Balance, Imbalance, and Rebalance: Understanding Robust Overfitting from a Minimax Game Perspective

Anonymous Author(s)
Affiliation
Address
email

Abstract

Adversarial Training (AT) has become arguably the state-of-the-art algorithm for extracting robust features. However, researchers recently notice that AT suffers from severe robust overfitting problems, particularly after the learning rate (LR) decay. In this paper, we explain this phenomenon by viewing adversarial training as a dynamic minimax game between the model trainer and the attacker. Specifically, we analyze how LR decay breaks the balance between the minimax game by empowering the trainer with a stronger memorization ability, and show such imbalance induces robust overfitting as a result of memorizing non-robust features. We validate this understanding with extensive experiments, and provide a holistic view of robust overfitting from the dynamics of both the two game players. This understanding further inspires us to alleviate robust overfitting by rebalancing the two players by either regularizing the trainer's capacity or improving the attack strength. Experiments show that the proposed ReBalanced Adversarial Training (ReBAT) can attain good robustness and does not suffer from robust overfitting even after very long training.

1 Introduction

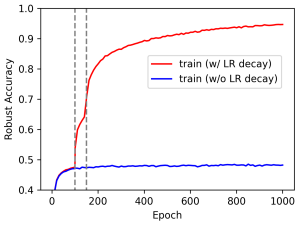
Overfitting seems to have become a history in the deep learning era. Contrary to the traditional belief in statistical learning theory that large hypothesis class will lead to overfitting, Zhang et al. [37] note that DNNs have good generalization ability on test data even if they are capable of memorizing random training labels. Nowadays, large-scale training often does not require early stopping, and it is observed that longer training simply brings better generalization [14]. However, researchers recently notice that in Adversarial Training (AT), overfitting is still a severe issue on both small and large scale data and models [26]. As the most effective defence against adversarial perturbation [2], AT solves a minimax optimization problem with training data $\mathcal{D}_{\text{train}}$ and model f_θ [22, 10]:

$$\underbrace{\min_{\theta}}_{\text{trainer } \mathcal{T}} \mathbb{E}_{\bar{x},y\sim\mathcal{D}_{\text{train}}} \underbrace{\max_{x\in\mathcal{E}_p(\bar{x})}}_{\text{attacker } \mathcal{A}} \ell_{\text{CE}}(f_\theta(x), y), \tag{1}$$

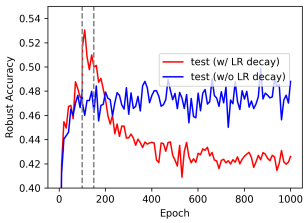
where ℓ_{CE} denotes the cross entropy (CE) loss function, $\mathcal{E}_p(\bar{x}) = \{x \mid \|x - \bar{x}\|_p \le \varepsilon\}$ denotes the ℓ_p-norm ball with radius ε. However, contrary to standard training (ST), AT suffers severely from robust overfitting (RO): after a particular point (*e.g.,* learning rate decay), its training robustness will keep increasing (Figure 1a, red line) while its test robustness decreases dramatically (Figure 1b, red line). This abnormal behavior of AT has attracted much interest. Previous works find that the AT loss landscape becomes much sharper during RO [29, 4, 33], but we are still unclear about what causes the sharp landscape. Some researchers try to explain RO through known ST phenomena, such as, random label memorization [8] and double descent [7]. However, these theories cannot explain why only AT overfits while ST hardly does.

Instead, we argue that the reason of RO should lie exactly in the *difference* between AT and ST. Sharing similar training recipes and loss functions, their key difference is that AT adopts a *bilevel*

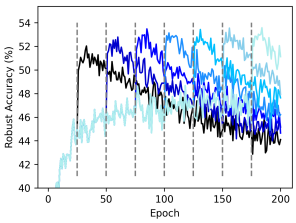
Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.



(a) training robustness during adversarial training



(b) test robustness during adversarial training



(c) test robustness when decay at different time

Figure 1: (a, b) Training/test robust accuracy on CIFAR-10 using vanilla PGD-AT [22] w/ and w/o LR decay. (c) RO happens whenever LR decays.

minimax objective, compared to the min-only ST objective. We can think of AT as a minimax game between two players (Eq. 1): the model trainer $\mathcal{T}$ (minimizing loss via updating θ) and the attacker $\mathcal{A}$ (maximizing loss via updating x), both adopting first-order algorithms in practice. With a proper configuration of both players, AT can strike a balance during which model robustness hinges on a constant level. In comparison, we also notice that when one player (typically the trainer) changes its original strategy (*e.g.,* decaying learning rate), the original balance of the game is broken and RO also occurs subsequently. We further verify this observation through a controlled experiment. As shown in Figure 1c, if we do not perform LR decay, RO will not occur till the end; but whenever we perform LR decay, RO will immediately happen. This provides strong evidence showing that breaking the balance between minimax layers can cause robust overfitting.

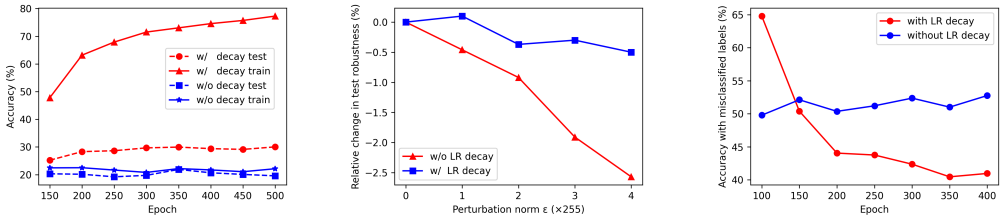
To gain further insights into this balance-imbalance process, in Section 3, we provide a generic explanation for RO from the perspective of feature learning. Specifically, when the balance breaks after the LR decay, the trainer with a smaller LR is endowed with a stronger local fitting ability capable of memorizing adversarial examples such that the attacker can no longer succeed in attack. However, during this process, the model learns false mapping of non-robust features contained in the training adversarial examples, which opens shortcuts for the test-time attacker and induces large test robust error. Accordingly, we design a series of experiments that align well with our analysis.

The minimax game perspective also indicates a simple fix to RO, that is to adjust the ability of the two minimax players to a new balance by modifying their strategies, namely, *rebalancing*. In this work, we explore three new strategies for rebalancing the game after LR decay: bootstrapping objective, smaller LR decay factor, and stronger attacker. The former two restrict the local fitting ability of the trainer while the third enhances the ability of the attacker. We show that all three strategies help mitigate robust overfitting to a large extent. Based on these insights, we propose ReBalanced Adversarial Training (ReBAT) that achieve superior robustness with a neglectable (if any) degree of robust overfitting, even if we train it further to 500 epochs. Meanwhile, ReBAT attains superior robustness (both best-epoch and final-epoch) on benchmark datasets including CIFAR-10, CIFAR-100, and Tiny-ImageNet. This suggests that with appropriately balanced training strategies, AT could also enjoy similar "train longer, generalize better" property like ST. To summarize, our main contributions are:

- We propose to understand adversarial training (AT) as a dynamic minimax game between two actual players: the model trainer $\mathcal{T}$ and the attacker $\mathcal{A}$. We characterize the balance status of the minimax game from the perspective of non-robust features.
- We show robust overfitting is caused by the imbalance between two minimax players induced by the LR decay, and explain this process as the false memorization of non-robust features. We design a series of experiments to verify this explanation and provide a holistic characterization of robust overfitting from the views of both players.
- From our dynamic perspective, we propose three effective approaches that mitigate robust overfitting by re-striking a balance between the model trainer and the attacker. Experiments show that our proposed ReBAT method largely alleviates robust overfitting while attaining good robustness on different network architectures and benchmark datasets.

2 Preliminaries

Adversarial Attack. Given an image classifier f, adversarial attack [30] is proposed to perturb the image by an imperceptible noise such that the image is misclassified by f. A well-known white-box



(a) memorization of non-robust features (b) change in test robustness with stronger non-robust features (ε). (c) target-class non-robust features

Figure 2: (a) Almost non-generalizable non-robust features in training data is memorized only after LR decay. (b) With LR decay, injecting stronger non-robust features (larger ε) induces severer degradation in test robustness, while the degradation is less obvious w/o LR decay. (c) After LR decay, test adversarial examples contains less and less target-class non-robust features.

attacker $\mathcal{A}$ is PGD (Projected Gradient Descent) [22], which uses multi-step gradient ascent steps to maximize the cross entropy loss while projecting intermediate steps to the feasible region:

$$\tilde{x}_{k+1} = x_k + \alpha \nabla_{x_k} \ell_{\mathrm{CE}}(f_\theta(x_k), y),\ x_{k+1} = \mathcal{P}_{\mathcal{E}_p(\bar{x})}(\tilde{x}_{k+1}),\ k = 0, \ldots, K-1. \tag{2}$$

where $x_0 = \bar{x} + \varepsilon, \varepsilon \sim U[0, \varepsilon]$, and $\mathcal{E}_p(\bar{x}) = \{x | x \in [0,1]^d, \|x - \bar{x}\| \leq \varepsilon\}$ is the feasible region.

Adversarial Training. To counter misclassification caused by the attacker, adversarial training (AT) [30, 10, 22, 26] aims to build a robust classifier through the minimax objective in Equation 1. In practice, AT iterates between two steps: 1) solve the inner-loop maximization *w.r.t.* input x using a few (*e.g.,* 10) PGD steps (Eq. 2) to generate a minibatch of adversarial examples $\{\hat{x}\}$, and then 2) solve the outer-loop minimization *w.r.t.* NN parameters θ with SGD as the trainer $\mathcal{T}$. Therefore, AT can be seen as a minimax game between two players (specifically, two first-order optimization algorithms), the attacker $\mathcal{A}$ and the trainer $\mathcal{T}$, with the objective function $\ell_{\mathrm{CE}}(f_\theta(\hat{x}), y)$ [22].

Robust and Non-robust Features. The arguably most prevailing understanding of adversarial training is the framework proposed by Ilyas et al. [15], where they regard training data as a composition of robust and non-robust features that are both useful for classification. Consider a binary classification task, a feature $f : \mathcal{X} \to \mathbb{R}$ is categorized as 1) a **useful feature** if $\mathbb{E}_{\bar{x},y}[y \cdot f(x)] \geq \rho$; 2) a **robust feature** if it is useful for $\rho > 0$ and $\mathbb{E}_{\bar{x},y} \inf_{x \in \mathcal{E}_p(\bar{x})}[y \cdot f(x)] \geq \gamma$; 3) a **non-robust feature** if it is useful for $\rho > 0$ and not robust for any $\gamma \geq 0$. We can also choose a threshold γ to determine its belonging class since there is no clear dividing line between robust and non-robust features in practice. According to Ilyas et al. [15], standard training (ST) adopts non-robust features for better accuracy, which also leads to adversarial vulnerability. Instead, AT only uses robust features and becomes resistant to adversarial attack. Our work also adopts the notion of non-robust features. But compared with their framework which mainly considers the ideal, static solution to the minimax problem (a learned robust classifier), our work zooms into the dynamic interplay of the two actual players (how a classifier learns the robust & non-robust features in AT) to understand RO.

3 A Minimax Game Perspective on Robust Overfitting

In this section, we propose a minimax game perspective to understand the cause of robust overfitting. We first show the minimax game in Equation 1 achieves a balance when the trainer $\mathcal{T}$ is not capable of memorizing the non-robust features, and then demonstrate through several verification experiments that the break of this balance, which leads to memorization of the non-robust features, actually induces shortcuts for test-time adversarial attack that results in RO.

3.1 AT Strikes a Balance when Non-robust Features Cannot be Memorized

First, we look at the balanced status of adversarial training. From Figure 1a (blue line), we observe that without LR decay, the two players can maintain a constant robustness level throughout training. Inside the equilibrium, the attacker $\mathcal{A}$ injects non-robust features from a different class y' to x in order to misclassify an input (x, y) [15] [1], while the trainer can only correctly classify a part of them using robust features. In other words, the game can strike a balance when there is a limit of the trainer's fitting ability that after a certain robustness level, it can no longer improve robustness by fitting the non-robust features that the attacker can modify (*i.e.,* non-robust features are not memorizable).

[1] We refer the reader to Appendix A for simple a verification and to Fowl et al. [9] for a further discussion.

To verify this point, we collect misclassified adversarial examples (containing non-robust features *w.r.t.* y' if misclassified to y') on an early checkpoint (60th epoch) and evaluate their accuracy on a later checkpoint ($\geq$150th epoch) *w.r.t.* their true labels y (detailed settings in Appendix B.2). If the adversarial non-robust features can be memorized, we would expect high accuracy on these adversarial examples on the later checkpoints. However, in practice, the blue solid line in Figure 2a shows the accuracy is still relatively low ($< 25\%$), suggesting that these non-robust features are almost not memorized under the same training configuration without LR decay.

3.2 Balance $\rightarrow$ Imbalance: Robust Overfitting as a Result of Imbalanced Minimax Game

Next, we look at how LR decay breaks the balance of the minimax game. As pointed out by the theoretical work [21], small LR favors easy-to-generalize and hard-to-fit patterns, which correspond to non-robust features in AT [15]. Thus, with a smaller LR, the trainer becomes more capable of memorizing these features by drawing more complex decision boundaries in a small local region. Indeed, we observe a significant increase in training robust accuracy in Figure 1a. Moreover, we can see from the red line in Figure 2a that most of the pre-LR-decay adversarial examples (with adversarial non-robust features) are correctly classified after LR decay, indicating stronger memorization of non-robust features. Therefore, we claim that *LR decay induces an imbalance between the attacker $\mathcal{A}$ and the trainer $\mathcal{T}$*: the boosted local fitting ability of $\mathcal{T}$ overpowers the attack ability of $\mathcal{A}$ that it is capable of memorizing the adversarial non-robust features that $\mathcal{A}$ can leverage.

Meanwhile, as the red dashed line in Figure 2a implies, it turns out that these memorized non-robust features in training data do not generalize to improve test robustness, resulting in a large robust generalization gap (see Figure 1). Next, we further reveal how such memorization harms test robustness by inducing robust overfitting.

3.2.1 How Imbalance Leads to Robust Overfitting

Under the imbalanced minimax game, the stronger trainer pushes up training robustness by memorizing the non-robust features in the training adversarial examples. However, roughly speaking, the trainer's (overly) strong fitting ability actually learns false mappings of non-robust features which induce shortcuts for test-time adversarial attack that lead to robustness degradation.

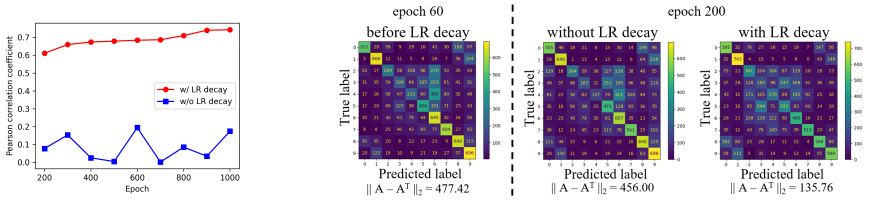
1) Model Learns False Mapping of Non-robust Training Features after Decay. Consider a training adversarial example x' from class y that is misclassified to class $y' \neq y$ before LR decay. According to [15], the non-robust feature g of x' belongs to class y', *i.e.,* $g \in \mathcal{F}_{y'}$. After LR decay, the trainer $\mathcal{T}$ becomes more capable of memorizing with a smaller LR and it successfully learns to map this adversarial example x' to its correct label y. Accordingly, as we argued above, the non-robust feature $g \in \mathcal{F}_{y'}$ in x is also (falsely) mapped to class y, *i.e.,* $\varphi(g) = y$.

2) False Non-robust Mapping Opens Shortcuts for Test-time Adversarial Attack. Now we consider a clean test sample $\hat{x}$ whose true label is y'. Before LR decay, in order to misclassify it, the attacker $\mathcal{A}$ has to add non-robust features *from a different class* , say $g' \in \mathcal{F}_{\hat{y}'}, \hat{y}' \neq y'$. However, after LR decay, because of the existence of false non-robust mapping φ, the attacker $\mathcal{A}$ can simply add the non-robust feature from the same class, *i.e.,* $g \in \mathcal{F}_{y'}$ such that $\varphi(g) = y, y \neq y'$ to misclassify $\hat{x}$ to y. In other words, the injected non-robust features do not even have to come from a different class to misclassify $\hat{x}$, which are much easier for the attacker to find when starting from $\hat{x}$ (compared with non-robust features g' from other classes). Therefore, the attacker will have a larger attack success rate with the help of these *falsely memorized non-robust features*, as they create easy-to-find shortcuts for test-time adversarial attack.

3.2.2 Verification

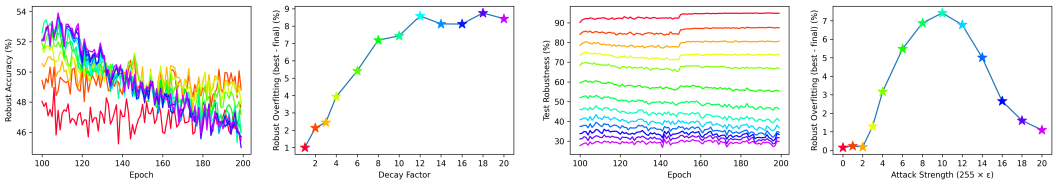
To further validate our explanation above, we design a series of experiments regarding the influence of memorization of the adversarial non-robust features after LR decay (Verification I & II). Guided by our explanation, we also observe some new intriguing phenomena of RO, which also help verify our minimax perspective (Verification III & IV). We refer the readers to Appendix B.2 for details of these experiments.

Verification I: More non-robust features, worse robustness. To study the effect of non-robust features, we deliberately add non-robust features of different strengths to the clean data (a larger ε indicates stronger non-robust features) and perform AT on these synthetic datasets. As shown in Figure 2b, we can see that 1) with LR decay, stronger non-robust features induce severer test robustness degradation, and 2) it is much less severe without LR decay. This directly shows that the memorization of non-robust features induced by LR decay indeed hurts test robustness a lot.



(a) bilateral class correlation (b) confusion matrix symmetry

Figure 3: (a) Increasingly strong correlation between training-time $y \to y'$ misclassification and test-time $y' \to y$ misclassification changes (due to RO). (b) Test-time confusion matrix of adversarial examples becomes symmetric with LR decay.



(a) different LR decay factor (b) different attack strength

Figure 4: (a) RO becomes increasingly severe as larger LR decay factor is adopted. (b) Varying the attack strength from extremely weak to extremely strong, the degree of RO first increases and then decreases after a turning point around $\varepsilon = 10/255$.

Verification II: Vanishing target-class[2] features in test adversarial examples. In our example above, the added non-robust feature g comes from the same class y' as $\hat{x}$, so the resulting adversarial example $\hat{x}'$ actually contains little features of class y, even if misclassified to y. Figure 2c shows that the test adversarial examples generated after LR decay indeed have less and less target-class features as RO becomes more and more severe, which proves the shortcuts indeed exist to make test-time adversarial attack easier.

Verification III: Bilateral Class Correlation. Our theory suggests that if RO happens, the training misclassification from class $y \to y'$ (before decay) will induce an increase in test misclassification from class $y' \to y$ (after decay), as the $y \to y'$ false non-robust mappings create $y' \to y$ shortcuts. We verify this in Figure 3a, in which we observe that this bilateral correlation indeed consistently increases after LR decay, while remaining very low (nearly no correlation) without decay.

Verification IV: Symmetrized Confusion Matrix. As a result of the bilateral correlation, we can deduce a very interesting phenomenon that the $y \to y'$ and $y' \to y$ misclassification will have more similar error rates, which means that the confusion matrix of test robustness will become more symmetric. Indeed, as shown in Figure 3b, the confusion matrix (denoted as A) becomes much more symmetric (smaller distance between A and $A^\top$) after LR decay than without LR decay.

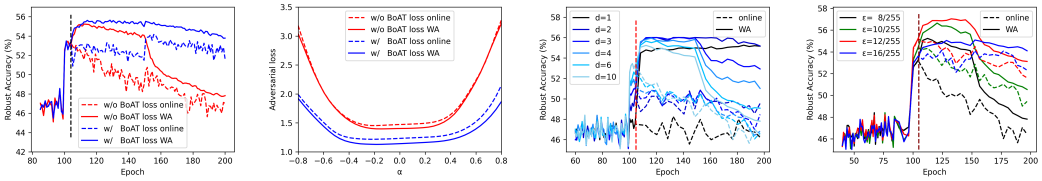
3.3 A Holistic View of Robust Overfitting from the Minimax Game Perspective

Based on our understanding of RO through the lens of the minimax game, we give a holistic view of RO, revealing the influence of both the trainer and the attacker.

Trainer. We first fix the attacker strength to be PGD-10 attack with $\varepsilon = 8/255$ and change the LR decay factor d from 1 (w/o LR decay) to 20 at epoch 100, and we observe that stronger LR decay induces a monotonically increasing degree of RO (Figure 4a). This agrees well with our explanation that smaller LR induces a severer imbalance and can memorize more non-robust features that eventually harm test robustness.

Attacker. We then fix the training configuration and vary the attack strength from PGD-0 with $\varepsilon = 0/255$ (equivalent to standard training (ST)) to PGD-25 with $\varepsilon = 20/255$. Different from previous beliefs that RO will become severer with stronger attack [34], we find an intriguing inverted U curve in Figure 4b: the degree of RO firstly rises with stronger attackers, peaks at around $\varepsilon = 10/255$, and decrease under even stronger attackers ($\varepsilon > 10/255$). Our theory can naturally explain this phenomenon: 1) when using a weak attacker (small ε), training robustness is very high and there are only a few non-robust features. Thus memorizing them only leads to slight or no RO; 2) a very strong

[2]By target-class we mean the class to which an adversarial examples is misclassified.



(a) robustness comparison (b) flatness comparison (c) LR decay factor (d) attack strength

Figure 5: Mitigating RO from three different techniques. (a, b) The purposed BoAT loss successfully boosts model robustness and mitigates RO by bootstrapping loss landscape flatness between online and WA model. (c) Small LR decay factor leads to better best-epoch robustness of WA model with only slight RO. (d) Stronger training attacker also helps mitigate RO and boost robustness.

attacker (e.g., $\varepsilon > 10/255$) can also counter the strong fitting ability of the post-decay trainer, which also prevents the memorization of non-robust features and alleviates RO.

In summary, the conditions for RO are two folds: 1) the attacker $\mathcal{A}$ should be strong enough to generate enough non-robust features (which explains why AT overfits while ST does not); and 2) the trainer $\mathcal{T}$ should be stronger than $\mathcal{A}$ to memorize the non-robust features (which explains how LR decay induces RO). This understanding also suggests that we can alleviate RO by restoring the balance between the minimax players, as we explore in the next section.

4 Mitigating Robust Overfitting by Rebalacing the Minimax Game

Based on our understanding of RO from the imbalanced minimax game perspective, we further investigate how to alleviate RO by restoring the balance. In particular, we can either weaken the trainer's fitting ability (Sections 4.1), or strengthen the attacker's attacking ability (Section 4.2). We observe that strategies from both directions can prevent the model trainer $\mathcal{T}$ from fitting non-robust features too quickly and too adequately, thus can mitigate RO.

4.1 Trainer Regularization

From the trainer's side, one approach to rebalance the game is to regularize the trainer's local fitting ability after LR decay, *e.g.,* by enforcing better landscape flatness. In this way, we could prevent the learner from drawing complex decision boundaries and memorizing non-robust features. Among existing approaches targeting at this, weight averaging (WA) [16] taking the (moving) average of history model weights is effective and efficient that it brings neglectable training overhead, so we focus on WA in this work. Along the online model f_θ that is adversarially trained, we also maintain a WA model f_φ that is an exponential moving average (EMA) of the online parameters $\varphi \leftarrow \gamma \cdot \varphi + (1-\gamma) \cdot \theta$, where $\gamma \in [0, 1]$ is the decay rate.

Bootstrapping. We notice that when the online model f_θ deteriorates, the WA model, although flatter [16], will eventually deteriorate as a moving average of f_θ (Figure 5a). Inspired from the latent bootstrap mechanism in self-supervised learning [11], we align the predictions of f_φ and f_θ to improve the flatness of the online model f_θ simutaneously. Specifically, given an adversarial example x generated by PGD attack using vanilla AT loss (Eq. 1), we have

$$\ell_{\text{BoAT}}(x, y; \theta) = \ell_{\text{CE}}(f_\theta(x), y) + \lambda \cdot \text{KL}\left(f_\theta(x) \| f_\varphi(x)\right), \tag{3}$$

where λ is a coefficient balancing the CE loss (for better training robustness) and the KL regularization term (for better landscape flatness). In one direction, the term provides more fine-grained supervision from the flatter WA model that helps avoid overfitting to the training labels [23, 4], leading to better loss landscape flatness and robustness of the online model; in return, a better online model will further improve the flatness of the WA model. Since this regularization encourages loss landscape flatness of both f_φ and f_θ in a bootstrapped fashion, we name it Bootstrapped Adversarial Training (BoAT) loss.

We further verify the effectiveness of the proposed BoAT loss in Figure 5a, in which both the online model and the WA model suffer significantly less from RO and achieve higher best-epoch robustness compared with using the vanilla AT loss (see Appendix C.1 for details). Moreover, we compare their 1-D loss landscapes [18] at the last checkpoints, and Figure 5b demonstrates that both the online model and the WA model trained with our BoAT loss enjoy much better flatness, which implies weak memorization of non-robust features under the rebalanced game that explains the alleviation in RO.

Smaller LR Decay Factor. As studied in Section 3, LR decay has a dramatic influence on RO and reducing the LR decay factor can largely ease RO (Figure 4a). Therefore, we also adopt a smaller LR

decay (together with the BoAT loss) to further regularize the trainer's fitting ability. To demonstrate its effectiveness, even when we go back to the vanilla AT loss, Figure 5c shows the WA model obtained with a very small decay rate $d = 2$ not only has very slight RO but also attains better best test robustness than $d = 10$. We also remark how the WA model benefits from continuous supply of good individuals, which couldn't be guaranteed under strong RO, also throws new meaningful light on our effort in mitigating RO.

4.2 Stronger Training-time Attacker

From our minimax game perspective, beside directly restricting the fitting ability of the trainer $\mathcal{T}$, another natural approach is to strengthen the attacker $\mathcal{A}$'s ability to counter it after LR decay to help strike a new balance between the two players. Therefore, we apply four different training perturbation budgets after decay, $\varepsilon = 8/255, 10/255, 12/255, 16/255$ (with fixed step size $\alpha = 2/255$ and increasing PGD steps $k \approx 10 \cdot \varepsilon/(8/255)$), where $\varepsilon = 8/255$ is the default budget used before LR decay and in test-time attack. As can be seen from Figure 5d, the adopted stronger attacker indeed suppress RO for both the online and WA models, thanks to the hard-to-memorize adversarial non-robust features they attached to the clean training data; and again we can see how WA benefits from the alleviation of RO, especially for $\varepsilon = 12/255$ that even hits $> 57\%$ WA model robustness against PGD-20 attack (see also Appendix B.3). Besides, a too strong training attacker may prevent the $\mathcal{T}$ from learning useful robust features (*e.g.,* $\varepsilon = 16/255$) that harms robustness, implying that we should carefully choose the attacker strength while rebalancing the minimax game via this approach.

4.3 Overall Approach

Above, we have introduced three effective techniques to mitigate RO from different perspectives: model regularization, learning rate decay, and attacker strength. Since stronger attacker often induces a large decrease in natural accuracy [1] (we additionally study this trade-off in Appendix B.3) and needs more careful balancing in order not to harm robustness as discussed above, we design two versions of our final approach: ReBalanced Adversarial Training **(ReBAT)**, which combines the BoAT loss with smaller LR decay factor; and **ReBAT[strong]**, which combines all the three techniques.

5 Revisting Previous Works from the Minimax Game Perspective

From a dynamic minimax game perspective of AT, we build a generic understanding of RO as a result of memorizing non-robust features under an imbalanced game, and introduce three strategies to rebalance the minimax game to mitigate RO. In fact, we notice that this perspective can also enable a unified understanding that existing approaches to alleviate RO can be perceived as rebalancing techniques developed from the following three dimensions.

Data Regularization. Since RO is induced by non-robust features in the training data, one way is to corrupt those features with various input transformations. Since non-robust features often correspond to local and high-frequency patterns in images [15], stronger augmentations like CutMix [35] and IDBH [19] will help mitigate RO [26, 19]. A complementary approach is to correct the *supervision* of the adversarial examples accordingly to avoid learning false label mapping (Section 3.2), *e.g.,* the soft labels adopted in Dong et al. [7], and the distillation-alike methods in KD+SWA [4] and TE [8].

Training Regularization. As discussed in Section 3.3, a necessary condition of RO is that the model trainer is stronger than the attacker, so restricting the trainer's ability in fitting non-robust features helps suppress RO. One approach is to directly promote model flatness, *e.g.,* AWP [33], MLCAT$_{\text{WP}}$ [34], EMA [25], SWA [4], AdvLC [20] and our BoAT loss (Section 4.1). Another approach is to adjust the learning rate schedule to avoid dramatic increase in $\mathcal{T}$'s ability, which is previously studied by Rice et al. [26] and Wang et al. [31]. While RO is indeed delayed by some mild LR decay schedules attempted, it still happens as $\mathcal{T}$ eventually becomes strong. Particularly, we show that simply piecewise decay with a small LR decay factor can be significantly helpful for both better robustness and less severe RO.

Stronger Training Attacker. Previous attempts in changing attacker strength, including Cai et al. [3], Qin et al. [24] and Wang et al. [32], mainly start training from a *weaker* attacker (*e.g.,* $\varepsilon = 0$), and gradually lift its strength to the standard level ($\varepsilon = 8/255$), which is later pointed out by Pang et al. [23] that these methods are hardly useful when measured by AA robustness. OA-AT [1] introduces a stronger attacker *from the beginning* but aiming to achieve robustness against stronger attacks without much concern in RO. Instead, our method is designed from a minimax game perspective that adopts a *stronger* training attacker ($\varepsilon > 8/255$) *after LR decay* in order to counter RO. Our method shows clear benefits on improving best AA robustness as well as mitigating RO.

Table 1: Comparing our method with several training methods on CIFAR-100 and Tiny-ImageNet under the perturbation norm $\varepsilon_\infty = 8/255$ based on the PreActResNet-18 architecture.

Method	PreActResNet-18						WideResNet-34-10					
	Natural		PGD-20		AutoAttack		Natural		PGD-20		AutoAttack	
	best	final	best	final	best	final	best	final	best	final	best	final
PGD-AT	81.61	84.67	52.51	46.20	47.51	42.31	86.38	87.04	56.62	48.54	51.94	45.87
TRADES	80.45	82.89	52.32	49.39	48.09	46.40	84.23	84.89	56.02	47.38	52.76	45.62
WA	83.50	**84.94**	55.05	47.60	49.89	43.83	**87.66**	87.12	56.62	49.12	52.65	46.34
KD+SWA	**84.06**	84.48	53.70	53.68	49.82	49.37	87.45	**88.21**	56.29	55.76	53.59	53.55
PGD-AT+TE	82.04	82.59	54.98	53.54	50.12	49.09	85.97	85.77	56.42	53.40	52.88	50.56
AWP	81.11	80.62	55.60	55.03	50.09	49.85	85.63	85.61	58.95	59.05	53.32	53.38
WA+CutMix	80.24	80.30	56.02	55.95	49.67	49.59	85.08	87.95	60.41	59.36	55.11	53.60
MLCAT_{WP}	-	-	**58.48**	**57.65**	50.70	50.32	-	-	60.34	57.79	51.90	49.76
ReBAT	81.86	81.91	56.36	56.12	51.13	51.22	85.25	85.52	59.59	59.29	54.78	54.80
ReBAT[strong]	78.71	78.85	56.68	56.70	**51.49**	**51.39**	81.65	81.59	59.81	59.88	54.80	54.91
ReBAT+CutMix	79.02	78.95	56.15	56.16	50.18	50.22	86.28	87.01	**61.33**	**61.24**	**55.75**	**55.72**

Table 2: Comparing our method with several training methods on CIFAR-100 and Tiny-ImageNet under the perturbation norm $\varepsilon_\infty = 8/255$ based on the PreActResNet-18 architecture.

Method	CIFAR-100						Tiny-ImageNet					
	Natural		PGD-20		AutoAttack		Natural		PGD-20		AutoAttack	
	best	final	best	final	best	final	best	final	best	final	best	final
PGD-AT	55.94	56.39	29.74	22.47	24.84	19.80	45.29	48.70	21.86	17.54	17.29	14.19
TRADES	55.04	57.09	29.17	26.36	24.21	23.06	48.32	47.59	22.25	21.14	16.55	15.99
WA	**57.26**	**58.40**	30.35	23.52	25.83	20.51	49.56	49.35	24.24	19.30	19.47	15.56
KD+SWA	57.17	58.23	29.50	29.33	25.66	25.65	**50.28**	**50.67**	24.37	24.30	19.46	19.51
PGD-AT+TE	56.41	57.26	30.90	29.07	25.84	24.99	46.71	50.50	22.76	19.26	18.02	16.13
AWP	54.10	54.78	30.47	30.15	25.16	25.01	43.54	43.33	23.75	23.55	18.12	18.07
WA+CutMix	56.73	57.17	31.70	31.71	26.43	26.08	46.48	46.48	24.80	24.72	18.87	18.87
MLCAT_{WP}	-	-	31.27	30.57	25.66	25.28	-	-	-	-	-	-
ReBAT	56.13	56.79	32.52	32.02	**27.60**	27.29	48.28	47.92	24.93	24.14	20.54	19.79
ReBAT[strong]	52.44	54.65	32.49	32.11	27.49	**27.33**	45.10	44.12	25.23	24.52	**20.67**	**20.51**
ReBAT+CutMix	56.05	56.02	**32.71**	**32.57**	27.08	27.20	46.47	46.62	**25.28**	**25.28**	19.40	19.35

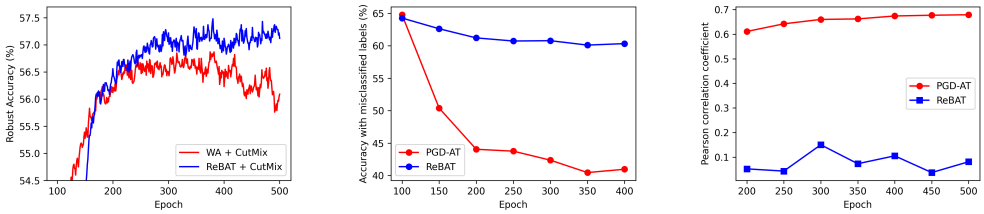
6 Experiments

In this section, we evaluate the effectiveness of the proposed ReBAT and ReBAT[strong] on several benchmark datasets using different network architectures.

Setup. We consider the classification tasks on CIFAR-10, CIFAR-100 [17], and Tiny-ImageNet [6] with the PreActResNet-18 [12] and WideResNet-34-10 [36] architectures. Besides the vanilla PGD-AT [22], we include the following baseline methods for alleviating robust overfitting: WA [16], KD+SWA [4], PGD-AT+TE [8], AWP [33], WA+CutMix [25] and MLCAT_{WP} [34]. During evaluation, we use PGD-20 [22] and AutoAttack (AA) [5] for the adversarial attack methods. We train each model for 200 epochs and record the checkpoint on which the validation set achieves the highest PGD-20 robustness and the final checkpoint for evaluation on the test set. Please refer to Appendix B.4 and B.5 for details and Appendix C for additional results.

6.1 Results on Benchmark Datasets

Performance across Different Networks. In Table 1, we evaluate ReBAT against previous AT variants on two popular backbone networks: PreActResNet-18 and WideResNet-34-10. We can see that without strong data augmentations like CutMix, ReBAT attains the best robustness on both models and outperforms all baseline methods against AutoAttack (AA), arguably the strongest white-box attacker. Meanwhile, ReBAT almost shows no RO on both PreActResNet-18 (51.13% best and 51.22% final AA robustness), and WideResNet-34-10 (54.78% best and 54.80% final AA robustness). For ReBAT[strong], it indeed results in even better best and final robustness than ReBAT on both models also with nearly perfect ability in suppressing RO, though at the cost of degradation in natural accuracy as discussed in Section 4.2. The results verify the effectiveness of our proposed method as well as the importance of rebalancing the minimax game.



(a) training longer with CutMix (b) target-class non-robust features (c) bilateral class correlation

Figure 6: Empirical understandings on ReBAT. (a) ReBAT can benefit from training longer (with CutMix). (b) Compared to PGD-AT, test adversarial examples still have to contain fairly many target-class non-robust features to successfully attack a model trained with ReBAT. (c) ReBAT does not result in strong bilateral class correlation as in PGD-AT.

Strong Augmentations. WideResNet-34-10 is more prone to overfit as it has larger capacity, and existing WA+CutMix strategy [25] still overfits by more than 1.5% (55.11% best and 53.60% final AA robustness). In comparison, ReBAT+Cutmix achieves 55.75% best robustness (chosen according to PGD) and 55.72% final robustness, showing that this combination indeed alleviates RO very effectively on the relatively large WideResNet-34-10 while maintaining very high robustness. In comparison, CutMix brings little improvement on PreActResNet-18, mainly because this model has lower capacity that still underfits under strong augmentations, and it requires longer training to reach higher performance. Even though, ReBAT+CutMix still outperforms WA+CutMix by 0.63% AA robustness at the conclusion of 200 epochs of training.

Performance across Different Datasets. We also conduct experiments on two additional benchmark datasets, CIFAR-100 and Tiny-ImageNet, that are more complex than CIFAR-10. As shown in Table 2, our ReBAT, ReBAT[strong] and ReBAT+Cutmix still achieve the highest best and final robustness, demonstrating its scalability to larger scale datasets. Similarly, CutMix does not help much as it often leads to underfitting with stronger regularization effects within relatively few training epochs.

6.2 Empirical Understandings

ReBAT Can Benefit from Longer Training. Table 1 suggests that when CutMix is applied to a PreActResNet-18 model, it demands longer training to fully unleash the beast. This motivates us to examine the effect of longer training, *e.g.,* 500 epochs. As shown in Figure 6a, longer training with WA+CutMix brings little improvement on final robustness (49.59% (200 epochs) *v.s.* 49.70% (500 epochs) under AA). Instead, ReBAT+CutMix achieves higher best robustness and the robustness keeps improving along training. In particular, the additional 300 epochs improve the final AA robustness of ReBAT+CutMix from 50.22% to 51.32%. This provides strong evidence on the effectiveness of ReBAT against RO and also shows that combining ReBAT with strong data augmentations can benefit from long training as in standard training.

Verification on Robust Overfitting Experiments. Beyond its success, we empirically study how ReBAT manages to suppress RO through our minimax game perspective. Following the experiments in Section 3.2.2, we compare PGD-AT and ReBAT in terms of 1) target-class non-robust features in test adversarial examples and 2) the strength of bilateral class correlation. We can see that the test adversarial examples still have to contain much target-class non-robust features to be misclassified (Figure 6b) and there is no longer a strong bilateral class correlation (Figure 6c). This confirms that ReBAT indeed successfully prevents the model from learning false mappings of non-robust training features which creates shortcuts for test-time attack.

7 Conclusion

In this paper, we investigate the underlying cause of robust overfitting through a dynamic minimax game perspective of adversarial training. Based on this perspective, we extend the robust feature framework to analyze the training dynamics of AT, and analyze how the change of LR decay induces an imbalance between the two players of AT. Based on this analysis, we develop a new understanding of robust overfitting through the memorization of non-robust features, and verify this perspective through extensive experiments. We also explore three new approaches to restore the balance between the model trainer and the attacker. Empirically, the proposed ReBalanced Adversarial Training (ReBAT) shows that adversarial training can also benefit from long training as long as the robust overfitting issue is resolved.

References

[1] Sravanti Addepalli, Samyak Jain, Gaurang Sriramanan, and R Venkatesh Babu. Scaling adversarial training to large perturbation bounds. In *ECCV*, 2022.

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

[3] Qi-Zhi Cai, Min Du, Chang Liu, and Dawn Song. Curriculum adversarial training. In *IJCAI*, 2018.

[4] Tianlong Chen, Zhenyu Zhang, Sijia Liu, Shiyu Chang, and Zhangyang Wang. Robust overfitting may be mitigated by properly learned smoothening. In *ICLR*, 2021.

[5] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[7] Chengyu Dong, Liyuan Liu, and Jingbo Shang. Double descent in adversarial training: An implicit label noise perspective. *arXiv preprint arXiv:2110.03135*, 2021.

[8] Yinpeng Dong, Ke Xu, Xiao Yang, Tianyu Pang, Zhijie Deng, Hang Su, and Jun Zhu. Exploring memorization in adversarial training. In *ICLR*, 2022.

[9] Liam H Fowl, Micah Goldblum, Ping-yeh Chiang, Jonas Geiping, Wojciech Czaja, and Tom Goldstein. Adversarial examples make strong poisons. In *NeurIPS*, 2021.

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

[11] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. In *NeurIPS*, 2020.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, 2016.

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[14] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *NeurIPS*, 2017.

[15] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *NeurIPS*, 2019.

[16] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

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

[18] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *NeurIPS*, 2018.

[19] Lin Li and Michael Spratling. Data augmentation alone can improve adversarial training. In *ICLR*, 2023.

[20] Lin Li and Michael Spratling. Understanding and combating robust overfitting via input loss landscape analysis and regularization. *Pattern Recognition*, 2023.

[21] Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards explaining the regularization effect of initial large learning rate in training neural networks. In *NeurIPS*, 2019.

[22] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

[23] Tianyu Pang, Xiao Yang, Yinpeng Dong, Hang Su, and Jun Zhu. Bag of tricks for adversarial training. In *ICLR*, 2020.

[24] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. In *NeurIPS*, 2019.

[25] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan Andrei Calian, Florian Stimberg, Olivia Wiles, and Timothy A Mann. Data augmentation can improve robustness. In *NeurIPS*, 2021.

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

[27] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.

[28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[29] David Stutz, Matthias Hein, and Bernt Schiele. Relating adversarially robust generalization to flat minima. In *ICCV*, 2021.

[30] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

[31] Hongjun Wang and Yisen Wang. Self-ensemble adversarial training for improved robustness. In *ICLR*, 2022.

[32] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *ICML*, 2019.

[33] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *NeurIPS*, 2020.

[34] Chaojian Yu, Bo Han, Li Shen, Jun Yu, Chen Gong, Mingming Gong, and Tongliang Liu. Understanding robust overfitting of adversarial training and beyond. In *ICML*, 2022.

[35] 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 *ICCV*, 2019.

[36] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.

[37] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *ICML*, 2019.