Applied Intelligence (2023) 53:19843–19859
https://doi.org/10.1007/s10489-023-04532-5

Exploring misclassifications of robust neural networks to enhance adversarial attacks

Leo Schwinn[1] · René Raab[1] · An Nguyen[1] · Dario Zanca[1] · Bjoern Eskofier[1]

Accepted: 15 February 2023 / Published online: 21 March 2023

Abstract

Progress in making neural networks more robust against adversarial attacks is mostly marginal, despite the great efforts of the research community. Moreover, the robustness evaluation is often imprecise, making it challenging to identify promising approaches. We do an observational study on the classification decisions of 19 different state-of-the-art neural networks trained to be robust against adversarial attacks. This analysis gives a new indication of the limits of the robustness of current models on a common benchmark. In addition, our findings suggest that current untargeted adversarial attacks induce misclassification toward only a limited amount of different classes. Similarly, we find that previous attacks under-explore the perturbation space during optimization. This leads to unsuccessful attacks for samples where the initial gradient direction is not a good approximation of the final adversarial perturbation direction. Additionally, we observe that both over- and under-confidence in model predictions result in an inaccurate assessment of model robustness. Based on these observations, we propose a novel loss function for adversarial attacks that consistently improves their efficiency and success rate compared to prior attacks for all 30 analyzed models.

Keywords Adversarial attacks · Deep learning · Computer vision · Robustness

1 Introduction

Deep neural networks (DNN) can be easily fooled into making wrong predictions by seemingly negligible perturbations to their input data, called adversarial examples. [1] first demonstrated the existence of adversarial examples for neural networks in the image domain. Since then, adversarial examples have been identified in various other domains such as speech recognition [2, 3] and natural language processing [4, 5]. The prevalence of adversarial examples has severe security implications for real-world applications, making the development of robust machine learning models essential. As a result, the robustness of neural networks has become a central research topic of deep learning in recent years [6].

Several defense strategies have been proposed to make Deep Neural Networks (DNNs) more robust and reliable [7–13]. However, most of them have later been shown to be ineffective against stronger attacks [14, 15] and overall progress has been slow [16]. As robustness improvements are mostly in the single-digit percentage range, a reliable evaluation of new defense strategies is critical to identify methods that actually improve robustness. An inaccurate evaluation of new defenses can lead to the adaption of ineffective defense strategies, which in turn may hinder progress in robustness research. Moreover, accurate quantification of network robustness is necessary to accurately assess the risk of deploying machine learning models in the real world.

To improve the robustness quantification of neural networks, the community has established helpful guidelines

✉ Leo Schwinn
leo.schwinn@fau.de

René Raab
rene.raab@fau.de

An Nguyen
an.nguyen@fau.de

Dario Zanca
dario.zanca@fau.de

Bjoern Eskofier
bjoern.eskofier@fau.de

[1] Department Artificial Intelligence in Biomedical Engineering,
Friedirch-Alexander Universität Erlangen Nürnberg,
Carl-Thiersch-Straße 2b, Erlangen, 91052, Bavaria, Germany

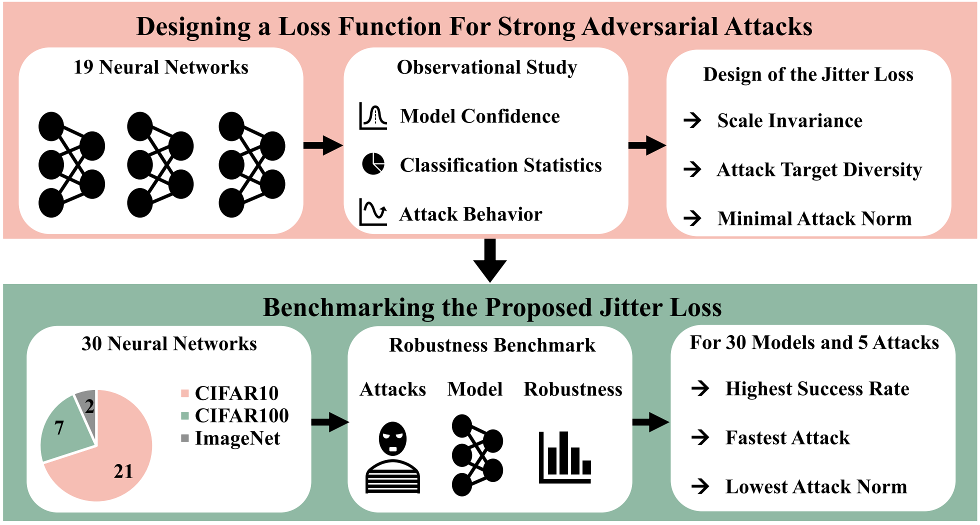


Fig. 1 We investigate failure cases of adversarial attacks for 19 different neural networks trained to be robust against adversarial attacks on the CIFAR10 dataset [20]. Based on the result of this observational study, we propose the Jitter loss function to improve the effectiveness of adversarial attacks. We benchmark Jitter against 5 state-of-the-art (SOTA) adversarial attacks and 30 different neural networks. In all cases, Jitter achieves the highest success rate and efficiency out of all attacks

[17–19]. Nevertheless, the worst-case robustness of DNNs is still reduced repetitively by even stronger attacks and a precise evaluation remains a challenging problem [16].

In this work, we explore the classification decisions of 19 recently published DNNs to identify weak points of current adversarial attacks (Fig. 1). Hereby, we restrict our analysis to DNNs which have been trained to be robust against adversarial attacks with a variety of different methods. Our analysis can be summarized by four main findings:

1. We show that none of the 21 CIFAR10 [20] models that we evaluated can correctly classify 24.2% of the CIFAR10 test data in the presence of adversarial attacks. This finding emphasizes the vast gap in robust and clean performance of current neural networks beyond prior analysis done for individual models.
2. Further, we observe that untargeted adversarial attacks cause misclassification towards only a limited amount of different classes in the dataset.
3. Additionally, we find that, where the loss of the model does not change along the direction of the initial gradient, samples are more difficult to attack. This limits current gradient-based attacks that exploit these gradient directions without sufficiently exploring the loss landscape.
4. Furthermore, we observe that if a model exhibits irregularly large over- and under-confidence, it is difficult to assess its robustness accurately.

We leverage these observations to design a new loss function that improves the success rate of adversarial attacks compared to current state-of-the-art loss functions. More specifically, we encourage target diversity in untargeted

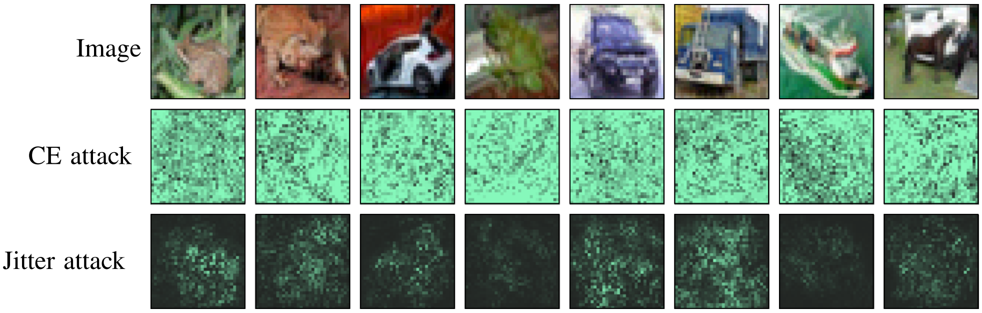


Fig. 2 Difference of adversarial perturbations created by Cross-Entropy (CE)-based attacks and Jitter-based attacks. Original images are shown in the first row, CE-based perturbations in the second row, and Jitter-based perturbations in the last row. In contrast to CE-based attacks, Jitter-based attacks mainly attack the most salient regions of the image. Thus, the ℓ_2 norm of Jitter-based attacks is considerably smaller, while Jitter-based attacks are still more effective than prior attacks

attacks by injecting noise into the model output. Additionally, we introduce scale invariance to the loss function by normalizing the output logits to a fixed value range. Thereby, we circumvent the gradient obfuscation problem generated by models with low-confidence predictions or irregularly large output logits [16, 21]. Moreover, we propose a simple yet effective mechanism that minimizes the magnitude of perturbations, as shown in Fig. 2, without compromising the success rate of an attack. This leads to the definition of an objective function for adversarial attacks, which we will refer to as *Jitter*. We empirically evaluate our loss function on an extensive benchmark consisting of 30 different models proposed in the literature. We show that Jitter-based attacks consistently outperform prior attacks in all 30 analyzed models by up to 11.8 percentage points. Additionally, Jitter-based attacks generate perturbations with a 4.20 times smaller norm on average. Lastly, we analyze the effect of Jitter on the classification decisions to explain its effectiveness. Pytorch-like pseudo code to compute the Jitter loss and a Jitter-based PGD attack is given in the Appendix[1].

2 Notation

Let $f_\theta : [0,1]^d \rightarrow \mathbb{R}^C$ be a DNN classifier parameterized by $\theta \in \Theta$ with $f_\theta : x \mapsto z$. Here x is a d-dimensional input image, z is the respective output vector (logits) of the DNN, and C denotes the number of classes. The ground truth class label of a given image is described by $y \in \{1, \ldots, C\}$ while the predicted class label $\hat{y} \in \{1, \ldots, C\}$ is given by $\text{argmax}(z)$. The confidence values for every class are given by $\text{softmax}(z)$.

Adversarial examples $x_{adv} = x + \gamma$ aim to change the input data of DNNs such that the classification decision of the network is altered, but the class label remains the same for human perception. Additionally, x_{adv} is constrained in the data domain, i.e., $x_{adv} \in [0,1]^d$. A common way to enforce semantic similarity to the original sample is to restrict the magnitude ϵ of the adversarial perturbation γ by an ℓ_p-norm bound, such that $\|\gamma\|_p \leq \epsilon$. We refer to the set of valid adversarial examples that fulfill these constraints as S. As prior work mainly focuses on $p = \infty$ and thus most models are available for this threat model, we focus on $p = \infty$ in this work as well. Furthermore, we restrict our analysis to untargeted white-box adversarial attacks, as done in prior work [16, 22].

3 Related work

One of the most often used adversarial attacks, Projected Gradient Descent (PGD), was proposed by [10]. PGD is an iterative gradient-based attack, in which multiple smaller gradient updates are used to find the adversarial perturbation

$$x_{adv}^{t+1} = \Pi_S\,(x_{adv}^t + \alpha \cdot \text{sign}(\nabla_x \mathcal{L}(f_\theta(x_{adv}^t), y)) \qquad (1)$$

where $0 < \alpha \leq \epsilon$ and x_{adv}^t describes the adversarial example at iteration t. The loss of the attack is given by $\mathcal{L}(f_\theta(x_{adv}^t), y)$. $\Pi_S(x)$ is a projection operator that keeps x_{adv}^{t+1} within the set of valid adversarial examples S and sign is the component-wise signum operator. The starting point of the attack x_{adv}^0 is randomly chosen in the ϵ-norm ball. Several variants of iterative gradient-based attacks have been proposed that are more effective than vanilla PGD [19, 22–24]. Recently, [16] proposed the Auto-PGD (APGD) attack. In contrast to previous PGD versions, APGD requires considerably less hyperparameter tuning and was shown to be more effective than other PGD-based attacks against a variety of models [16]. Nevertheless, one important component of all gradient-based attacks is their optimization objective. The most commonly used objective is the Cross-Entropy (CE) loss. Carlini and Wagner [21] observe that CE-based attacks fail against models with large logits. They propose the Carlini & Wagner (CW) loss function $-z_y + \max_{i \neq y}(z_i)$ which does not make use of the softmax function and thereby reduces the scaling problem. Nevertheless, [16] observe that the scale dependence of the CW loss can still lead to failed attacks against models with exceptionally large logits. They address this issue with the scale- and shift-invariant Difference of Logit Ratio (DLR) loss and show its effectiveness on an extensive benchmark. Recently, Pintor et al. [25] proposed the Fast Minimum Norm (FMN) attack that is robust to hyperparameter choices, creates low-norm adversarial perturbations, and is computationally less complex than previous attack approaches. Another method to improve the robustness evaluation of machine learning models is to combine multiple conceptually different attacks into an attack ensemble [16, 26]. We explore and discuss the limitations of current adversarial attacks in the following section.

4 Robust misclassifications

While several benchmarks of adversarial robustness exist in the literature, they generally do not investigate the classification decisions of multiple models simultaneously [16]. To investigate common limitations among models in

[1] Code is available at: https://git.io/JyJ0N

Table 1 Summary of the most important abbreviations and symbols used in the methods and experiments sections of this work

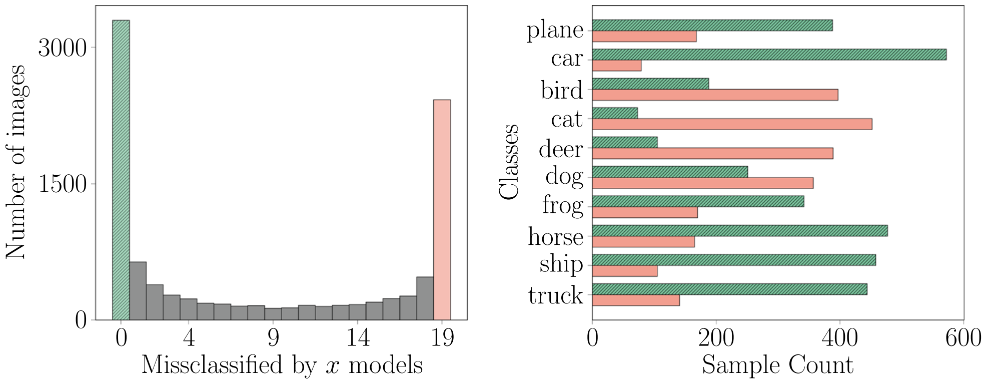
Symbol	Description
PGD	Projected Gradient Descent
APGD	Auto-PGD
DLR	Difference of Logit Ratio
CW	Carlini & Wagner
FMN	Fast Minimum Norm
CE	Cross-Entropy
$\hat{z}$	Rescaled and normalized output logits of a neural network
α	Rescaling factor of the output logits
$\hat{s}$	Softmax output of the rescaled logits $\hat{z}$
$\hat{s}_{Noise}$	$\hat{s}$ perturbed with Gaussian noise $\hat{s}_{Noise} = \hat{s} + \mathcal{N}(0, \sigma)$
σ	Variance of the Gaussian noise $\mathcal{N}(0, \sigma)$ used to perturb $\hat{s}$
Y	One-hot encoded label vectors
$\mathcal{L}_{Jitter}$	Loss function proposed in this work

the literature and get a more holistic overview of model robustness, we explore the classification decisions of 21 out of the 30 different models in the presence of adversarial attacks. We restrict our analysis to the 21 models trained on the CIFAR10 dataset [20], as only a limited amount of pre-trained models are available for the other datasets. The labels "airplane" and "automobile" have been renamed to "plane" and "car", respectively. We choose the recently proposed Auto-PGD (APGD) with the Difference of Logit Ratio (DLR) loss as an attack to perturb the images, as it is one of the most efficient and reliable gradient-based attacks [16]. These choices and specific hyperparameters are described in more detail in Section 6. A summary of the most important symbols and abbreviations used in this work is given in Table 1.

4.1 Distribution of misclassifications

Recent studies mainly focus on common evaluation metrics to assess the robustness of DNNs. This includes the worst-case robustness of a classifier [16] and the magnitude of the perturbation norm necessary to fool the classifier for individual inputs [21]. Here, we provide insights into the classification decisions and numerical properties of a large and diverse set of models from the literature. We focus on models that are trained to be robust to adversarial attacks. Furthermore, all models are attacked individually to find the respective worst-case robustness.

Figure 3a shows how the 19 most robust models misclassify inputs attacked by APGD. We left out the models by [14] and [27] from this analysis, as they show negligible robustness against strong adversarial attacks. Out of the 10,000 test samples of the CIFAR10 dataset, 3298 are correctly classified by all 19 models, while 2423 samples are consistently misclassified by all models. This is shown by the leftmost (green, dashed) and rightmost (red) bars of the histogram plot. This shows that none of the 19 models is able to robustly classify a considerable fraction of the test set. Further, it highlights the vast accuracy gap between adversarial and clean data beyond prior analysis of this trade-off between robustness and accuracy on individual models. Inspired by prior work [28], we will refer to images in the first group that are never misclassified as *robust images* and images in the second group that are always misclassified as *non-robust images*. The gray bars in between show the remaining 4279 samples misclassified by at least one model, but not by all models. Figure 3b summarizes the class distribution of robust and non-robust images. There is a considerable difference in frequency for most classes between the two groups. Images from the



(a)

(b)

Fig. 3 Analysis of misclassification decisions of 19 out of the 21 analyzed CIFAR10 models. The two models by [14] and [27] showed no considerable robustness against strong attacks and were therefore excluded from this analysis. Subfigure (a) shows by how many models each attacked input is misclassified. Robust images that are never misclassified are shown in the leftmost column (green, dashed) and non-robust images that are always misclassified are shown in the rightmost column (red). Subfigure (b) displays the difference between the class distributions between robust and non-robust images. Both statistics are calculated on the test set of CIFAR10

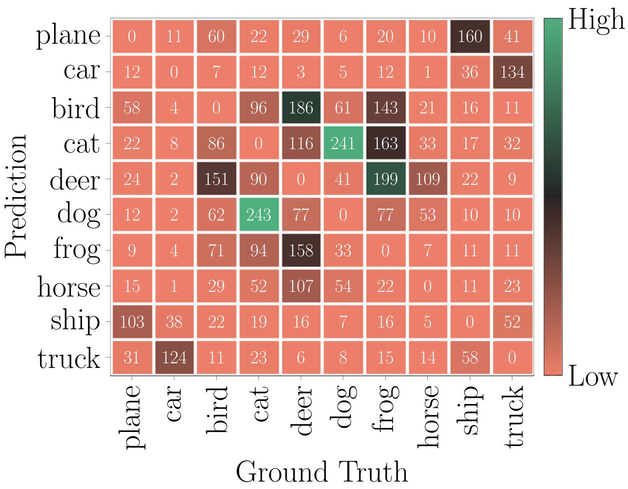
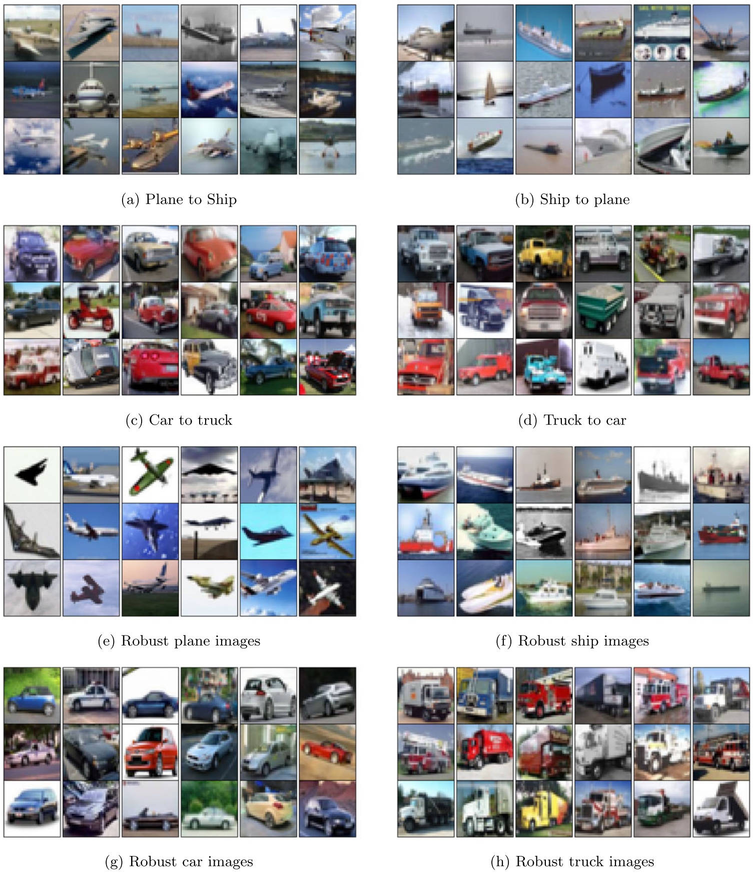


Fig. 4 Averaged confusion matrices of all models for adversarially perturbed inputs for the CIFAR10 dataset (Only misclassified samples are shown). We also observe substantial sparsity in the summed confusion matrix for both all the CIFAR100 and all the ImageNet models, where only a small fraction of all entries is larger than 0

classes “plane”, “car”, “horse”, “ship”, and “truck” are often classified correctly while “bird”, “cat”, “deer” and “dog” are mostly misclassified.

We additionally explored the average of the confusion matrices of all models for adversarially perturbed images. Note that the CIFAR10 dataset is balanced and contains an equal amount of samples for all classes. Figure 4 shows the confusion matrix of only the misclassifications. The confusion matrix contains only a few large values, which is in line with the previous observation that some classes are easier to perturb than others. Furthermore, the matrix is largely symmetric. Classes are mainly confused amongst pairs. This includes semantically meaningful pairs such as “cats” and “dogs” or “car” and “truck”, but it also includes other pairs that generally share similar image backgrounds such as “plane” and “ship”, “deer” and “frog”, and “deer” and “bird”. Examples of non-robust and robust images are shown in Fig. 5. Non-robust images contain outliers and wrongly labeled images. These include:

- Subfigure (a): A seaplane that is classified as a ship.
- Subfigure (b): A ship in the air that is classified as a plane.
- Subfigure (c): A golf cart and an ambulance that are labeled as a car but classified as a truck.
- Subfigure (d) An oldtimer car that is labeled as a truck and classified as a car.


(a) Plane to Ship

(b) Ship to plane

(c) Car to truck

(d) Truck to car

(e) Robust plane images

(f) Robust ship images

(g) Robust car images

(h) Robust truck images

Fig. 5 (a-d) Non-robust images that are correctly classified by all CIFAR10 models under normal conditions but misclassified by all models under attack (DLR-based APGD). Additionally, only examples of images that are misclassified as the same target class by all models (e.g., plane images that are misclassified as ships by all models) are shown. (e-h) Robust images that are correctly classified by all CIFAR10 models even under attack

We additionally observe considerable sparsity of the misclassification confusion matrices of the CIFAR100 [20] and ImageNet [29] datasets. For all CIFAR100 and ImageNet models, only 17% and 1.4% of the entries in the confusion matrix are higher than 0 (note that for an attack that induces optimal target class diversity, 100% and 7.2% of the entries in the confusion matrix would be higher than 0, respectively). Concurrent work by [30] made a similar observation on the ImageNet dataset. They find that untargeted adversarial attacks mostly cause misclassifications into semantically similar classes.

4.2 Attacks against robust and non-robust images

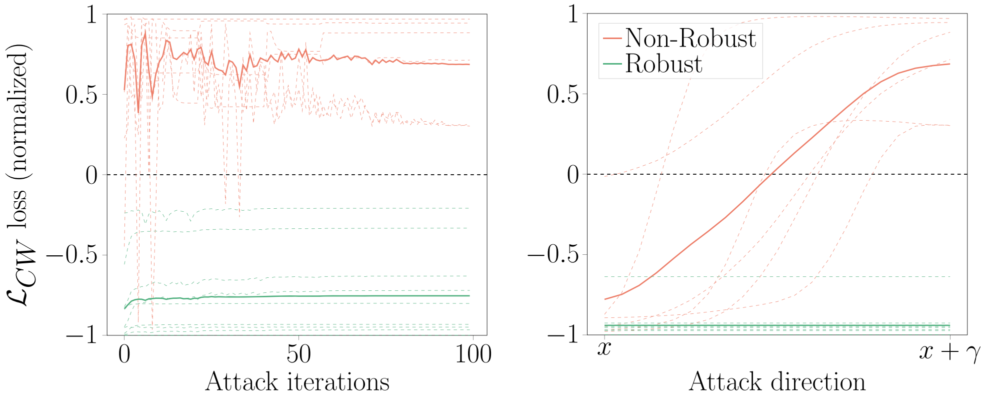
We further analyzed the behavior of the DLR-based adversarial attack for robust and non-robust images. This is exemplified in Fig. 6 for the model proposed by [8]. We explored how the CW loss [21] (y-axis) changes during the attack optimization (x-axis in Fig. 6a) and how the CW loss changes along the direction of the final adversarial perturbation starting from a clean image (x-axis in Fig. 6b). We choose to display the CW loss and not the DLR loss as the CW loss can directly be related to the classification decision of a classifier (inputs with $\mathcal{L}_{CW} > 0$ are misclassified). Note that we still use the DLR loss in the attack optimization and only use the CW loss for display purposes. We additionally calculated the CW loss on the softmax output of the network such that the output is scaled between -1 and 1. The subfigures show the mean loss value over the sets of robust and non-robust images by a solid line. The individual loss values for 10 randomly drawn samples from each set are shown as dashed lines. For non-robust images, the CW loss increases rapidly during the first attack iterations and most images are successfully attacked in the first attack iteration. Moreover, for non-robust images, the CW loss increases steadily along the direction of the final adversarial perturbations on average, which indicates that the initial gradient direction is a good approximation of the final attack direction. In contrast, for robust images following the gradient directions in the vicinity of the original image is not effective and the CW loss changes only marginally during the attack.

4.3 Logit and confidence distribution

Next, we analyze the distribution of the output logits z and the confidence of all CIFAR10 models and relate these properties to the difficulty of the robustness evaluation. Prior work observed that simply scaling the output of a DNN will lead to vanishing gradients when the softmax function is used in the last layer of the network [16]. This phenomenon occurs due to finite arithmetic and thus limited precision, where the CE loss is quantized to 0 and the model effectively obfuscates the gradient from the attack. The CE loss is given by

$$\mathrm{CE}(z, y) = -\log(\mathrm{softmax}(z)_y) \quad \text{where } \mathrm{softmax}(z)_y = \frac{e^{z_y}}{\sum_{j=1}^{C} e^{z_j}}. \tag{2}$$

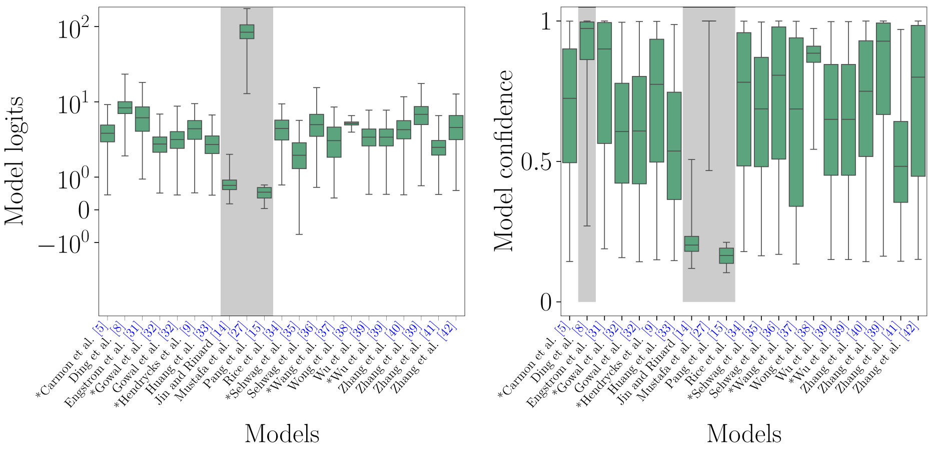
Figure 7 summarizes the logit and confidence distributions for the analyzed CIFAR10 models. The model proposed by [27] shows exceptionally large logits that are outside of the floating-point precision, thus obfuscating its gradients. Furthermore, the models by [27] and [8] exhibit a considerably higher average confidence (0.948) than all other models (0.666 excluding models with exceptionally low confidence [14, 15]). In contrast, the models by [14]



(a) Loss during attack optimization

(b) Loss along attack direction

Fig. 6 Analysis of the CW loss [21] (y-axis) for robust and non-robust images during (a) DLR-based adversarial attack optimization and (b) along the direction of the final adversarial attack perturbation γ found by the DLR-based attack. In (b) x describes a clean image and $x + \gamma$ the adversarial example. For both images, the average loss over the whole sets of robust and non-robust images is shown by a solid line. Additionally, the loss for 10 individual examples from each of the sets is shown by dashed lines



(a) Logits distribution

(b) Confidence distribution

Fig. 7 (a) Box plots of the logits distribution of clean images from the CIFAR10 test set for all models analyzed in this paper. (b) Box plots of the confidence distribution of all models. Only the highest softmax output for every prediction is considered. The models highlighted by gray shading and text show considerably lower robustness against strong attacks compared to standard PGD [8, 14, 15, 27]. Models that use additional data during training are marked with a *

and [15] reveal a different phenomenon where the logits are close to zero and show a substantially lower standard deviation than the other models. Consequently, the logits are generally mapped to a limited value range by the softmax function, where all values are similar. Thus, the loss may only change slightly between different attack iterations, decreasing the attack performance. This is also reflected in low confidence for the two models, where the most confident prediction has a probability of < 0.51 while it is ≈ 1 for all other models.

Note that prior work already observed that models with exceptionally large logits are difficult to attack [16, 21]. However, we observe that the robustness of all 4 models identified in the above analysis is difficult to assess in practice, including models with above or below average confidence but without exceptionally large logits. For these 4 models, the difference between a standard robustness evaluation with CE-based PGD and stronger attacks is larger than 7% and considerably less accurate than for the other 17 models [31]. This highlights that the distribution of the output logits z can be a possible failure case for an accurate robustness evaluation, even when the logits are not exceptionally large. We explain how we combat this problem in comparison to prior work in Section 5.1.

5 Enhancing adversarial attacks

In the previous section, we explored the misclassification of robust DNNs under adversarial attacks. The experiments showed a general consistency between the different models. Specifically, we discovered that common attacks mostly focus on a limited amount of different classes to attack in the untargeted setting. At the same time, current attacks often fail to find adversarial examples if the initial gradient direction is not a good approximation of the final attack direction and cannot change the classification loss even slightly in these cases. Additionally, we observed that the scale and distribution of the output logits are linked to the success rate of adversarial attacks. Based on these observations, we now design a novel loss function for adversarial attacks to make them more effective. We first describe the two main components of this loss function. Subsequently, we elaborate on how we can minimize the norm of the final adversarial perturbation without compromising the attack's success rate. This is important as adversarial attacks should not change the label for human perception, which is linked to the perturbation magnitude.

5.1 Scale invariance

Previous work has already demonstrated that high output logits can lead to gradient obfuscation and weaken adversarial attacks [16, 21]. We additionally observe that a small value range of the logits can also lead to attack failure. We propose to scale the output logits by the following rule:

$$\hat{z} = \alpha \cdot \frac{z}{\|z\|_\infty} \tag{3}$$

where α is an easy-to-tune scalar value that controls the lowest and highest possible output values of the softmax function. After rescaling, the logits are within a fixed value range $\hat{z} \in [0, \alpha]^C$, which solves the aforementioned problems. We additionally define the scaled softmax output as $\hat{s} = \text{softmax}\left(\hat{z}\right)$, where softmax is the element-wise softmax operator. While other loss functions are already designed to be scale invariant [16] or handle large output logits [21] they are not suited to be combined with the loss function modification that we propose in the next section.

5.2 Attack target diversity and attack exploration

Figure 4 demonstrates that untargeted adversarial attacks mainly induce misclassifications for a limited amount of classes. We argue that this behavior limits the effectiveness of adversarial attacks. This is further supported by prior work showing that performing targeted attacks against every possible class is usually more effective than applying a single untargeted attack [16, 32]. However, these so-called multi-targeted attacks are computationally expensive and do not scale to datasets with a large number of output classes. Besides, in Fig. 6, we show that current adversarial attacks have difficulties changing the loss for robust images. We argue that this stems from a bad trade-off between attack exploitation and attack exploration. Gradient-based attacks exploit the local gradient information to find adversarial examples with no incentive to explore.

To address the above problems, we propose to perturb the scaled softmax output of a model after each forward pass with Gaussian noise $\hat{s}_{\text{Noise}} = \hat{s} + \mathcal{N}(0, \sigma)$ to prompt adversarial attacks to further explore the input space. Here, the noise magnitude is controlled by the hyperparameter σ.

Still, the CE loss is only dependent on the output of the ground truth class and adding noise to the other output values has no impact. Other loss functions such as DLR or CW have non-normalized logits, which make it difficult to find a suitable σ, as the logit range changes between every input. Instead, we exchange the CE loss function with the Euclidean distance loss between the one-hot encoded ground truth vector Y of the class label and the scaled output of the model. The proposed rescaling makes it easier to tune the σ hyperparameter for individual models in our experiments. We additionally observe the scaled Euclidean distance loss to be more effective than the DLR or CW loss even without noise injection (see Table 2). More details are given in Section 6. The loss function is given by the following equation:

$$\mathcal{L}_2 = \|\hat{s} - Y\|_2. \tag{4}$$

Combining the Euclidean loss function and the scaling described in (3) the loss function can be described by the following equation.

$$\mathcal{L}_{Noise} = \|\hat{s}_{\text{Noise}} - Y\|_2. \tag{5}$$

Injecting gradient noise to improve the convergence of optimization algorithms is well-motivated by previous work [33–36]. Neelakantan et al. [36], found that adding noise to the weight updates of a neural network during training does not only improve the generalization ability of the model but additionally leads to a lower training loss. They attribute this to the additional exploration of the parameter space induced by the noise. Furthermore, non-gradient-based algorithms like simulated annealing [33] or genetic algorithms [34] utilize randomness to escape local optima in non-convex optimization landscapes. However, to the best of our knowledge enhancing gradient-based adversarial attacks by adding noise during the optimization has not been investigated by existing work.

Table 2 Ablation results for the individual Jitter components for the model proposed by [14]

Attack	Accuracy	Improvement
APGD_{CE}	52.34	N/A
$\text{APGD}_{\text{CE \& Scaled}}$	18.29	+34.05
$\text{APGD}_{\text{Scaled \& L2}}$	18.13	+0.16
$\text{APGD}_{\text{Scaled \& L2 \& Noise}}$	7.54	+10.59
APGD_{DLR}	21.22	N/A
$\text{APGD}_{\text{DLR \& Noise}}$	7.61	+13.61
APGD_{CW}	47.78	N/A
$\text{APGD}_{\text{CW \& Noise}}$	35.21	+12.57

5.3 Minimizing the norm of the perturbation

Finally, we aim to encourage the attack to find small perturbations. As long as no successful perturbation is found, we apply the loss function presented in (5). As soon as the adversarial attack is able to change the predicted label of the model, we additionally aim to minimize the norm of the adversarial perturbation. Furthermore, we only override the current perturbation if the norm-minimized perturbation also leads to a successful attack. This procedure can never decrease the success rate of the attack and effectively minimizes the norm of the adversarial perturbation in our experiments. In addition, the norm (or other distance measures) can be freely chosen according to the respective problem (e.g., ℓ_1, ℓ_2, ℓ_∞) as long as it is differentiable. The final loss function can be defined as

$$\mathcal{L}_{Jitter} = \begin{cases} \frac{\|\hat{s}_{\text{Noise}} - Y\|_2}{\|\gamma\|_p} & \text{if } \hat{y} \neq y \\ \|\hat{s}_{\text{Noise}} - Y\|_2 & \text{if } \hat{y} \equiv y \end{cases}. \tag{6}$$

The effect of the different components is exemplified in Table 2 for the model proposed in [14]. Every component decreases the accuracy of the model and therefore increases the success rate of the attack. The norm minimization does not affect the performance and is therefore excluded from the table. We additionally analyzed the influence of noise injection for the DLR and CW loss functions. However, since the logits of the DLR and CW loss functions are not normalized we additionally scaled the sigma value by the largest logit for every sample in the batch. Note that noise injection to the output does not improve the performance when using the CE loss in our experiments. This may be attributed to the fact that the CE loss is only dependent on the output of the ground truth class and adding noise to the other output values has little impact.

6 Experiments

We conducted a series of experiments to evaluate the effectiveness of the proposed Jitter loss function. Furthermore, we inspect the perturbations generated with the Jitter loss function to explain its effectiveness compared to other state-of-the-art loss functions. All experiments were conducted on a single NVIDIA V100 GPU.

6.1 Data and models

All experiments were performed on the CIFAR10, CIFAR100 [20], and ImageNet datasets [29]. We chose CIFAR10 for our initial analysis as many pre-trained models exist for this dataset. CIFAR100 and ImageNet were used to evaluate if the findings on CIFAR10 and the proposed Jitter loss generalize to more complicated classification tasks. We gathered 30 models from the literature for the attack evaluation. All models were either taken from the RobustBench library [31] or from the GitHub repositories of the authors directly [14, 27, 39, 49]. We excluded some models from RobustBench as we had difficulties getting them to run correctly. We only considered models which are trained to be robust against ℓ_∞-norm attacks. The resulting benchmark contains a diverse set of models which are trained with different methods.

6.2 Threat model

We compare the performance of different loss functions for the Auto-PGD (APGD) attack [16], which is one of the state-of-the-art iterative gradient-based attacks. Moreover, APGD has no hyperparameters such as step size and thus enables a less biased comparison between different loss functions. We compare Jitter to three different loss functions and two popular gradient-based adversarial attacks. This includes the Cross-Entropy (CE) loss, which is the standard loss function for training supervised DNNs and is the most often used loss function for gradient-based adversarial attacks. We also consider the Carlini & Wagner (CW) loss proposed by [21] that shows considerably better results compared to CE when the model shows high output logits. Additionally, we include the Difference of Logit Ratio (DLR) loss proposed in [16] that was shown to achieve more stable results compared to the CE and CW loss. Lastly, we compare Jitter to the recently proposed B&B and the Fast Minimum Norm (FMN) attacks, which have shown to be effective against several different defenses and robust to hyperparameter choices [22, 25]. All attacks are untargeted ℓ_∞-norm attacks and use 100 attack iterations. We use a perturbation budget of $\epsilon = 8/255$ for CIFAR10 and CIFAR100 models and a perturbation budget of $\epsilon = 4/255$ for ImageNet models.

6.3 Jitter hyperparameter

Compared to CE and DLR, Jitter introduces two additional hyperparameters. The first hyperparameter α rescales the softmax input and directly controls the possible minimum and maximum value of the output logits and the average magnitude of the gradient. Note that values for α close to or greater than ≈ 88 will result in an overflow of 32-bit float values ($e^{88} \approx 3.402823 \cdot 10^{38}$) in the softmax function and thereby lead to numerical issues. Thus, we can focus on $0 < \alpha \ll 88$. In a preliminary experiment, we explored different values for α between 2 and 20 and observed a stable performance for all values and therefore chose $\alpha = 10$ for all remaining experiments. The second hyperparameter σ controls the amount of noise added to the rescaled softmax output $\hat{s}$. We tuned σ for every model individually on a batch of 100 samples by testing values for $\sigma \in \{0, 0.05, 0.1, 0.15, 0.2\}$. Note that tuning σ on a small batch for each model introduces only a negligible overhead ($\approx 1\%$ additional runtime). Additionally, we analyzed the sensitivity of the attack performance with respect to σ for all models. Values between 0.05 and 0.2 resulted in similar success rates, while values above 0.25 decreased the attack performance compared to no noise injection on average.

7 Results and discussion

In this section, we summarize and discuss the results of the experiments.

7.1 Attack performance

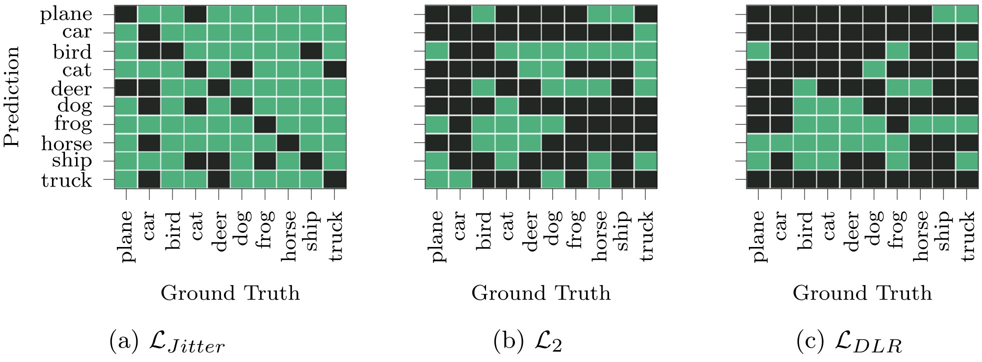
Table 3 compares the performance of the different loss functions on the CIFAR10, CIFAR100, and ImageNet datasets. The best result for every model is highlighted in bold. The best attack for every model is highlighted in bold. The minimum and maximum difference between Jitter and the other attacks is shown in the two rightmost columns. The proposed Jitter loss achieves superior performance compared to the other attacks for 29 out of 30 models. For the model proposed in [27] the B&B and FMN attacks achieve a marginally higher success rate in the 100 attack iterations. Nevertheless, every iteration of the B&B and FMN attack are considerably slower than an iteration of Jitter-based attacks in our experiments (We use the implementation provided in the GitHub repositories of the authors). Jitter still achieves 100% success rate

Table 3 Accuracy [%] of the evaluated models when attacked with ℓ_∞ norm APGD attacks using different loss functions

Model	CE	CW	DLR	B&B	FMN	Jitter	Min Diff.	Max Diff.
CIFAR10								
Mustafa et al. [27]	19.12	0.10	0.05	**0.00**	**0.00**	0.14	-0.14	19.1
Jin and Rinard [14]	52.34	47.78	21.33	19.48	50.12	**7.60**	11.80	44.7
Wong et al. [37]	45.83	45.95	47.05	47.66	46.14	**44.54**	1.29	3.12
Zhang et al. [38]	46.12	47.15	47.71	49.58	47.36	**46.01**	0.11	3.57
Ding et al. [8]	50.14	51.07	51.29	48.22	50.21	**47.88**	0.34	3.41
Engstrom et al. [39]	51.77	52.27	53.09	53.89	52.5	**51.09**	0.68	2.80
Zhang et al. [40]	54.80	53.53	53.64	55.54	53.89	**53.03**	0.50	2.51
Huang et al. [41]	55.86	53.94	54.41	55.85	54.34	**53.25**	0.69	2.61
Zhang et al. [42]	56.84	54.49	54.78	56.45	54.81	**53.97**	0.52	2.87
Rice et al. [43]	56.9	55.36	56.0	55.58	55.42	**54.44**	0.92	2.46
Pang et al. [15]	61.62	55.44	56.29	57.35	55.63	**54.46**	0.98	7.16
Sehwag et al. [44]	57.32	56.35	56.86	57.84	56.48	**55.30**	1.05	2.54
Hendrycks et al. [9]*	57.15	56.44	57.23	58.67	56.6	**55.94**	0.50	2.73
Wu et al. [45]	58.80	56.76	56.82	59.34	57.08	**56.60**	0.16	2.74
Gowal et al. [46]	59.50	57.82	57.61	59.28	57.59	**57.18**	0.41	2.32
Wang et al. [47]*	61.82	58.23	58.95	60.99	58.6	**57.64**	0.59	4.18
Sehwag et al. [48]*	59.61	58.31	58.45	60.84	58.64	**57.72**	0.59	3.12
Zhang et al. [49]*	66.45	60.21	60.40	62.38	60.49	**59.66**	0.55	6.79
Carmon et al. [7]*	61.74	60.61	60.88	62.72	61.18	**60.12**	0.49	2.6
Wu et al. [45]*	63.32	60.62	60.67	63.12	61.13	**60.38**	0.24	2.94
Gowal et al. [46]*	65.69	63.76	63.92	65.28	64.30	**63.40**	0.36	2.29
CIFAR100								
Rice et al. [43]	20.54	20.20	20.44	22.37	20.38	**19.51**	0.69	2.86
Sitawarin et al. [50]	26.31	26.79	27.38	29.27	27.10	**25.52**	0.79	3.75
Chen et al. [51]	30.96	28.27	28.51	30.38	28.24	**27.54**	0.70	3.42
Cui et al. [52]	29.94	28.17	29.62	31.99	28.37	**27.73**	0.44	4.26
Hendrycks et al. [9]	32.92	30.73	32.08	32.85	30.67	**29.41**	1.26	3.51
Cui et al. [52]**	34.01	30.29	30.85	32.22	30.49	**29.45**	0.84	4.56
Wu et al. [45]	33.28	30.90	31.26	33.09	30.93	**29.46**	1.44	3.82
ImageNet								
Wong et al. [37]	26.89	27.12	27.50	30.86	26.54	**26.15**	0.39	4.71
Engstrom et al. [39]	32.14	32.40	33.01	32.33	33.57	**30.33**	1.81	3.24

The minimum and maximum difference between the Jitter and the other attacks is shown in the two rightmost columns. The most successful attack is highlighted in bold (lowest accuracy), the second best is underlined, and models that use additional data are marked with *. Models from the same authors with different hyper parameters or architecture are marked with **. The abbreviations of the loss functions and attacks can be found in Table 1

faster than the B&B and FMN attack, which makes it the most efficient attack in all experiments. Furthermore, compared to the other attacks Jitter achieves the same success rate 49% faster than CE-based attacks, 37% faster than CW-based attacks, 35% faster than DLR-based attacks, 162% faster than B&B attacks, and 46% faster than FMN attacks on average. Moreover, the Jitter loss is the only loss function that is consistently better than the other loss functions. In contrast, the other five attacks differ in performance for the individual experiments as shown in Table 3, where the second-best attack is underlined in each row. Combining the DLR and CW loss with noise injection led to equal or higher success rates in all cases but both losses remain less effective than Jitter on average. A more extensive overview is given in Appendix C. To evaluate the performance of Jitter with a higher computational budget we compared DLR and Jitter using 1000 model evaluations (5 restarts and 200 iterations) for all CIFAR10 models. While the success rate increased up to 6.51% for Jitter, the high-budget version of DLR performed worse than 100 iteration Jitter in all cases. The results are summarized in Appendix D.



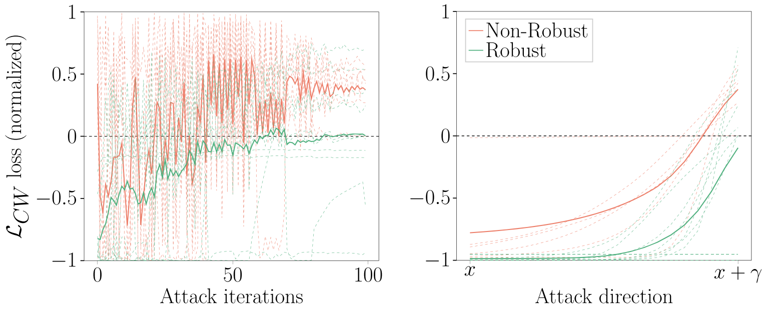
(a) $\mathcal{L}_{Jitter}$ (b) $\mathcal{L}_2$ (c) $\mathcal{L}_{DLR}$

Fig. 8 Illustration of the attack target diversity for $\mathcal{L}_{Jitter}$-based, $\mathcal{L}_2$-based, and $\mathcal{L}_{DLR}$-based attacks. Subfigure (a), (b), and (c) show binarized confusion matrices for the different attacks, where more green squares indicate a higher target diversity

7.2 Induced misclassifications

We designed Jitter to increase target diversity for untargeted adversarial attacks and compare the target diversity of the different loss functions for all 30 models from all explored datasets. The average increase in target diversity of Jitter compared to the other loss functions is: CE: 36%, CW: 52%, DLR: 155%, and $\mathcal{L}_2$: 57% (given in (4)). Moreover, noise injection into the output for the CW and DLR loss increases the attack target diversity by 49% and 56%, respectively. This shows that noise injection to the output increases target diversity and not the $\mathcal{L}_2$ loss. Nevertheless, the combination of noise injection with the $\mathcal{L}_2$ loss was more effective in our experiments than the combination of noise injection with CW and DLR (more details are given in Appendix C). Figure 8 exemplifies the increased attack target diversity of $\mathcal{L}_{Jitter}$-based attacks for the model proposed in [8]. Green squares denote that an attack changed the classification decision to the respective class at least once. $\mathcal{L}_{Jitter}$-based attacks show a considerably higher amount of different target classes compared to the other two attacks. Further, the $\mathcal{L}_{DLR}$-based attack was not able to successfully attack the classes car and truck, which reduces the attack success rate compared to Jitter.

Analogous to the analysis presented in Fig. 6, we explored the behavior of Jitter-based adversarial attacks for robust and non-robust images for the same model [8]. The different subfigures of Fig. 9 show the CW loss [21] on the y-axis during the attack optimization (Fig. 9a) and along the direction of an adversarial perturbation (Fig. 9b). As in Fig. 6 the individual loss values for 10 randomly drawn samples are shown as dashed lines for the sets of robust and non-robust images, while the mean over the whole sets is shown as a solid line. In comparison to DLR-based attacks (Fig. 6), Jitter-based attacks (Fig. 9) exhibit a considerably larger fluctuation of the loss during the attack optimization for both robust and non-robust images. In contrast to DLR-based attacks, Jitter-based attacks also show considerable loss changes for robust images and thereby achieve higher success rates. Besides, while DLR-based attacks generally find adversarial directions which directly increase the CW loss, Jitter-based attacks mainly find adversarial directions which do not directly increase the CW loss, which can be seen by the constant mean near the clean input x in Fig. 9b. Moreover, the mean CW loss value of Jitter-based attacks exceeds the threshold of misclassification ($\mathcal{L}_{CW} = 0$) noticeably later than DLR-based attacks even for non-robust images (Jitter:0.87, DLR:0.48). DLR-based attacks



(a) Loss during attack optimization (b) Loss along attack direction

Fig. 9 Analysis of the CW loss [21] (y-axis) during (a) $\mathcal{L}_{Jitter}$-based adversarial attack optimization and (b) along the direction of the final adversarial attack perturbation γ found by the Jitter-based attack. In (b) x describes a clean image and $x + \gamma$ the adversarial example. For both images, the average loss over the whole sets of robust and non-robust images is shown by a solid line. Additionally, the loss for 10 individual examples from each of the sets is shown by dashed lines

by design follow the direction of the steepest ascent. In contrast, Jitter-based attacks have a better trade-off between attack exploration and attack exploitation due to the injected noise. This enables Jitter-based attacks to find perturbation directions that are sub-optimal in the beginning but lead to a misclassification at the final adversarial perturbation.

7.3 Attack norm and structure

In a final experiment, we examined the average perturbation norm of the different attack configurations for all 30 models. We choose to minimize the ℓ_2 norm with Jitter, as differences in the ℓ_2 norm are easier to interpret than for the ℓ_∞ norm (e.g. the attack focusing on specific regions). The average ℓ_2 perturbation norm over all samples for the different loss functions are: CE: 0.52, CW: 0.54, DLR: 0.55, B&B: 1.29, FMN: 1.09, and Jitter: 0.19. Jitter achieves considerably lower average perturbation norms than the other attacks. In contrast to Jitter, the B&B and FMN attacks are not able to minimize the perturbation norm considerably with 100 attack iterations. Note that the average ell_∞ norm of both B&B and FMN are also considerable larger than for Jitter in our experiments (B&B: 0.024, FMN: 0.022, Jitter: 0.009). An overview is given in Fig. 10.

We also inspect the structure of the perturbations. Figure 2 displays the perturbation for CE- and Jitter-based attacks for several images. To plot the perturbations, we calculate the absolute sum over every color channel and show the magnitude as a color gradient, where no change is denoted by black color. CE-based attacks generally attack every pixel in an image. In comparison, Jitter-based attacks mainly focus on the salient regions of an image. We enforce this by regularization of the perturbation γ within our loss function and thereby enable Jitter-based attacks to create successful and low-norm adversarial perturbations. In an ablation study, we found that combing other loss functions with the norm-minimization objective described in (5) successfully reduces the perturbation norm of those attacks. For this experiment, we evaluated CE-based, CW-based, and $\mathcal{L}_2$-based attacks. However, in our experiments the perturbation norm of Jitter-based attacks was always the lowest on average. This is expected, as we only minimize the perturbation norm of an attack once the attack is successful. This is done to prevent the perturbation norm minimization from reducing the effectiveness of the attack. As Jitter-based attacks find successful adversarial perturbations faster than other attacks in our experiments (see Section 7.1), the number of iterations that are dedicated to minimizing the perturbation norm is greater for Jitter-based attacks compared to other attacks.

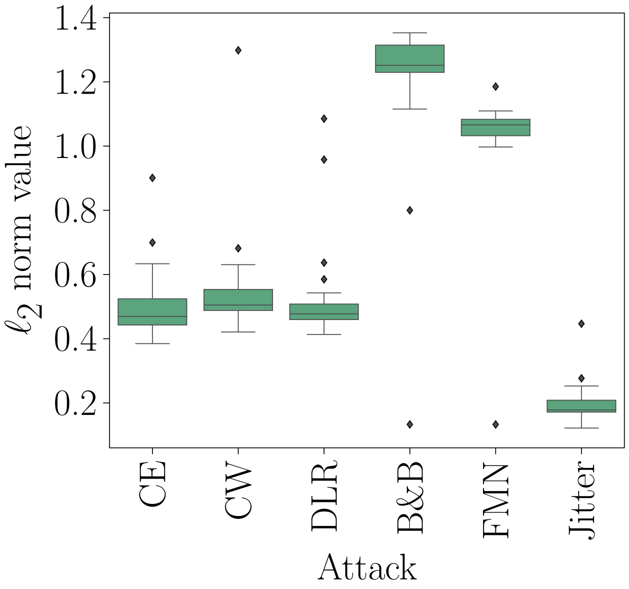


Fig. 10 Box plots that show the ℓ_2 norm perturbation magnitude distribution for all models between the different loss functions

7.4 Discussion

In an extensive benchmark study, we compared the proposed Jitter loss function to other adversarial attacks from the literature. Our experiments showed that Jitter achieves higher attack success rates and efficiency compared to prior methods. Moreover, Jitter-based attacks exhibit lower perturbation norms. The proposed Jitter loss has two key components that lead to increased attack success rate and efficiency.

1) **Scale invariance.** Previous work observed that exceptionally large output logits in neural networks reduce the efficiency of gradient-based attacks [21]. We further found that gradient-based attacks are also inefficient when models show exceptionally low or high confidence values. Both issues can be solved by normalizing and rescaling the output logits of neural networks to a specific range. In addition, rescaling the output logits of neural networks makes it simpler to integrate noise injection into the Jitter loss function. For a more detailed explanation, refer to Section 5.2. Noise injection is the second key component that improves the success rate of Jitter-based attacks.
2) **Noise Injection.** Our experiments revealed that existing untargeted adversarial attacks induce misclassifications in only a limited amount of target classes. We show that the target class diversity can be increased by injecting noise into the scaled softmax output of the model. We observe that this simultaneously increased the success rate of the attack, which indicates a connection between the target class diversity of an untargeted attack and its effectiveness. Moreover, noise injection also increased the success rate of previous adversarial attack

Table 4 Pytorch-like implementation of the Jitter loss function

Algoritm 1: Jitter loss

```
# X: input data, X_adv: adversarial input data, B: batch size
# z: logits, y: labels, Y: one-hot encoded labels
# alpha: value range, sigma: noise magnitude, norm: norm to minimize
# Function call: Jitter(X, X_adv, B, z, y, Y, alpha, sigma, norm)
############################## logit scaling ###########################
z_scaled = z / norm(z.view(B, -1), p=float(‘‘inf’’), dim=1, keepdim=True)
z_scaled = softmax(z_scaled * alpha, dim=1)
z_noisy = z_scaled + randn_like(z_scaled) * sigma
############################## l2 loss #################################
l2 = norm((z_noisy - Y).view(B, -1), p=2, dim=1)
############################## perturbation magnitude ##################
non_adversarial_mask = z.argmax(1) != y
magnitude = norm((X - X_adv).view(B, -1), p=norm, dim=1)
masked_magnitude = ones_like(l2)
masked_magnitude[non_adversarial_mask] = magnitude[non_adversarial_mask]
############################## final loss ##############################
loss = l2 / masked_magnitude
return loss
```

methods in our experiments and is not limited to the proposed Jitter loss function. The effect of injecting noise into the output of a neural network was studied from a theoretical perspective in prior work, which could yield another explanation for the effectiveness of noise injection apart from the empirical observation

Table 5 Pytorch-like pseudo-code of an untargeted PGD-like attack using the Jitter loss function

Algoritm 2 Jitter-based PGD-like attack

```
# f: neural network, X: input data, y: labels, Y: one-hot encoded labels
# N: number of attack iterations, step_size;  step size of the attack
# eps: maximum perturbation norm

X_adv = X.clone()
B = X.shape(0)
alpha = 10
sigma = 0.1
norm = 2

for i in range(N):

  z = f(X)
  loss = Jitter(X, X_adv, B, z, y, Y, alpha, sigma, norm)
  loss.backward()
  gradients = X_adv.grad.sign() * step_size
  X_adv = X_adv + gradients

 # project X_adv to the l-norm ball and to the a valid range (i.e., (0,1))
  X_adv = torch.max(torch.min(X_adv, X + eps), X - eps)
  X_adv = X_adv.clamp((0, 1))
return X_adv
```

The attack is conducted in the ℓ_∞ norm of increased target class diversity. Zhu et al. [53] demonstrate that using gradient Langevin dynamics (GLD) instead of regular gradient descent can help to escape local minima during optimization from a theoretical perspective. The ability to escape sharp and poor local minima could also improve the effectiveness of adversarial attacks. However, in GLD Gaussian noise is directly added to the gradient and not to the output before performing the gradient calculation. Further investigations are necessary to clarify the connection between Jitter and GLD.

8 Conclusion and outlook

In this paper, we analyze the classification decisions of a diverse set of models that are trained to be robust against adversarial attacks. This analysis gives an indication of the limits of the robustness of current models on the CIFAR10 dataset. We utilize insights from our analysis to create a novel loss function which we name Jitter that increases the efficiency and success rate of adversarial attacks. Specifically, we enforce scale invariance of the lossfunction and encourage attack exploration and a diverse

Table 6 Accuracy [%] of the evaluated models when attacked with APGD using different loss functions

Model	CE	CW	DLR	B&B	CW noise	DLR noise	Jitter
CIFAR10							
[27]	19.12	0.1	0.05	**0.0**	0.01	0.04	0.02
[14]	52.33	47.78	21.22	19.48	35.21	7.61	**7.6**
[37]	45.83	45.95	47.05	47.66	**44.26**	45.7	44.54
[38]	46.12	47.15	47.71	49.58	**45.86**	46.11	46.01
[8]	50.13	51.07	51.29	48.22	48.38	**47.34**	47.88
[39]	51.77	52.27	53.09	53.89	**50.92**	51.43	51.09
[40]	54.80	53.53	53.64	55.54	53.23	53.36	**53.03**
[41]	55.86	53.94	54.41	55.85	53.42	53.56	**53.25**
[42]	56.84	54.49	54.77	56.45	54.04	54.84	**53.97**
[43]	56.89	55.36	56.00	5558.	54.55	54.65	**54.44**
[15]	61.62	55.44	56.28	57.35	54.74	54.66	**54.45**
[44]	57.32	56.35	56.86	57.84	55.81	56.65	**55.30**
[9]	57.15	56.44	57.23	58.67	**55.33**	55.87	55.94
[45]	58.8	56.76	56.82	59.34	**56.48**	57.01	56.59
[46]	59.5	57.82	57.60	59.28	57.24	57.33	**57.18**
[47]*	61.82	58.23	58.95	60.99	**57.63**	58.26	57.64
[48]*	59.61	58.30	58.45	60.84	58.14	58.37	**57.72**
[49]*	66.45	60.20	60.4	62.38	60.07	60.57	**59.66**
[7]*	61.73	60.61	60.88	62.72	60.52	60.41	**60.12**
[45]*	63.32	60.62	60.67	63.12	**60.37**	60.5	60.38
[46]*	65.69	63.75	63.92	65.28	**63.16**	63.39	63.4
CIFAR100							
[43]	20.54	20.20	20.44	22.37	20.01	20.25	**19.50**
[50]	26.31	26.79	27.38	29.27	26.15	27.02	**25.52**
[51]	30.95	28.27	28.51	30.38	27.96	28.4	**27.54**
[52]	29.94	28.17	29.62	31.99	27.93	27.87	**27.73**
[9]	32.92	30.73	32.08	32.85	30.0	29.84	**29.40**
[52]**	34.01	30.29	30.85	32.22	29.95	30.7	**29.45**
[45]	33.28	30.9	31.25	33.09	30.20	30.66	**29.45**

The difference between the best and second-best loss function is given in the right-most column. The most successful attack is highlighted in bold, the second best is underlined, and models that use additional data are marked with *. Models from the same authors with different hyper parameters are marked with ** set of target classes for the attack by adding Gaussian noise to the output logits. In addition to the analysis on CIFAR10, we show that the proposed attack generalizes to two other benchmark datasets CIFAR100 and ImageNet. Our experiments demonstrate that analyzing failure cases of adversarial attacks over multiple models at the same time is an effective way to design stronger adversarial attacks. The proposed method shows superior attack efficiency for all 30 analyzed models for all three datasets compared to five other popular attacks from the literature. In all cases, Jitter achieved a higher success rate in a shorter amount of time. Moreover, the average perturbation norm of Jitter-based attacks is considerably lower compared to prior methods, which is achieved without compromising the success rate of the attack. Future work will explore if using Jitter for adversarial training can further improve the robustness of models against strong attacks. Theoretical analysis was beyond the scope of this paper but will be explored in future work. This includes connections between the proposed Jitter loss function and gradient Langevin dynamics.

Appendix A: Jitter code

The Jitter loss can be implemented with just a few lines of code, which makes it easy to combine with prior approaches. Table 4 shows a PyTorch-like implementation of Jitter.

Appendix B: Jitter adversarial attack

Table 5 exemplifies how Jitter can be combined with adversarial attack algorithms like PGD.

Appendix C: DLR and CW with noise injection

The performance of DLR- and CW-based attacks with noise injection is shown in Table 6. The proposed Jitter loss function achieves the highest success rate most often. Moreover, injecting noise to the logits of the other two loss functions is highly effective as well. Here the performance is equal or better in all cases compared to no noise injection.

Appendix D: Attack performance for a higher computational budget

The performance of DLR- and Jitter-based attacks for more model evaluations is shown in Table 7. Attacks with a

Table 7 Accuracy [%] of the CIFAR10 models when attacked with APGD using either the DLR or Jitter loss function

Models	DLR	Jitter	DLR strong	Jitter strong	Min Diff.
Mustafa et al. [27]	0.05	0.02	0.03	0.00	0.02
Jin and Rinard [14]	21.33	7.54	12.43	1.03	6.51
Wong et al. [37]	47.05	44.49	46.69	43.45	1.04
Zhang et al. [38]	47.71	46.01	47.31	45.79	0.22
Ding et al. [8]	51.29	47.85	50.19	43.62	4.23
Engstrom et al. [39]	53.09	51.08	52.59	50.83	0.24
Zhang et al. [40]	53.64	53.05	53.42	52.88	0.17
Huang et al. [41]	54.41	53.33	54.24	53.25	0.09
Zhang et al. [42]	54.77	53.98	54.54	53.64	0.34
Rice et al. [43]	56.00	54.36	55.70	53.66	0.70
Pang et al. [15]	56.28	54.48	55.97	54.10	0.38
Sehwag et al. [44]	56.86	55.30	56.56	54.65	0.65
Hendrycks et al. [9]	57.23	55.94	56.98	55.10	0.84
Wu et al. [45]	56.82	56.45	56.69	56.10	0.35
Gowal et al. [46]	57.60	57.09	57.44	57.08	0.01
Wang et al. [47]	58.95	57.58	58.55	57.28	0.31
Sehwag et al. [48]	58.45	57.66	58.23	57.50	0.15
Zhang et al. [49]*	60.40	59.66	60.11	59.16	0.5
Carmon et al. [7]	60.88	60.08	60.62	59.90	0.19
Wu et al. [45]	60.67	60.44	60.56	60.19	0.25
Gowal et al. [46]	63.92	63.31	63.74	62.73	0.57

Attacks with the keyword "strong" suffix use 200 iterations and 5 restarts, while the other attacks use 100 iterations without additional restarts "strong" suffix use 200 iterations and 5 restarts, while the other attacks use 100 iterations without additional restarts. Low-budget Jitter-based attacks achieve a higher success rate than both normal and strong DLR-based attacks in all cases. Overall more model evaluations do only marginally improve the performance for DLR-based attacks except for the model proposed by [14], where the success rate increases by 8.9 percentage points. For Jitter-based attacks more model evaluations improve the performance considerably for the models proposed by [14] and [8] and slightly for the models proposed by [37, 43], and [9].

Acknowledgements Bjoern Eskofier gratefully acknowledges support by the German Research Foundation (DFG) within the framework of the Heisenberg professorship program (grant number ES434/8-1). An Nguyen gratefully acknowledges support by the Siemens Healthcare GmbH, Germany.

Author Contributions Conceptualization: Leo Schwinn proposed the initial idea. Dario Zanca and Bjoern Eskofier contributed further ideas; Methodology: The initial methodology was proposed by Leo Schwinn. René Raab and An Nguyen contributed with ideas and discussion of critical points; Conducting experiments; Leo Schwinn; Writing - original draft preparation: Leo Schwinn; Writing - review and editing: Leo Schwinn, René Raab, An Nguyen, Dario Zanca, and Bjoern Eskofier; Supervision: Dario Zanca and Bjoern Eskofier

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Competing interests The authors have no competing interests to declare that are relevant to the content of this article. All the data used in the experiments is open source and freely available.

References

1. Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow IJ, Fergus R (2014) Intriguing properties of neural networks. In: International conference on learning representations, ICLR
2. Qin Y, Carlini N, Cottrell GW, Goodfellow IJ, Raffel C (2019) Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In: International conference on machine learning, ICML of proceedings of machine learning research, PMLR, vol 97, pp 5231–5240
3. Hu S, Shang X, Qin Z, Li M, Wang Q, Wang C (2019) Adversarial examples for automatic speech recognition: attacks and countermeasures. IEEE Commun Mag 57(10):120–126. https://doi.org/10.1109/MCOM.2019.1900006
4. Morris JX, Lifland E, Yoo JY, Grigsby J, Jin D, Qi Y (2020) Textattack: a framework for adversarial attacks, data augmentation, and adversarial training in NLP. In: Conference on empirical methods in natural language processing: system demonstrations, EMNLP, demo track, association for computational linguistics, pp 119–126
5. Yang P, Chen J, Hsieh C-J, Wang J-L, Michael I, Jordan MI (2020) Greedy attack and gumbel attack: generating adversarial examples for discrete data. J Mach Learn Res JMLR 21(43):1–36
6. Ren K, Zheng T, Qin Z, Liu X (2020) Adversarial attacks and defenses in deep learning. Engineering 6(3):346–360. https://doi.org/10.1016/j.eng.2019.12.012. ISSN 2095-8099
7. Carmon Y, Raghunathan A, Schmidt L, Duchi JC, Liang P (2019) Unlabeled data improves adversarial robustness. In: Advances in neural information processing systems, NeurIPS, pp 11190–11201
8. Ding GW, Sharma Y, Lui KYC, Huang R (2020) MMA training: direct input space margin maximization through adversarial training. In: International conference on learning representations, ICLR
9. Hendrycks D, Lee K, Mazeika M (2019) Using pre-training can improve model robustness and uncertainty. In: International conference on machine learning, ICML of proceedings of machine learning research, PMLR, vol 97, pp 2712–2721
10. Madry A, Makelov A, Schmidt L, Tsipras D, Vladu A (2018) Towards deep learning models resistant to adversarial attacks. In: 6th International conference on learning representations, ICLR
11. Leon Bungert, Raab R, Roith T, Schwinn L, Tenbrinck D (2021) CLIP: cheap lipschitz training of neural networks. In: Scale space and variational methods in computer vision, SSVM of lecture notes in computer science. Springer, vol 12679, pp 307–319
12. Schwinn L, Nguyen A, Raab R, Bungert L, Tenbrinck D, Zanca D, Burger M, Eskofier BM (2021a) Identifying untrustworthy predictions in neural networks by geometric gradient analysis. In: Conference on uncertainty in artificial intelligence, UAI of proceedings of machine learning research, AUAI Press, vol 161, pp 854–864
13. Richardson E, Weiss Y (2021) A bayes-optimal view on adversarial examples. J Mach Learn Res JMLR 22(221):1–28
14. Jin C, Rinard M (2020) Manifold regularization for adversarial robustness. arXiv:2003.04286
15. Pang T, Yang X, Dong Y, Xu T, Zhu J, Su H (2020) Boosting adversarial training with hypersphere embedding. In: Advances in neural information processing systems, NeurIPS
16. Croce F, Hein M (2020) Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In: International conference on machine learning, ICML, vol 119 of proceedings of machine learning research, pp 2206–2216. PMLR
17. Athalye A, Carlini N, Wagner DA (2018) Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples. In: International conference on machine learning, ICML, vol 80 of proceedings of machine learning research, pp 274–283. PMLR
18. Tramèr F, Carlini N, Brendel W, Madry A (2020) On adaptive attacks to adversarial example defenses. In: Larochelle H, Marc'Aurelio Ranzato RH, Balcan M-F, Lin H-T (eds) Advances in neural information processing systems, NeurIPS

19. Uesato J, O'Donoghue B, Kohli P, van den Oord A (2018) Adversarial risk and the dangers of evaluating against weak attacks. In: International conference on machine learning, ICML, vol 80 of proceedings of machine learning research, pp 5032–5041. PMLR
20. Krizhevsky A (2009) Learning multiple layers of features from tiny images. Tech Rep
21. Carlini N, Wagner DA (2017) Towards evaluating the robustness of neural networks. In: 2017 IEEE symposium on security and privacy, SP, pp 39–57. IEEE Computer Society
22. Brendel W, Rauber J, Kümmerer M, Ustyuzhaninov I, Bethge M (2019) Accurate, reliable and fast robustness evaluation. In: Advances in neural information processing systems, NeurIPS, pp 12841–12851
23. Lin J, Song C, He K, Wang L, Hopcroft JE (2020) Nesterov accelerated gradient and scale invariance for adversarial attacks. In: International conference on learning representations, ICLR
24. Schwinn L, Nguyen A, Raab R, Zanca D, Eskofier BM, Tenbrinck D, Burger M (2021) Dynamically sampled nonlocal gradients for stronger adversarial attacks. In: International joint conference on neural networks, IJCNN, pp 1–8. IEEE
25. Pintor M, Roli F, Brendel W, Biggio B (2021) Fast minimum-norm adversarial attacks through adaptive norm constraints. In: Advances in neural information processing systems, NeurIPS, pp 20052–20062
26. Mao X, Chen Y, Wang S, Su H, He Y, Xue H (2021) Composite adversarial attacks. In: Conference on artificial intelligence, AAAI, pp 8884–8892. AAAI Press
27. Mustafa A, Khan SH, Hayat M, Goecke R, Shen J, Shao L (2019) Adversarial defense by restricting the hidden space of deep neural networks. In: IEEE/CVF international conference on computer vision, ICCV, pp 3384–3393. IEEE
28. Ilyas A, Santurkar S, Tsipras D, Engstrom L, Tran B, Madry A (2019) Adversarial examples are not bugs, they are features. In: Advances in neural information processing systems, NeurIPS, pp 125–136
29. Deng J, Dong W, Socher R, Li LJ, Kai Li, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: IEEE/CVF computer society conference on computer vision and pattern recognition CVPR, pp 248–255. IEEE
30. Ozbulak U, Pintor M, Van Messem A, De Neve W (2021) Evaluating adversarial attacks on imagenet: a reality check on misclassification classes. In: Advances in neural information processing systems, NeurIPS, workshop on ImageNet: past, present, and future
31. Croce F, Andriushchenko M, Sehwag V, Debenedetti E, Flammarion N, Chiang M, Mittal P, Hein M (2021) Robustbench: a standardized adversarial robustness benchmark. In: Advances in neural information processing systems, NeurIPS, track Datasets and benchmarks
32. Kwon H, Kim Y, Park KW, Yoon H, Choi D (2018) Multi-targeted adversarial example in evasion attack on deep neural network. IEEE Access 6:46084–46096
33. Kirkpatrick S, Gelatt D, Vecchi M (1983) Optimization by simulated annealing. Science 220(4598):671–680
34. Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning. Mach Learn 3:95–99
35. Hinton GE, Roweis ST (2002) Stochastic neighbor embedding. In: Advances in neural information processing systems, NeurIPS, pp 833–840. MIT Press
36. Neelakantan A, Vilnis L, Le QV, Sutskever I, Kaiser L, Kurach K, Martens J (2015) adding gradient noise improves learning for very deep networks. CoRR arXiv:1511.06807
37. Wong E, Rice L, Kolter JZ (2020) Fast is better than free: revisiting adversarial training. In: International conference on learning representations, ICLR
38. Zhang D, Zhang T, Lu Y, Zhu Z, Dong B (2019) You only propagate once: accelerating adversarial training via maximal principle. In: Advances in neural information processing systems, NeurIPS, pp 227–238
39. Engstrom L, Ilyas A, Salman H, Santurkar S, Tsipras D (2019) Robustness python library. https://github.com/MadryLab/robustness. [Accessed May 25th, 2021]
40. Zhang H, Yu Y, Jiao J, Xing EP, El Ghaoui L, Jordan MI (2019) Theoretically principled trade-off between robustness and accuracy. In: International conference on machine learning, ICML, vol 97 of proceedings of machine learning research, pp 7472–7482. PMLR
41. Huang L, Zhang C, Zhang H (2020) Self-adaptive training: beyond empirical risk minimization. In: Advances in neural information processing systems, NeurIPS
42. Zhang J, Xu X, Han B, Niu G, Cui L, Sugiyama M, Kankanhalli MS (2020) Attacks which do not kill training make adversarial learning stronger. In: International conference on machine learning, ICML, vol 119 of proceedings of machine learning research, pp 11278–11287. PMLR
43. Rice L, Wong E, Kolter JZ (2020) Overfitting in adversarially robust deep learning. In: International conference on machine learning, ICML, vol 119 of proceedings of machine learning research, pp 8093–8104. PMLR
44. Sehwag V, Mahloujifar S, Handina T, Dai S, Xiang C, Chiang M, Mittal P (2021) Improving adversarial robustness using proxy distributions. CoRR arXiv:2104.09425
45. Wu D, Xia S-T, Wang Y (2020) Adversarial weight perturbation helps robust generalization. In: Advances in neural information processing systems, NeurIPS
46. Gowal S, Qin C, Uesato J, Mann TA, Kohli P (2020) Uncovering the limits of adversarial training against norm-bounded adversarial examples. CoRR arXiv:2010.03593
47. Wang Y, Zou D, Yi J, Bailey J, Ma X, Gu Q (2020) Improving adversarial robustness requires revisiting misclassified examples. In: International conference on learning representations, ICLR
48. Sehwag V, Wang S, Mittal P, Jana S (2020) HYDRA: pruning adversarially robust neural networks. In: Advances in neural information processing systems, NeurIPS
49. Zhang J, Zhu J, Niu G, Han B, Sugiyama M, Kankanhalli MS (2021) Geometry-aware instance-reweighted adversarial training. In: International conference on learning representations, ICLR
50. Sitawarin C, Chakraborty S, Wagner DA (2020) Improving adversarial robustness through progressive hardening. CoRR arXiv:2003.09347
51. Chen J, Cheng Y, Gan Z, Gu Q, Liu J (2022) Efficient robust training via backward smoothing. In: Conference On artificial intelligence, AAAI, pp 6222–6230. AAAI Press
52. Cui J, Liu S, Wang L, Jia J (2021) Learnable boundary guided adversarial training. In: IEEE/CVF international conference on computer vision, ICCV, pp 15701–15710. IEEE
53. Zhu Z, Wu J, Yu B, Wu L, Ma J (2020) The anisotropic noise in stochastic gradient descent: its behavior of escaping from sharp minima and regularization effects. In: International conference on machine learning, ICML , vol 97 of proceedings of machine learning research, pp 7654–7663. PMLR