Noise Augmentation Is All You Need For FGSM Fast Adversarial Training: Catastrophic Overfitting And Robust Overfitting Require Different Augmentation

Chaoning Zhang [*1] **Kang Zhang** [*1] **Axi Niu** [2] **Chenshuang Zhang** [1] **Jiu Feng** [3] **Chang D. Yoo** [1] **In So Kweon** [1]

Abstract

Adversarial training (AT) and its variants are the most effective approaches for obtaining adversarially robust models. A unique characteristic of AT is that an inner maximization problem needs to be solved repeatedly before the model weights can be updated, which makes the training slow. FGSM AT significantly improves its efficiency but it fails when the step size grows. The SOTA GradAlign makes FGSM AT compatible with a higher step size, however, its regularization on *input gradient* makes it 3 to 4 times slower than FGSM AT. Our proposed NoiseAug removes the extra computation overhead by directly regularizing on the *input itself*. The key contribution of this work lies in an empirical finding that single-step FGSM AT is not as hard as suggested in the past line of work: **noise augmentation is all you need for (FGSM) fast AT**. Towards understanding the success of our NoiseAug, we perform an extensive analysis and find that mitigating Catastrophic Overfitting (CO) and Robust Overfitting (RO) need different augmentations. Instead of more samples caused by data augmentation, we identify what makes NoiseAug effective for preventing CO might lie in its improved local linearity.

1. Introduction

Deep neural networks are often vulnerable to adversarial examples (AEs) where the quasi-invisible adversarial perturbation causes misclassification (Szegedy et al., 2013). Early attempts to improve adversarial robustness include various image processing techniques and detection techniques, however, most of them are found to give a false sense of robustness (Carlini & Wagner, 2017; Athalye et al., 2018; Croce & Hein, 2020). In recent years, there is an emerging consensus that adversarial training (AT) and its variants are the most effective approaches that are robust to various strong white-box attacks, such as PGD attack (Madry et al., 2018) and AutoAttack (Croce & Hein, 2020). However, AT is resource-intensive because generating adversarial examples with the multi-step gradient ascent before training the network is a heavy computation overhead (Madry et al., 2018).

To speed up AT, successful attempts range from multi-step "free" approaches (Shafahi et al., 2019; Zhang et al., 2019) to single-step FGSM AT (Wong et al., 2020; Andriushchenko & Flammarion, 2020). FGSM (Goodfellow et al., 2015) is not a new technique and has been used to improve adversarial robustness in its early development of adversarial attack and defense. It was previously shown in (Tramèr et al., 2018) to be not effective against a multi-step PGD attack. With an appropriate initialization, however, Wong et al. have shown that FGSM AT achieves reasonable robustness against PGD attack. Notably, it still suffers from Catastrophic Overfitting (CO) when the step size grows.

To address the above limitations of FGSM AT, various attempts have been suggested in the literature. For example, a straightforward solution to alleviate CO is to use early stop Wong et al., however, this makes the training procedure more complex and insufficient training might also lead to a sub-optimal robust model. To this end, multiple works (Vivek & Babu, 2020; Andriushchenko & Flammarion, 2020; Chen et al., 2021) have attempted various regularization methods for improving the robustness without the practice of early stop. However, these regularization methods often cause extra computation overhead. Our work is mainly inspired by (Andriushchenko & Flammarion, 2020), where GradAlign has been proposed to regularize the gradient similarity between a clean example and a noisy example. GradAlign helps increase the tolerance of FGSM AT with a larger step size, such as $\epsilon = 16/255$. This work is motivated to search a more simple variant of regularization method that causes less or no extra computation overhead.

[*]Equal contribution [1]Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea [2]Northwestern Polytechnical University, Shaanxi, China [3]Sichuan University, Sichuan, China. Correspondence to: Chaoning Zhang <chaoningzhang1990@gmail.com>.

In this work, we argue that single-step FGSM AT is not as had hard as suggested by the past line of work (Vivek & Babu, 2020; Andriushchenko & Flammarion, 2020; Kim et al., 2020; Chen et al., 2021). The result of our investigation conveys to the community a simple yet non-trivial message: *noise augmentation is all you need for fast adversarial training.* This empirical finding motivates us to improve FGSM with our proposed NoiseAug which causes zero computation overhead yet outperform other SOTA regularization methods. Towards understanding why NoiseAug helps improve FGSM AT, we investigate it from various perspectives, such as data augmentation and local linearity. Since adding noise to input is one of the most widely used data augmentation techniques, it might be tempting to guess that increased training samples due to data augmentation effect help avoid CO. However, this is not supported by our empirical finding: other types of augmentation that avoid RO via increasing training samples do not help alleviate CO. Moreover, Through the lens of data augmentation, our study on CO and RO reveals that they require different types of augmentation: mitigating RO requires Content-type augmentation, such as MixUp or CutOut, while CO needs Noise-type augmentation, such as uniform noise or Gaussian noise. Instead, we find that local linearity constitutes a better perspective to explain the effectiveness NoiseAug for mitigating CO. Overall, the contributions of our work are summarized as follows:

- The key contribution of this work lies in an empirical finding that CO in single-step FGSM AT can be avoided by adding noise to images. This finding motivates a simple yet effective regularization method for improving FGSM AT. Without any conditional intervention like early stop or additional computation overhead like GradAlign, our NoiseAug avoids CO and outperforms existing SOTA methods.
- We perform an extensive analysis to identify the reason why our NoiseAug helps FGSM AT to avoid CO. Our results show that CO can *only* be mitigated by *Noise-type* augmentation, which contradicts a tempting guess that increasing training samples helps avoid CO. Moreover, we show that among all investigated augmentations, only NoiseAug improves local linearity, which suggests that what makes NoiseAug effective for preventing CO lies in its improved local linearity. Finally, we demonstrate that adding images to the input can improve local linearity is a general phenomenon, regardless of the specific training setup.

2. Related work

It has been an active topic in the adversarial machine learning community to make AT more computation efficient. Here, we summarize the main progress in the past few years.

"Free" to FGSM AT. To make AT more computation efficient, there are two lines of works. Early attempts (Shafahi et al., 2019; Zhang et al., 2019) investigated the possibility of "free" adversarial training to achieve robustness with similar computation overhead as standard training. Another line of work attempt to minimize the number of steps to generate the adversarial examples. A major advantage of the second line of approaches is that it has extremely few parameters to tune, which makes it easily compatible with most training procedures. For example, it can be drastically accelerated (Wong et al., 2020) by using standard techniques for boost training, such as cyclic learning rates (Smith & Topin, 2018) and mixed-precision training (Micikevicius et al., 2017). In practice, "Free" AT is not really free because their minibatch replays often require much more training steps even though it is faster than FGSM-AT for a single training step. Recently, the trend has shifted from multi-step "free" AT to single-step FGSM AT.

CO and countermeasures. Since the first success of FGSM+RS (Wong et al., 2020) to show reasonable robustness against PGD-50-10, multiple works have attempted to improve FGSM AT with various techniques. (Li et al., 2020) claims that the success of FGSM-RS lies in improved success factor to recover from CO and proposed to use PGD when the CO is detected. (Vivek & Babu, 2020) has claimed that the CO is caused by model parameter overfitting in the early stage, which motivates their dynamic dropout scheduling. (Vivek & Babu, 2020) has proposed to regularize the FGSM AT by introducing dropout layer after each non-linear layer. Specifically, those dropout layers are initialized with a high dropout probability which is linearly decayed during the training. (Kim et al., 2020) assumed that the overfitting is caused by a fixed perturbation magnitude and thus proposed to search a sample-wise minimum perturbation to avoid CO. Specifically, they set c checkpoints in the local region to find the smallest perturbation that causes misclassification. Motivated from an observation that CO often occurs when the local linearity of the model is low, (Andriushchenko & Flammarion, 2020) has introduced a regularization loss (GradAlign) to explicitly maximize local linearity for avoiding CO. More recently, (Chen et al., 2021) claims that the plus-minus sign of the higher-order terms is the underlying cause of CO and proposes a regularization loss (FLAT) to make the model more locally linear.

3. Background

Standard *vs*. Adversarial Training. Let's assume $\mathcal{D}$ is a data distribution with (x, y) pairs and $f(\cdot, \theta)$ is a model parametrized by θ. Standard training (ST) minimizes the risk of $\mathbb{E}_{(x,y)\sim\mathcal{D}}[l(f(x, \theta), y)]$, where l indicates the cross-entropy loss. By contrast, adversarial training (AT) finds

model parameter θ to optimize an adversarial risk:

$$\underbrace{\arg\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}} \underbrace{\left[\max_{\delta\in\mathbb{S}} l(f(x+\delta;\theta),y)\right]}_{\text{inner maximization}}}_{\text{outer minimization}}, \quad (1)$$

where $\mathbb{S}$ denotes the allowed perturbation budget that is a typically l_p norm-bounded ϵ. A key difference between ST and AT is that AT generates adversarial examples as an inner maximization problem before optimizing the model weights. Following the convention in (Andriushchenko & Flammarion, 2020), we mainly study AT under the constraint of $\epsilon = 8/255$ or $\epsilon = 16/255$.

Algorithm 1 Classical PGD-N AT for a network f_θ with T epochs, given some radius ϵ, adversarial step size α and N PGD steps and a dataset of size M

for $t = 1 \dots T$ **do**
  for $i = 1 \dots M$ **do**
    $\delta = 0$ or $\delta = \text{Uniform}(-\epsilon, \epsilon)$ // δ *initialization*
    for $j = 1 \dots N$ **do**
      $\delta = \delta + \alpha \cdot \text{sign}(\nabla_\delta \ell(f_\theta(x_i + \delta), y_i))$
      $\delta = \max(\min(\delta, \epsilon), -\epsilon)$
    end for
    $\theta = \theta - \nabla_\theta \ell(f_\theta(x_i + \delta), y_i)$ // *Outer minimization*
  end for
end for

PGD *vs*. FGSM AT. As discussed above, a unique nature of AT lies in solving an inner maximization problem. Projected gradient decent (PGD) is the most widely used approach for solving this problem. A critical hyperparameter in PGD-AT is the number of steps for generating adversarial examples. Following the convention in prior works (Wong et al., 2020; Andriushchenko & Flammarion, 2020), we term it PGD-N when N steps are used. The classical PGD-N AT is shown in Algorithm 1. FGSM can be seen as a special case of PGD-N when N is set to 1 and we stick to term it FGSM AT. A major advantage of applying FGSM is that it makes the slow inner maximization more computation efficient. In Algorithm 1, the to-be-optimized perturbation δ in the inner-maximization by default starts with zero, but can be optionally initialized in a random manner. Since the advent of FGSM, it has been widely used for training adversarially robust models (Goodfellow et al., 2015; Tramèr et al., 2018). For example, (Goodfellow et al., 2015) shows that FGSM AT is an effective way to train a model robust to their proposed FGSM attack. **Overfitting in AT.** Since the model is only trained on adversarial examples generated by FGSM, it has a risk of being overfitted to FGSM while losing its robustness against stronger attacks (PGD attack for instance). In practical FGSM AT, its robustness against PGD attack might suddenly drop to zero, which is identified as a failure mode termed as *CO*. On the other hand, (Rice et al., 2020) has reported that the robustness on the evaluation dataset decreases in the later stage of training, and termed this phenomenon *RO* (RO). CO and RO are two intriguing phenomena in AT, yet there is no consensus on the underlying reason.

4. How to avoid CO in FGSM fast AT?

4.1. Prior attempts

First yet the limited success of FGSM+RS. FGSM AT was long dismissed as ineffective against PGD attack, a seminal work (Wong et al., 2020) has shown that this is not necessarily true. Their key finding is that a proper initialization of perturbation can help FGSM AT avoid CO and achieve reasonable robustness against a PGD attack. FGSM with such initialized perturbation in the random step (RS) is termed FGSM-RS following (Andriushchenko & Flammarion, 2020). Even though FGSM+RS shows success for $\epsilon = 8/255$, CO still occurs when the step size grows. For example, (Wong et al., 2020) has reported that CO still occurs when the step size α is larger than $11/255$, which hinders its use for a larger ϵ (ℓ_∞ $16/255$ for instance). Furthermore, (Wong et al., 2020) has also alleviated CO by early stop which is also a common practice to avoid overfitting in standard training. However, inefficient training often leads to a sub-optimal robust model as suggested in (Andriushchenko & Flammarion, 2020).

Gradient-based regularization. Most attempts (Vivek & Babu, 2020; Kim et al., 2020; Chen et al., 2021) for improving FGSM AT are performed by limiting ϵ to $8/255$, which renders FGSM AT for a large ϵ like $16/255$ remain a challenging problem. Without the practice of early stop, (Andriushchenko & Flammarion, 2020) is the first yet so far the only one found to report success of FGSM AT against PGD attack with $\epsilon = 16/255$. The core of their method is a regularization loss GradAlign which is termed as such due to its goal for increasing the gradient alignment, *i.e*. a metric to measure local linearity of model. Despite its success, as acknowledged in (Andriushchenko & Flammarion, 2020), a major drawback of GradAlign is that the regularization on the input gradient yields heavy computation overhead compared with FGSM AT, which makes it 3 to 4 times slower than FGSM AT. Beyond GradAlign, their work also investigates other gradient-based penalties, such as curvature regularization (Demontis et al., 2019) and l_2-based gradient norm regularization, both of which consistently perform less well than GradAlign. For future directions, (Andriushchenko & Flammarion, 2020) has suggested speeding up GradAlign with approaches like parallelization or coming up with other regularization methods. In the following, we report the findings of our investigation to improve GradAlign.

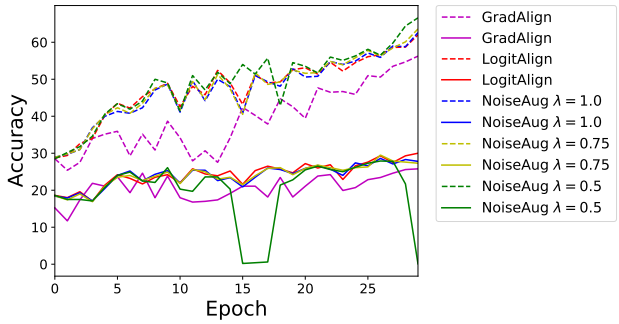

Figure 1. Comparison among GradAlign, LogitAlign with KL regularization, and NoiseAug with various λ. Dash line indicates clean accuracy and solid line indicates accuracy under the attack of PGD-50-10.

4.2. Our findings

GradAlign to LogitAlign. This work focuses on exploring alternative regularization methods that require less or zero computation overhead. In essence, GradAlign encourages the model to behave similarly to clean example (x) and corresponding noisy example ($x+\eta$) for input gradient. Since the input gradient can be used for generating adversarial examples with FGSM, we conjecture that increasing the similarity of the output logit for adversarial examples generated from clean and corresponding noisy samples might achieve an equivalent effect. Specifically the total loss with a KL regularization is shown as:

$$l(f(x+\delta_1;\theta),y)+KL(f(x+\delta_1;\theta),f(x+\eta+\delta_2;\theta)), \quad (2)$$

where δ_1 and δ_2 are the adversarial perturbations for clean and noisy examples, respectively. Since the regularization is performed on the output logit, we term it LogitAlign to differentiate from GradAlign. An advantage of LogitAlign is that it avoids double backpropagation and thus facilitates the computation parallelization by simply concatenating the inputs in the PyTorch implementation. The results in Figure 1 show that our proposed LogitAlign achieves performance superior to GradAlign in terms of both robustness performance and required computation.

Is alignment necessary? As shown above, the alignment on both input gradient and output logit help avoid CO. However, it remains unclear whether such alignment is really necessary. Following LogitAlign, we use adversarial examples from both clean examples and noisy examples in the outer minimization to update the model weights. By contrast, we do not use KL divergence to increase their similarity but optimize them directly with a combined CE loss as:

$$(1-\lambda)l(f(x+\delta_1;\theta),y)+\lambda l(f(x+\eta+\delta_2;\theta),y). \quad (3)$$

where $\lambda \in [0,1]$. As shown in Figure 1, we find that the increase of λ helps alleviate CO and setting λ to 1 achieves comparable performance as LogitAlign. Somewhat surprisingly, our empirical results suggest that the alignment is not absolutely necessary and simply augmenting images with noise might be sufficient for avoiding CO. Due to its simplicity as well as competitive performance, we adopt it as our final regularization method and term it *NoiseAug*.

Algorithm 2 NoiseAug-based PGD-N AT for a network f_θ with T epochs, given some radius ϵ, adversarial step size α and N PGD steps and a dataset of size M

for $t = 1 \dots T$ **do**
  for $i = 1 \dots M$ **do**
    $x_i \longleftarrow x_i + \eta$ // *Noise augmentation*
    $\delta = 0$
    for $j = 1 \dots N$ **do**
      $\delta = \delta + \alpha \cdot \text{sign}(\nabla_\delta \ell(f_\theta(x_i+\delta), y_i))$
      $\delta = \max(\min(\delta, \epsilon), -\epsilon)$
    end for
    $\theta = \theta - \nabla_\theta \ell(f_\theta(x_i+\delta), y_i)$ // *Outer minimization*
  end for
end for

4.3. Our proposed method

FGSM + NoiseAug. Compared with GradAlign and LogitAlign, a major advantage of NoiseAug is that it causes zero [1] additional overhead. Following the practice in (Andriushchenko & Flammarion, 2020), our proposed NoiseAug by default initializes the perturbation as zero, rendering our single-step AT method as FGSM + NoiseAug. Without any conditional intervention like early stop, our single-step AT alleviates CO by combining the *very first* FGSM AT method with a common noise augmentation. Our proposed NoiseAug is also easily compatible with the multiple-step PGD (see results in Table 2. Implementation-wise, it is no more than a single line of code as shown in Algorithm 2 based on the classical PGD-N AT pseudo code. In the following, we mainly focus on the special case PGD-1, *i.e.* FGSM, by discussing its relation with two seminal single-step approaches with FGSM + GradAlign (Andriushchenko & Flammarion, 2020) and FGSM-RS (Wong et al., 2020).

Relation with prior methods. (a) FGSM+RS and

[1] Technically, augmenting images with the noise still consumes computation, however, it is negligible compared with the time to generate adversarial examples and training the model.

	Method	Standard	PGD-20-10	PGD-50-10	Time
	Standard	94.04±0.19	0.0±0.0	0.0±0.0	1
Multi-step	YOPO-3-5	81.20	36.30	N.A.	N.A.
	YOPO-5-3	83.99	44.72	N.A.	N.A.
	AT for free ($m = 8$)	77.92 ±0.65	45.90±0.98	N.A.	N.A.
	PGD-10	81.88 ±0.37	N.A.	50.04 ±0.79	~ 11
	PGD-10 +NoiseAug	79.39 ±0.32	N.A.	**50.53** ±0.17	~ 11
Single-step	FGSM	85.16 ±1.30	N.A.	0.02 ±0.04	~ 2
	FGSM-RS	84.32 ±0.08	N.A.	45.10 ±0.56	~ 2
	FGSM+FLAT	82.27±0.15	47.81±0.17	N.A.	~ 3.6
	FGSM+Checkpoint	89.1±0.0 ±0.27	N.A.	37.8±0.5	~ 4
	FGSM+GradAlign	81.28±0.27	N.A.	46.90±0.57	~ 8
	FGSM+NoiseAug($\mathcal{U}$)	81.04±0.39	N.A.	48.26±0.29	~ 2
	FGSM+NoiseAug($\mathcal{N}$)	80.19±0.10	N.A.	**48.43**±0.15	~ 2

Table 1. Standard accuracy and robustness (%) under the attack of PGD-50-10 for $\epsilon = 8/255$. For the required training time, we report the value that is relative to the standard training.

FGSM+NoiseAug resemble each other in the sense that both adopt random noise for improving FGSM AT. NoiseAug can be interpreted as separating the δ in FGSM-RS into a disentangled random noise and FGSM perturbation. This disentangling might look trivial but yield a significant difference. First, due to constraint on perturbation magnitude, the initialized random noise in FGSM-RS can not be set to be larger than $\mathcal{U}(-\epsilon, \epsilon)$. Our results in Table 4 suggest that an appropriate noise magnitude is critical for achieving high robustness. Disentangling random noise from the perturbation, NoiseAug allows to freely choose the optimal noise type or magnitude by just treating it as data augmentation. Second, random initialization can yield inferior performance (See Table 5). It is worth mentioning that (Andriushchenko & Flammarion, 2020) also shows that removing random initialization results in a stronger PGD-2 baseline. (b) This motivates FGSM+GradAlign and our FGSM+NoiseAug to not use the random initialization as in FGSM-RS. Our work differs from (Andriushchenko & Flammarion, 2020) mainly by replacing their GradAlign with our proposed NoiseAug. Conceptually, GradAlign regularizes *input gradient* while NoiseAug regularizes *input itself*.

5. Experimental setup and results

Experimental setup. Unless mentioned otherwise, we follow (Wong et al., 2020; Andriushchenko & Flammarion, 2020) to train PreAct ResNet18 (He et al., 2016) for 30 epochs with the cyclic learning rates (Smith, 2017) and half-precision training (Micikevicius et al., 2017). The maximal learning rate in the cyclic schedule is 0.3. For the perturbation budget, we adopt ℓ_∞-norm constraint and set ϵ to $8/255$ or $16/255$. Following prior works, we evaluate the adversarial robustness with the attack of PGD-50-10, *i.e.* 50 iterations and 10 restarts, where the step size is set to $\alpha = \epsilon/4$. For the step size in the training, we follow the setup in (Andriushchenko & Flammarion, 2020) by setting α to 1.25ϵ and $\epsilon/2$ for single-step (FGSM) and twp-step (PGD-2) AT, respectively. Note that the training is performed with half-precision for speeding up but the evaluation is always conducted with single-precision for fair comparison because limited numerical precision in the gradient calculation might overestimate the model robustness.

5.1. Main results

Basic results. Table 1 reports the results for both multi-step and single-step setups. Specifically, multiple setup includes PGD-10 (Madry et al., 2018), YOPO (Zhang et al., 2019), AT for free (Shafahi et al., 2019); single-step setup includes FGSM (Goodfellow et al., 2015), FGSM-RS (Wong et al., 2020), FGSM+GradAlign (Andriushchenko & Flammarion, 2020), FGSM+Checkpoint (Kim et al., 2020), FGSM+FLAT (Chen et al., 2021). All the results are either retrieved from the original works or reproduced with their official code. Since (Andriushchenko & Flammarion, 2020) only reports results on a subset of the evaluation dataset, we reproduce it on the full evaluation dataset. The parameter λ is set to 0.2 and 2 for $\epsilon = 8/255$ and $\epsilon = 16/255$, respectively, as their paper suggested. Among all existing single-step AT methods, our simple FGSM+NoiseAug performs the best, bridging its gap with PGD-10. We highlight that Checkpoint, GradAlign and FLAT all induce extra computation overhead to different degrees over the baseline FGSM AT, while our NoiseAug is overhead-free. FGSM-RS is also overhead-free but it yields less satisfactory performance. Moreover, it can fail in the following more challenging setup ($\epsilon = 16/255$).

Method	Standard	PGD-50-10
Standard	94.04±0.19	0.0±0.0.
PGD-10 ($\alpha = 2\epsilon/10$)	60.28 ±0.50	33.24 ±0.52
PGD-10+GradAlign	60.70 ±0.89	31.28±0.42
PGD-10+NoiseAug	60.62 ±0.41	29.97±0.51
FGSM	73.76 ± 7.40	0.0±0.0
FGSM+GradAlign	58.46 ± 0.22	26.88±1.81
FGSM+NoiseAug	62.51±0.24	**28.16**±1.01
PGD-2	68.65 ± 5.83	20.50±6.54
PGD-2 + GradAlign (Original)	61.38 ± 0.71	29.80± 0.42
PGD-2 + GradAlign (Reproduce)	63.21 ± 0.82	28.96± 0.40
PGD-2 + NoiseAug	61.55 ± 0.17	29.71± 0.34

Table 2. Standard accuracy and robustness (%) under the attack of PGD-50-10 for $\epsilon = 16/255$.

Method	AA
FGSM	0.0±0.0.
FGSM + GradAlign	20.56+0.36
FGSM + NoiseAug	21.91+0.32

Table 3. Robustness under AutoAttack with $\epsilon = 16/255$..

More challenging setup. The main challenge of FGSM AT is that it suffers from CO when the step size increases, *otherwise the basic FGSM AT can be sufficient*. Thus, it is critical for the proposed methods to also work for a relatively large ϵ. Most of the above methods only report the results for $\epsilon = 8/255$, while (Andriushchenko & Flammarion, 2020) is the only one that also reports success for $\epsilon = 16/255$. We argue that it is important for the proposed methods to also work for this challenging setup. Table 2 reports the comparison of our NoiseAug against GradAlign with various steps. The basic FGSM or PGD-2 suffers from CO, leading to poor performance. Both GradAlign and NoiseAug are found to be effective for alleviating CO, while NoiseAug consistently outperforms GradAlign in both single-step and two-step scenarios. PGD-10 performs the best but at the cost of much more computation resources. Neither GradAlign nor NoiseAug can further improve PGD-10, which is expected.

Robustness under AA. To further ensure that FGSM+NoiseAg do not benefit from gradient masking, following (Andriushchenko & Flammarion, 2020), Table 3 report the robustness under AutoAttack (AA). We observe that FGSM+NoiseAug achieves superior robustness against AA.

5.2. Ablation study

Noise type and scale. Table 4 reports the influence of noise type and scale. For uniform noise, the basic magnitude is set to $\mathcal{U}(-\epsilon, \epsilon)$ and $\epsilon \times \mathcal{N}(0, 1)$ for Gaussian noise and uniform noise, respectively. Both of them are multiplied by a scale factor (s). We investigate the scale in the range from 0 to 3. The results show that both noise types significantly improve the robustness. Moreover, the standard accuracy decreases when the noise magnitude is set to too large.

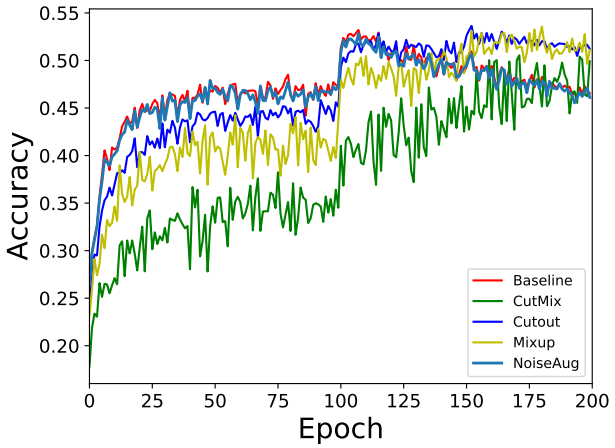

Figure 2. Robustness under the attack of PGD-10. CutMix, Coutout, and Mixup help alleviate RO, while NoiseAug has little benefit for avoiding RO.

6. Why does NoiseAug improve FGSM AT?

6.1. Preliminary guess

As shown above, NoiseAug outperforms other regularization methods for improving FGSM AT without suffering from CO. It is natural to ask why such a simple regularization is so effective to avoid CO. Overfitting is a general concern in machine learning and data augmentation like adding noise to the input (Bishop et al., 1995; Reed & MarksII, 1999; Goodfellow et al., 2016; Vincent et al., 2010; Brownlee, 2019) is the most widely used technique to avoid such concerns for training DNNs. Thus, a tempting guess on why NoiseAug improves FGSM AT is as follows:

The benefit comes from the effect of data augmentation, i.e. *more training samples*. Here, we roughly divide the common data augmentation into two types based on their characteristic. The Noise-type augmentation does not change the image content itself except adding some additional noise, such as Gaussian noise or Uniform noise. The other type of augmentation techniques does not involve noise but introduces content change, such as Mixup (Zhang et al., 2017), Cutout (DeVries & Taylor, 2017), CutMix (Yun et al., 2019). To facilitate discussion, non-noise augmentation is termed Content-type.

Does NoiseAug alleviate RO? Figure 2 reports the effect of both types of augmentations on RO. CutMix, Cutout and Mixup mitigate the RO, which aligns the finding in (Rebuffi

Type	Scale	$\epsilon = 8/255$		$\epsilon = 16/255$	
		Standard	PGD-50-10	Standard	PGD-50-10
$\mathcal{U}$	$0 \times \epsilon$	85.16±1.30	0.02±0.04	73.76±7.4	0.00±0.00
$\mathcal{U}$	$1 \times \epsilon$	**81.04**±0.39	48.26±0.29	66.88±4.65	20.85±11.66
$\mathcal{U}$	$2 \times \epsilon$	80.23±0.50	48.41±0.14	62.51±0.24	**28.16**±1.01
$\mathcal{U}$	$3 \times \epsilon$	79.15±0.37	48.22±0.29	60.13±0.29	27.56±0.22
$\mathcal{N}$	$0 \times \epsilon$	85.16±1.30	0.02±0.04	73.76±7.4	0.00±0.00
$\mathcal{N}$	$1 \times \epsilon$	80.19±0.10	**48.43±0.15**	62.26±1.05	27.57±0.99
$\mathcal{N}$	$2 \times \epsilon$	77.89±1.15	48.07±0.37	58.95±0.41	**28.31±0.94**
$\mathcal{N}$	$3 \times \epsilon$	76.40±0.11	47.39±0.17	55.43±0.10	27.54±0.30

Table 4. Ablation study on noise type and magnitude.

	δ init	Standard	PGD-50-10
$\epsilon = 8/255.$	zero	80.19±0.10	48.43±0.15
	random	82.38±0.02	46.36±0.02
$\epsilon = 16/255.$	zero	58.96±0.10	28.31±0.17
	random	66.25±0.14	25.84±0.04

Table 5. Standard accuracy and robustness under the attack of PGD-50-10. The results are reported to check whether random initialization helps improve performance. Random initialization leads to a higher standard accuracy but at the cost of a lower robustness.

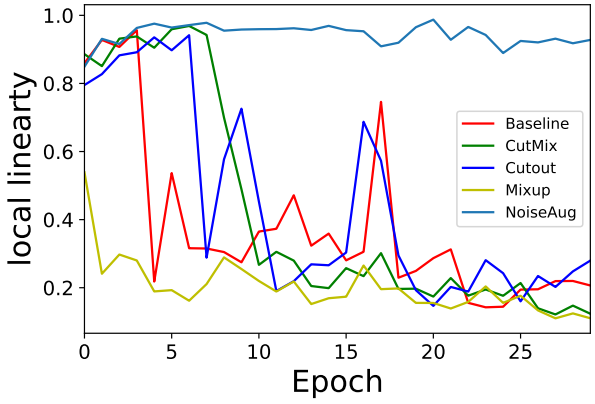

Figure 3. Local linearity with FGSM+AT. Among the investigated four types of Augmentations, only NoiseAug helps increase local linearity, while CutMix, Cutout, Mixup have no effect in increasing local linearity.

et al., 2021). Moreover, unlabeled data is also found in (Carmon et al., 2019; Gowal et al., 2021) to alleviate RO for improving robustness. The findings support that RO can be mitigated by creating more training samples. Interestingly, we find that Noise-type brings little benefit to mitigate RO, suggesting Noise-type might not be as strong as other Content-type augmentation for generating more training samples. This is somewhat reasonable because the extra samples created by Content-type augmentation might be more diverse than just adding noise. It is worth mentioning that (Andriushchenko & Flammarion, 2020) also reports that, like our NoiseAug, their GradAlign does not prevent RO. This suggests that RO and CO might be less related as their shared term "overfitting" could imply.

Does Content-type augmentation alleviate RO? The results in Table 6 show that none of the investigated Content-type augmentation techniques helps alleviate CO, which suggests that our preliminary guess is unlikely to be true. The benefit of NoiseAug to help mitigate CO is not caused by more training samples. Otherwise, Content-type augmentation techniques are also expected to mitigate CO.

6.2. Local linearity

It has been shown in (Moosavi-Dezfooli et al., 2018; Qin et al., 2019) that the model robustness can be improved by regularization methods for increasing the local linearity of a model. Despite various motivations, we find that there is a consensus to explain the CO from the local linearity perspective (Kim et al., 2020; Andriushchenko & Flammarion, 2020; Chen et al., 2021). This linearity perspective well explains why CO only occurs in single-step FGSM with a large step size but is not reported in multi-step PGD. Straightforwardly, CO occurs because FGSM fails to solve the inner maximization problem which does not allow a larger step size when the model is locally non-linear (the term *local* is characterized by the fact that perturbation is often limited to a small value) (Andriushchenko & Flammarion, 2020). (Benz et al., 2020) has quantified such linearity as local input gradient similarity (LIGS): $\mathbb{E}_{(x,y)\sim D}[\cos(\nabla_x \ell(x, y; \theta), \nabla_x \ell(x + \eta, y; \theta))]$, where η is randomly sampled from a certain distribution. The metric is suggested to be interpreted as follows: it is close to one when the model is locally linear and its value decreases to

zero when the model is very non-linear. (Andriushchenko & Flammarion, 2020) has found that CO often occurs when the above gradient alignment (similarity) is small, establishing a correlation between them.

Adopting the same metric in (Benz et al., 2020; Andriushchenko & Flammarion, 2020), we investigate whether augmenting images with noise improves local linearity. The results in Figure 3 show that NoiseAug significantly improves the local linearity. We argue that improved local linearity might be the reason that our proposed NoiseAugment. To further corroborate this claim, we also test with other Content-type data augmentations and find that they do not improve the local linearity. Note that GradAlign also improves FGSM AT through increasing local linearity. On the one hand, this result suggests that local linearity can be increased by directly regularizing on the input itself, which is overhead-free and renders the regularizing on input gradient like GradAlign unnecessary. On the other hand, it also challenges a claim in (Andriushchenko & Flammarion, 2020) regarding why random initialization in FGSM-RS helps mitigate CO. Specifically, (Andriushchenko & Flammarion, 2020) claims that it "*boils down to reducing the average magnitude of the perturbations*". Note that random noise in NoiseAug does not decrease the perturbation magnitude but it still helps avoid CO.

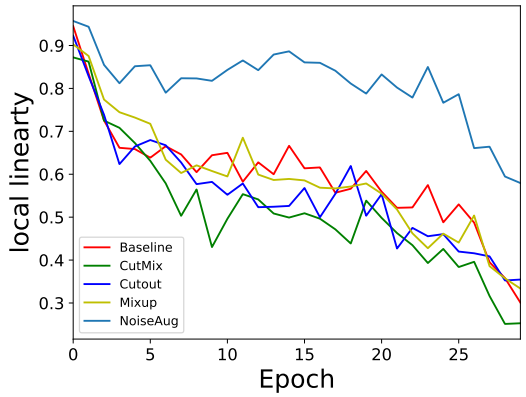

Figure 4. Local linearity with standard training. Among the investigated four types of Augmentations, only NoiseAug helps increase local linearity, while CutMix, Cutout, Mixup have no effect in increasing local linearity.

6.3. Beyond FGSM+NoiseAug

Standard training. We further investigate the influence of NoiseAug in standard training and the results are reported in Figure 4. The local linearity decreases when the training epoch increases, which aligns with the finding in (Benz et al., 2020; Andriushchenko & Flammarion, 2020). Compared with the baseline, NoiseAug improves local linearity in the whole training process. Before the advent of adversarial attack and defense, adding noise to images has been a well-established practice in standard training for avoiding overfitting to the training dataset. In other words, it has been mainly perceived as one of the many data augmentation techniques. We advocate that future researchers who study noise augmentation should also be aware of its effect to improve local linearity.

Method	Standard	PGD-50-10
Baseline	15.5±0.91	6.82±9.64
Cutout	15.41±0.43	0.085±0.12
Mixup	18.81±1.56	0±0
CutMix	15.46±0.91	0±0
NoiseAug	62.51±0.24	28.16±1.01

Table 6. Standard accuracy and robustness under the attack of PGD-50-10. Results with various augmentations.

7. Closing remark

This work has studied how to improve FGSM AT, especially in terms of preventing CO. With the motivation to avoid double backpropagation in GradAlign, our investigation shows that LogitAlign achieves comparable performance. More interestingly, we find that simply augmenting images with noise achieves the best performance. Despite the simplicity, our proposed NoiseAugment outperforms existing regularization methods by a visible margin yet causes zero computation overhead. We investigate why NoiseAugment improves FGSM AT. Specifically, we have performed a comprehensive study on CO and RO through the lens of data augmentation and found that they need different augmentations. We have shown that in both AT and standard training, only Noise-type augmentation improves local linearity of model and thus improve FGSM AT.

Discussion. Overall, we find that the literature has no consensus on why CO occurs. Most identified reasons have been motivated to justify their proposed method, which is often not compatible with the success of other methods. Nonetheless, there is a growing consensus that CO is related to low local linearity of the model. The observation that Noise-type augmentation increases local linearity and prevents CO while Content-type augmentation has no such effect further corroborates this consensus. Among all the existing regularization methods to increase local linearity, NoiseAugment is the most simple yet effective one. Due to its simplicity, it can be readily extended for more investigation in future works. We highlight that the key contribution of this work lies in conveying a message that noise augmentation is all you need for (FGSM) fast AT.

References

Andriushchenko, M. and Flammarion, N. Understanding and improving fast adversarial training. *NeurIPS*, 2020.

Athalye, A., Carlini, N., and Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Benz, P., Zhang, C., and Kweon, I. S. Batch normalization increases adversarial vulnerability: Disentangling usefulness and robustness of model features. *arXiv preprint arXiv:2010.03316*, 2020.

Bishop, C. M. et al. *Neural networks for pattern recognition*. Oxford university press, 1995.

Brownlee, J. Train neural networks with noise to reduce overfitting. *Machine Learning Mastery*, 2019.

Carlini, N. and Wagner, D. Adversarial examples are not easily detected. In *ACM Workshop on Artificial Intelligence and Security*, 2017.

Carmon, Y., Raghunathan, A., Schmidt, L., Liang, P., and Duchi, J. C. Unlabeled data improves adversarial robustness. *arXiv preprint arXiv:1905.13736*, 2019.

Chen, R., Luo, Y., and Wang, Y. Towards understanding catastrophic overfitting in fast adversarial training. 2021.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

Demontis, A., Melis, M., Pintor, M., Jagielski, M., Biggio, B., Oprea, A., Nita-Rotaru, C., and Roli, F. Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In *USENIX*, 2019.

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

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. The MIT Press, 2016.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Gowal, S., Rebuffi, S.-A., Wiles, O., Stimberg, F., Calian, D. A., and Mann, T. A. Improving robustness using generated data. *NeurIPS*, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *ECCV*, 2016.

Kim, H., Lee, W., and Lee, J. Understanding catastrophic overfitting in single-step adversarial training. *arXiv preprint arXiv:2010.01799*, 2020.

Li, B., Wang, S., Jana, S., and Carin, L. Towards understanding fast adversarial training. *arXiv preprint arXiv:2006.03089*, 2020.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.

Moosavi-Dezfooli, S.-M., Fawzi, A., Fawzi, O., Frossard, P., and Soatto, S. Robustness of classifiers to universal perturbations: A geometric perspective. In *ICLR*, 2018.

Qin, C., Martens, J., Gowal, S., Krishnan, D., Dvijotham, K., Fawzi, A., De, S., Stanforth, R., and Kohli, P. Adversarial robustness through local linearization. In *NeurIPS*, 2019.

Rebuffi, S.-A., Gowal, S., Calian, D. A., Stimberg, F., Wiles, O., and Mann, T. Fixing data augmentation to improve adversarial robustness. *NeruIPS*, 2021.

Reed, R. and MarksII, R. J. *Neural smithing: supervised learning in feedforward artificial neural networks*. 1999.

Rice, L., Wong, E., and Kolter, Z. Overfitting in adversarially robust deep learning. In *ICML*, 2020.

Shafahi, A., Najibi, M., Ghiasi, M. A., Xu, Z., Dickerson, J., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial training for free! In *NeurIPS*, 2019.

Smith, L. N. Cyclical learning rates for training neural networks. In *WACV*, 2017.

Smith, L. N. and Topin, N. Super-convergence: Very fast training of residual networks using large learning rates. 2018.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. *ICLR*, 2018.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A., and Bottou, L. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 2010.

Vivek, B. and Babu, R. V. Single-step adversarial training with dropout scheduling. In *CVPR*, 2020.

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. *ICLR*, 2020.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019.

Zhang, D., Zhang, T., Lu, Y., Zhu, Z., and Dong, B. You only propagate once: Accelerating adversarial training via maximal principle. In *NeurIPS*, 2019.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.