/2006.02047 (2020).

[6] Y. Carmon, Aditi Raghunathan, Ludwig Schmidt, Percy Liang, and John C. Duchi. 2019. Unlabeled Data Improves Adversarial Robustness. In *NeurIPS*.

[7] Anirban Chakraborty, Manaar Alam, Vishal Dey, A. Chattopadhyay, and Debdeep Mukhopadhyay. 2018. Adversarial Attacks and Defences: A Survey. *ArXiv* abs/1810.00069 (2018).

[8] P. Chaudhari, Adam M. Oberman, S. Osher, Stefano Soatto, and G. Carlier. 2017. Deep relaxation: partial differential equations for optimizing deep neural networks. *Research in the Mathematical Sciences* 5 (2017), 1–30.

[9] Jinghui Chen and Quanquan Gu. 2020. RayS: A Ray Searching Method for Hard-Label Adversarial Attack. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 1739–1747. https://doi.org/10.1145/3394486.3403225

[10] Francesco Croce and Matthias Hein. 2020. Reliable Evaluation of Adversarial Robustness with an Ensemble of Diverse Parameter-Free Attacks. In *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*. JMLR.org, Article 206, 11 pages.

[11] Hadi Daneshmand, Jonas Kohler, Aurelien Lucchi, and Thomas Hofmann. 2018. Escaping Saddles with Stochastic Gradients. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 1155–1164. https://proceedings.mlr.press/v80/daneshmand18a.html

[12] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. 2020. MMA Training: Direct Input Space Margin Maximization through Adversarial Training. In *International Conference on Learning Representations*. https://openreview.net/forum?id=HkeryxBtPB

[13] Simon S. Du and Wei Hu. 2019. Width Provably Matters in Optimization for Deep Linear Neural Networks. *CoRR* abs/1901.08572 (2019). arXiv:1901.08572 http://arxiv.org/abs/1901.08572

[14] Igor Gitman, Hunter Lang, Pengchuan Zhang, and Lin Xiao. 2019. Understanding the Role of Momentum in Stochastic Gradient Methods. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 9633–9643. http://papers.nips.cc/paper/9158-understanding-the-role-of-momentum-in-stochastic-gradient-methods.pdf

[15] I. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. *CoRR* abs/1412.6572 (2015).

[16] Priya Goyal, Piotr Dollár, Ross B. Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2017. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. *CoRR* abs/1706.02677 (2017). arXiv:1706.02677 http://arxiv.org/abs/1706.02677

[17] Haotian Gu and Xin Guo. 2021. An SDE Framework for Adversarial Training, with Convergence and Robustness Analysis. *ArXiv* abs/2105.08037 (2021).

[18] Benjamin Guedj. 2019. A Primer on PAC-Bayesian Learning. *ArXiv* abs/1901.05353 (2019).

[19] Fengxiang He, Tongliang Liu, and Dacheng Tao. 2019. Control Batch Size and Learning Rate to Generalize Well: Theoretical and Empirical Evidence. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 1143–1152. http://papers.nips.cc/paper/8398-control-batch-size-and-learning-rate-to-generalize-well-theoretical-and-empirical-evidence.pdf

[20] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778.

[21] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity Mappings in Deep Residual Networks. *ArXiv* abs/1603.05027 (2016).

[22] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. 2019. Using Pre-Training Can Improve Model Robustness and Uncertainty. *Proceedings of the International Conference on Machine Learning* (2019).

[23] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-Excitation Networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7132–7141. https://doi.org/10.1109/CVPR.2018.00745

[24] Wenqing Hu, Chris Junchi Li, Lei Li, and Jian-Guo Liu. 2018. On the diffusion approximation of nonconvex stochastic gradient descent. arXiv:1705.07562 [stat.ML]

[25] Mengdi Huai, Jianhui Sun, Renqin Cai, Liuyi Yao, and Aidong Zhang. 2020. Malicious Attacks against Deep Reinforcement Learning Interpretations. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 472–482. https://doi.org/10.1145/3394486.3403089

[26] Feihu Huang, Xidong Wu, and Heng Huang. 2021. Efficient mirror descent ascent methods for nonsmooth minimax problems. *Advances in Neural Information Processing Systems (NeurIPS)* 34 (2021), 10431–10443.

[27] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. 2017. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2261–2269.

[28] Stanisław Jastrzębski, Zac Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Amos Storkey, and Yoshua Bengio. 2018. Three factors influencing minima in SGD. https://openreview.net/forum?id=rJma2bZCW

[29] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2016. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. *CoRR* abs/1609.04836 (2016). arXiv:1609.04836 http://arxiv.org/abs/1609.04836

[30] Walid Krichene and Peter Bartlett. 2017. Acceleration and Averaging in Stochastic Descent Dynamics. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6799–6809.

[31] A. Kurakin, I. Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, A. Yuille, Sangxia Huang, Yao Zhao, Yuzhe Zhao, Zhonglin Han, Junjiajia Long, Yerkebulan Berdibekov, Takuya Akiba, Seiya Tokui, and Motoki Abe. 2018. Adversarial Attacks and Defences Competition. *ArXiv* abs/1804.00097 (2018).

[32] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the Loss Landscape of Neural Nets. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal, Canada) *(NIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 6391–6401.

[33] Qianxiao Li, Cheng Tai, and Weinan E. 2017. Stochastic Modified Equations and Adaptive Stochastic Gradient Algorithms *(Proceedings of Machine Learning Research, Vol. 70)*. PMLR, International Convention Centre, Sydney, Australia, 2101–2110. http://proceedings.mlr.press/v70/li17f.html

[34] Kangqiao Liu, Liu Ziyin, and Masahito Ueda. 2021. Noise and Fluctuation of Finite Learning Rate Stochastic Gradient Descent. arXiv:2012.03636 [stat.ML]

[35] Ben London. 2017. A PAC-Bayesian Analysis of Randomized Learning with Application to Stochastic Gradient Descent. In *NIPS*.

[36] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJzIBfZAb

[37] Stephan Mandt, Matthew D. Hoffman, and David M. Blei. 2017. Stochastic Gradient Descent as Approximate Bayesian Inference. *J. Mach. Learn. Res.* 18, 1 (Jan. 2017), 4873–4907.

[38] Chengzhi Mao, Ziyuan Zhong, Junfeng Yang, Carl Vondrick, and Baishakhi Ray. 2019. Metric Learning for Adversarial Robustness. In *NeurIPS*.

[39] David A. McAllester. 1998. Some PAC-Bayesian Theorems. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory* (Madison, Wisconsin, USA) *(COLT' 98)*. Association for Computing Machinery, New York, NY, USA, 230–234. https://doi.org/10.1145/279943.279989

[40] Y. Nesterov. 1983. A method for solving the convex programming problem with convergence rate O(1/k^2).

[41] Tianyu Pang, Kun Xu, Yinpeng Dong, Chao Du, Ning Chen, and Jun Zhu. 2020. Rethinking Softmax Cross-Entropy Loss for Adversarial Robustness. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Byg9A24tvB

[42] Tianyu Pang, Xiao Yang, Yinpeng Dong, Hang Su, and Jun Zhu. 2021. Bag of Tricks for Adversarial Training. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Xb8xvrtB8Ce

[43] B.T. Polyak. 1964. Some methods of speeding up the convergence of iteration methods. *U. S. S. R. Comput. Math. and Math. Phys.* 4, 5 (1964), 1 – 17. https://doi.org/10.1016/0041-5553(64)90137-5

[44] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and P. Kohli. 2019. Adversarial Robustness through Local Linearization. In *NeurIPS*.

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

[46] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. 2018. Adversarially Robust Generalization Requires More Data. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal, Canada) *(NIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 5019–5031.

[47] A. Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, L. Davis, G. Taylor, and T. Goldstein. 2019. Adversarial Training for Free!.

In *NeurIPS*.

[48] John Shawe-Taylor and Robert C. Williamson. 1997. A PAC Analysis of a Bayesian Estimator. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory* (Nashville, Tennessee, USA) *(COLT '97)*. Association for Computing Machinery, New York, NY, USA, 2–9. https://doi.org/10.1145/267460.267466

[49] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. http://arxiv.org/abs/1409.1556

[50] Sanchit Sinha, Mengdi Huai, Jianhui Sun, and Aidong Zhang. 2022. Understanding and Enhancing Robustness of Concept-based Models. *ArXiv* abs/2211.16080 (2022).

[51] Sam Smith and Quoc V. Le. 2018. A Bayesian Perspective on Generalization and Stochastic Gradient Descent. https://openreview.net/pdf?id=BJij4yg0Z

[52] Samuel L. Smith, Pieter-Jan Kindermans, and Quoc V. Le. 2018. Don't Decay the Learning Rate, Increase the Batch Size. In *International Conference on Learning Representations*. https://openreview.net/forum?id=B1Yy1BxCZ

[53] Jianhui Sun, Mengdi Huai, Kishlay Jha, and Aidong Zhang. 2022. Demystify Hyperparameters for Stochastic Optimization with Transferable Representations. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Washington DC, USA) *(KDD '22)*. Association for Computing Machinery, New York, NY, USA, 1706–1716. https://doi.org/10.1145/3534678.3539298

[54] Jianhui Sun, Ying Yang, Guangxu Xun, and Aidong Zhang. 2021. A Stagewise Hyperparameter Scheduler to Improve Generalization. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery amp; Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 1530–1540. https://doi.org/10.1145/3447548.3467287

[55] Jianhui Sun, Ying Yang, Guangxu Xun, and Aidong Zhang. 2023. Scheduling Hyperparameters to Improve Generalization: From Centralized SGD to Asynchronous SGD. *ACM Trans. Knowl. Discov. Data* 17, 2, Article 29 (mar 2023), 37 pages. https://doi.org/10.1145/3544782

[56] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2018. Ensemble Adversarial Training: Attacks and Defenses. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rkZvSe-RZ

[57] Jonathan Uesato, Jean-Baptiste Alayrac, Po-Sen Huang, Robert Stanforth, Alhussein Fawzi, and Pushmeet Kohli. 2019. Are Labels Required for Improving Adversarial Robustness?. In *NeurIPS*.

[58] Huaxia Wang and Chun-Nam Yu. 2019. A Direct Approach to Robust Deep Learning Using Adversarial Networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=S1lIMn05F7

[59] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. 2019. On the Convergence and Robustness of Adversarial Training. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 6586–6595. https://proceedings.mlr.press/v97/wang19i.html

[60] Yeming Wen, Kevin Luk, Maxime Gazeau, Guodong Zhang, Harris Chan, and Jimmy Ba. 2019. Interplay Between Optimization and Generalization of Stochastic Gradient Descent with Covariance Noise. *ArXiv* abs/1902.08234 (2019).

[61] Eric Wong, Leslie Rice, and J. Zico Kolter. 2020. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*. https://openreview.net/forum?id=BJx040EFvH

[62] Jingfeng Wu, Wenqing Hu, Haoyi Xiong, Jun Huan, Vladimir Braverman, and Zhanxing Zhu. 2019. On the Noisy Gradient Descent that Generalizes as SGD. In *International Conference on Machine Learning*.

[63] Lei Wu, Chao Ma, and Weinan E. 2018. How SGD Selects the Global Minima in Over-parameterized Learning: A Dynamical Stability Perspective. In *NeurIPS*. 8289–8298. http://papers.nips.cc/paper/8049-how-sgd-selects-the-global-minima-in-over-parameterized-learning-a-dynamical-stability-perspective

[64] Yihan Wu, Aleksandar Bojchevski, and Heng Huang. 2022. Adversarial Weight Perturbation Improves Generalization in Graph Neural Network. *ArXiv* abs/2212.04983 (2022).

[65] Yihan Wu, Hongyang Zhang, and Heng Huang. 2022. RetrievalGuard: Provably Robust 1-Nearest Neighbor Image Retrieval. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 24266–24279. https://proceedings.mlr.press/v162/wu22o.html

[66] Zeke Xie, Li Yuan, Zhanxing Zhu, and Masashi Sugiyama. 2021. Positive-Negative Momentum: Manipulating Stochastic Gradient Noise to Improve Generalization. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 11448–11458.

[67] Zheng Xu, Ali Shafahi, and Tom Goldstein. 2020. Exploring Model Robustness with Adaptive Networks and Improved Adversarial Training. *ArXiv* abs/2006.00387 (2020).

[68] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. https://arxiv.org/abs/1611.03530

[69] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. 2019. You Only Propagate Once: Accelerating Adversarial Training via Maximal Principle. *arXiv preprint arXiv:1905.00877* (2019).

[70] Haichao Zhang and Jianyu Wang. 2019. Defense Against Adversarial Attacks Using Feature Scattering-based Adversarial Training. In *NeurIPS*.

[71] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, E. Xing, L. Ghaoui, and Michael I. Jordan. 2019. Theoretically Principled Trade-off between Robustness and Accuracy. In *ICML*.

[72] Zhanxing Zhu, Jingfeng Wu, Bing Yu, Lei Wu, and Jinwen Ma. 2019. The Anisotropic Noise in Stochastic Gradient Descent: Its Behavior of Escaping from Minima and Regularization Effects. https://openreview.net/forum?id=H1M7soActX

A APPENDIX

A.1 Proof of Theorem 1

In this section, we give the proof of Theorem 1. We only keep primary proof procedures and omit most of the algebraic transformations.

A.1.1 Pseudocode of PGD-AT. Constrained optimization is typically transformed into unconstrained optimization in real-world deployment. After adding a regularization term $R(\delta)$ in (14), the constrained inner optimization $\delta_k = \Pi_\Delta(\delta_{k-1} + \frac{\alpha_I}{b}\sum_{j=1}^{b}\nabla_x l_{\theta_{t-1}}(\hat{x}_j + \delta_{k-1}, y_{i_j}))$ (constrained by perturbation budget set Δ) is transformed into an unconstrained optimization, with λ a hyperparameter. As perturbation budget set Δ is typically in the form $\{\delta \in \mathbb{R}^d : \|\delta\|_p \le \epsilon\}$, i.e., the L_p norm of perturbation is smaller than a constant ϵ, objective function (14) is simply a Lagrange relaxation of (3) with $R(\delta_i) = \|\delta\|_p - \epsilon$.

$$\min_{\theta\in\mathbb{R}^{d_\theta}} \max_{\delta_i, i=1,\dots,N} \frac{1}{N}\sum_{i=1}^{N} J(\theta, x_i, \delta_i), \quad \text{where} \quad J(\theta, x_i, \delta_i) = l(\theta, x_i + \delta_i, y_i) - \lambda R(\delta_i) \tag{14}$$

With the above modification, the following modified PGD-AT is also widely used (i.e., Algorithm 3).

Algorithm 3: Modified PGD-AT

Input: Loss function $J(\theta, x, \delta) = l(\theta, x + \delta, y) - \lambda R(\delta)$, initialization θ_0, total training steps T, PGD steps K, inner/outer learning rates α_I/α_O, batch size b;

for $t \in \{1, 2, ..., T\}$ **do**
  Sample a mini-batch of random examples $\zeta = \{(x_{i_j}, y_{i_j})\}_{j=1}^{b}$;
  Set $\delta_0 = 0, \hat{x}_j = x_{i_j}$;
  for $k \in \{1, ..., K\}$ **do**
    $\delta_k = \delta_{k-1} + \frac{\alpha_I}{b}\sum_{j=1}^{b}\nabla_\delta J(\theta_{t-1}, \hat{x}_j, \delta_{k-1})$;
  end
  $\theta_t = \theta_{t-1} - \alpha_O \hat{g}_{t-1}$,
  where $\hat{g}_{t-1} = \frac{1}{b}\sum_{j=1}^{b}\nabla_\theta J(\theta_{t-1}, \hat{x}_j, \delta_K)$;
end
return θ_T

A.1.2 Proof of Theorem 1. The representation of PGD-AT in SDE form is followed from [17]. Recall PGD-AT is a min-max game as in Algorithm 3. Without loss of generality, we assume $\alpha_O = \alpha_I = \alpha$. The inner loop starts with a random initialization δ_0, we have

$$\begin{aligned}\delta_1 &= \delta_0 + \frac{\alpha}{b}\sum_{j=1}^{b}\nabla_\delta J(\theta_t, \hat{x}_j, 0) = \frac{\alpha}{b}\sum_{j=1}^{b}\nabla_x l(\theta_t, \hat{x}_j)\\ \delta_2 &= \delta_1 + \frac{\alpha}{b}\sum_{j=1}^{b}\Big(\nabla_x l(\theta_t, \hat{x}_j + \delta_1) - \lambda\nabla_\delta R(\delta_1)\Big)\end{aligned} \tag{15}$$

Based on Taylor's expansion at $\delta = 0$, we could get:

$$\delta_2 = \frac{2\alpha}{b}\sum_{j=1}^{b}\nabla_x l(\theta_t, \hat{x}_j) + O(\alpha^2) \tag{16}$$

We could continue this calculation, we will see that for any K,

$$\delta_K = \frac{K\alpha}{b}\sum_{j=1}^{b}\nabla_x l(\theta_t, \hat{x}_j) + O(\alpha^2) \tag{17}$$

We only keep the $O(\alpha)$ term and higher order term is negligible. The outer loop updating dynamic will subsequently become,

$$\begin{aligned}\theta_{t+1} &= \theta_t - \frac{\alpha}{b}\sum_{j=1}^{b}\nabla_\theta J(\theta_t, \hat{x}_j, \delta_K)\\ &= \theta_t - \frac{\alpha}{b}\sum_{j=1}^{b}\nabla_\theta l(\theta_t, \hat{x}_j + \frac{K\alpha}{b}\sum_{j=1}^{b}\nabla_x l(\theta_t, \hat{x}_j) + O(\alpha^2))\\ &= \theta_t - \frac{\alpha}{b}\sum_{j=1}^{b}\nabla_\theta l(\theta_t, \hat{x}_j)\\ &\quad - \frac{K\alpha^2}{b^2}\sum_{i,j=1}^{b}\nabla_{x\theta} l(\theta_t, \hat{x}_j)\nabla_x l(\theta_t, \hat{x}_i) + O(\alpha^3)\end{aligned} \tag{18}$$

where $\nabla_{x\theta} l$ is the Jacobian matrix, where $\frac{\partial^2 l}{\partial x_j \partial \theta_i}$ is calculated.

We compute the first and second order moments of the one-step difference $D = \theta_1 - \theta_0$ in order to find a continuous-time SDE for PGD-AT, i.e., calculate $\mathbb{E}[D]$ and $\mathbb{E}[DD^T]$, where the expectation is taken with respect to the randomness of mini-batch sampling.

By some algebraic transformations, we are able to show the following results after omitting higher order term of $O(\alpha^3)$ [17],

$$\begin{aligned}\mathbb{E}[D] &= -K\alpha^2\mathbb{E}[\nabla_{x\theta} l(\theta_0, x)]\mathbb{E}[\nabla_x l(\theta_0, x)]\\ &\quad -\alpha\mathbb{E}[\nabla_\theta l(\theta_0, x)] - \frac{K\alpha^2}{b}\Big(\mathbb{E}[\nabla_{x\theta} l(\theta_0, x)\nabla_x l(\theta_0, x)]\\ &\quad -\mathbb{E}[\nabla_{x\theta} l(\theta_0, x)]\mathbb{E}[\nabla_x l(\theta_0, x)]\Big)\\ \mathbb{E}[DD^T] &= \alpha^2\mathbb{E}[\nabla_\theta l(\theta_0, x)]\mathbb{E}[\nabla_\theta l(\theta_0, x)]^T\\ &\quad + \frac{\alpha^2}{b}\mathrm{Var}_x(\nabla_\theta l(\theta_0, x))\end{aligned} \tag{19}$$

Let us notate,

$$G(\theta) \triangleq g(\theta) + \frac{K\alpha}{2b}\Big(\mathbb{E}[\|\nabla_x l(\theta, x)\|^2] - \|D(\theta)\|^2\Big) \tag{20}$$

for the sake of convenience. Here $g(\theta) \triangleq \mathbb{E}[l(\theta, x)]$, and $D(\theta) \triangleq \mathbb{E}[\nabla_x l(\theta, x)]$.

With $\mathbb{E}[D]$ and $\mathbb{E}[DD^T]$, we are able to show the following SDE could approximate the continuous-time dynamic of PGD-AT,

$$d\theta = \Big(s_0 + \alpha s_1\Big)dt + \sigma dW_t \tag{21}$$

where $dW_t = \mathcal{N}(0, Idt)$ is a Wiener process. And s_0, s_1, and σ are $-\nabla_\theta G(\theta)$, $-\frac{K}{2}\nabla_\theta(\|D(\theta)\|^2) - \frac{1}{4}\nabla_\theta(\|\nabla_\theta G(\theta)\|^2)$, and $\sqrt{\frac{\alpha}{b}}(\mathrm{Var}_x(\nabla_\theta l(\theta, x)))^{\frac{1}{2}}$, respectively.

As we assume the risk function is locally quadratic, and gradient noise is Gaussian. Suppose Hessian matrix of risk function be A, and covariance matrix of Gaussian noise be $H = BB^T$. Consider the following second-order Taylor approximation of the loss function, $l(\theta, x) = \frac{1}{2}(\theta - x)^T A(\theta - x) - \mathrm{Tr}(A)$. Gradient noise is equivalent to assuming x is sampled from $\mathcal{N}(0, H)$ (without loss of generality, we could assume the data is centered at mean 0).

We could thus computing the following,

$$g(\theta) \triangleq \mathbb{E}_x[l(\theta,x)] = \frac{1}{2}\theta^T A\theta$$
$$\nabla_x l(\theta,x) = -\nabla_\theta l(\theta,x) = A(x-\theta), \qquad \nabla_{x\theta} l(\theta,x) = A \tag{22}$$

Thus, we consequently have, the drift term is $s_0 + \alpha s_1 = -(A + (K+\frac{1}{2})\alpha A^2)\theta$, and diffusion term $\sqrt{\frac{\alpha}{b}}(\mathrm{Var}_x(\nabla_\theta l(\theta,x)))^{\frac{1}{2}} = \sqrt{\frac{\alpha}{b}}AB$, i.e., the continuous-time SDE of PGD-AT is,

$$d\theta = fdt + \sigma dW_t, \quad \text{where} \quad dW_t = \mathcal{N}(0, Idt) \quad \text{is Wiener process,}$$
$$f = -(A + (K+\frac{1}{2})\alpha A^2)\theta \qquad \text{and} \quad \sigma = \sqrt{\frac{\alpha}{b}}AB \tag{23}$$

We know this is an Ornstein-Uhlenbeck (OU) process and it has a Gaussian distribution as its stationary distribution [37]. Suppose the stationary distribution of this stochastic process has covariance Σ. Using the same technique in [19, 37], it is straightforward to verify that Σ is explicit and the norm of Σ is positively correlated with $\frac{\alpha}{b}$ and norm of B (see e.g. Section 3.2 in [37]). Specifically, its explicit form is $\Sigma = \frac{\alpha}{2b}E$, where $E = A^2H\hat{A}^{-1}$ and $\hat{A} \triangleq A + (K+\frac{1}{2})\alpha A^2$. The proof is that, the above is an OU process and thus its analytic solution is

$$\theta_t = \exp(-\hat{A}t)\theta_0 + \sqrt{\frac{\alpha}{b}}\int_0^t \exp[-\hat{A}(t-s)]ABdW_s$$

. By algebraic transformations, we could verify $\hat{A}\Sigma + \Sigma\hat{A} = \frac{\alpha}{b}A^2H$, where $H = BB^T$. When Σ is symmetric (as it is covariance matrix), we get $\Sigma = \frac{\alpha}{2b}E$.

With Lemma 2, we are ready to prove the last statement in Theorem 1. The posterior covariance is Gaussian with covariance Σ. We assume a plain Gaussian prior distribution $\mathcal{N}(\theta_0, \lambda_0 I_d)$. The density of prior and posterior distributions:

$$f_P = \frac{1}{\sqrt{2\pi\det(\lambda_0 I_d)}}\exp\Big\{-\frac{1}{2}(\theta-\theta_0)^T(\lambda_0 I_d)^{-1}(\theta-\theta_0)\Big\}$$
$$f_Q = \frac{1}{\sqrt{2\pi\det(\Sigma)}}\exp\Big\{-\frac{1}{2}\theta^T\Sigma^{-1}\theta\Big\} \tag{24}$$

We observe the upper bound $\mathcal{G}$ in Lemma 2, the only term that correlates with Σ is $\mathrm{KL}(Q||P)$. Therefore, we calculate their $\mathrm{KL}(Q||P)$ as follows:

$$\begin{aligned}\mathrm{KL}(Q||P) &= \int\Big(\frac{1}{2}\log\frac{|\lambda_0 I_d|}{|\Sigma|} - \frac{1}{2}\theta^T\Sigma^{-1}\theta \\ &\quad + \frac{1}{2}(\theta-\theta_0)^T(\lambda_0 I_d)^{-1}(\theta-\theta_0)\Big)f_Q(\theta)d\theta \\ &= \frac{1}{2}\Big\{\mathrm{tr}((\lambda_0 I_d)^{-1}\Sigma) + \theta_0^T(\lambda_0 I_d)^{-1}\theta_0 - d + \log\frac{|\lambda_0 I_d|}{|\Sigma|}\Big\} \\ &= \frac{1}{2\lambda_0}\theta_0^T\theta_0 - \frac{d}{2} + \frac{d}{2}\log\lambda_0 + \frac{1}{2\lambda_0}\mathrm{tr}(\Sigma) - \frac{1}{2}\log|\Sigma|\end{aligned} \tag{25}$$

The second statement indicates increasing diffusion would scale Σ up to $c\Sigma$, where $c > 1$, the problem is now how does $\mathrm{KL}(Q||P)$ change with $c\Sigma$. The only terms that will change are $\frac{1}{2\lambda_0}\mathrm{tr}(\Sigma) - \frac{1}{2}\log|\Sigma|$. Note that $\frac{1}{2\lambda_0}\mathrm{tr}(\Sigma)$ will become $\frac{c}{2\lambda_0}\mathrm{tr}(\Sigma)$, and $-\frac{1}{2}\log|\Sigma|$ will become $-\frac{1}{2}\log|\Sigma| - \frac{d}{2}\log c$. As d is number of parameters and is potentially extremely large, $\frac{c}{2\lambda_0}\mathrm{tr}(\Sigma) - \frac{1}{2}\log|\Sigma| - \frac{d}{2}\log c$ will be smaller than $\frac{1}{2\lambda_0}\mathrm{tr}(\Sigma) - \frac{1}{2}\log|\Sigma|$. Therefore, $\mathrm{KL}(Q||P)$ will decrease. And consequently $\mathcal{G}$ will decrease. More specifically, we plug in the explicit form of Σ, and can calculate the derivative

$$\frac{\partial \mathrm{KL(Q||P)}}{\partial r} = \frac{r}{4\lambda_0}\mathrm{tr}(E) - \frac{d}{2}\log(\frac{r}{2}) - \frac{\log|E|}{2}$$

, where $r = \frac{\alpha}{b}$. We could easily verify $\frac{\partial \mathrm{KL(Q||P)}}{\partial r} < 0$ when $d > \frac{r\cdot\mathrm{tr}(E)}{2\lambda_0}$. Thus, for sufficiently large d, larger $\frac{\alpha}{b}$ results in smaller $\mathcal{G}$. Similarly, we could show for norm of B. We complete our proof of the last statement.