Splitting the Difference on Adversarial Training

Matan Levi
Ben-Gurion University of the Negev
matanle@post.bgu.ac.il

Aryeh Kontorovich
Ben-Gurion University of the Negev
karyeh@bgu.ac.il

Abstract

The existence of adversarial examples points to a basic weakness of deep neural networks. One of the most effective defenses against such examples, adversarial training, entails training models with some degree of robustness, usually at the expense of a degraded natural accuracy. Most adversarial training methods aim to learn a model that finds, for each class, a common decision boundary encompassing both the clean and perturbed examples. In this work, we take a fundamentally different approach by treating the perturbed examples of each class as a separate class to be learned, effectively splitting each class into two classes: "clean" and "adversarial." This split doubles the number of classes to be learned, but at the same time considerably simplifies the decision boundaries. We provide a theoretical plausibility argument that sheds some light on the conditions under which our approach can be expected to be beneficial. Likewise, we empirically demonstrate that our method learns robust models while attaining optimal or near-optimal natural accuracy, e.g., on CIFAR-10 we obtain near-optimal natural accuracy of 95.01% alongside significant robustness across multiple tasks. The ability to achieve such near-optimal natural accuracy, while maintaining a significant level of robustness, makes our method applicable to real-world applications where natural accuracy is at a premium. As a whole, our main contribution is a general method that confers a significant level of robustness upon classifiers with only minor or negligible degradation of their natural accuracy.

1 Introduction

Despite their success in a wide variety of challenging tasks, Neural Networks are brittle when faced with small, imperceptible perturbations to their input; these are commonly referred to as *adversarial examples*, which will, with high probability, alter the neural network's classification [10, 11, 25, 29, 41, 47, 59, 68, 69, 72, 88]. Early methods for defense against such attacks were soon broken by stronger adversaries [4]; subsequently, *adversarial training* emerged as one of the most effective defenses [29, 45, 68, 96]. These adversarial training techniques aim to learn robust models by solving a min-max optimization problem.

While the inner maximization searches for worst-case adversarial examples during training, and then augments the training data with them, the outer minimization optimizes across model parameters given natural and adv. examples.

Usually, in standard adversarial training methods, each generated adversarial example is annotated with the source class label. Some works also attach a domain label (*clean* or *adversarial*) for enhanced techniques, such as using an additional domain classifier [44], adversarial examples detection [10, 70], etc. On the contrary, we hypothesize that adversarial examples generated from a given source class induce a totally distinct class distribution. Therefore, in case one wishes to avoid significant natural accuracy degradation, adversarial training should be adjusted to take these additional classes into account during training. Overall, we make the following contributions:

- We introduce a novel approach for training robust models, which departs from the established paradigm of attempting to learn a common decision boundary for each natural class and its adversarially perturbed version. Rather, we claim that for each class, the adversarial perturbations induce a distinct distribution on the examples, so much so that it makes more sense to learn it as a separate label, rather than attempting to shoehorn it into the original one. Thus, our method doubles the number of classes but ends up learning much simpler decision boundaries; we provide both theory and experiments in evidence of the efficacy of this trade-off (more below). To our knowledge, this approach of "splitting the difference" (which we formally dub *Double boundary adversarial training*, DBAT), is completely novel in the adversarial training setting.
- We perform a comprehensive battery of experiments to demonstrate that our approach learns robust models

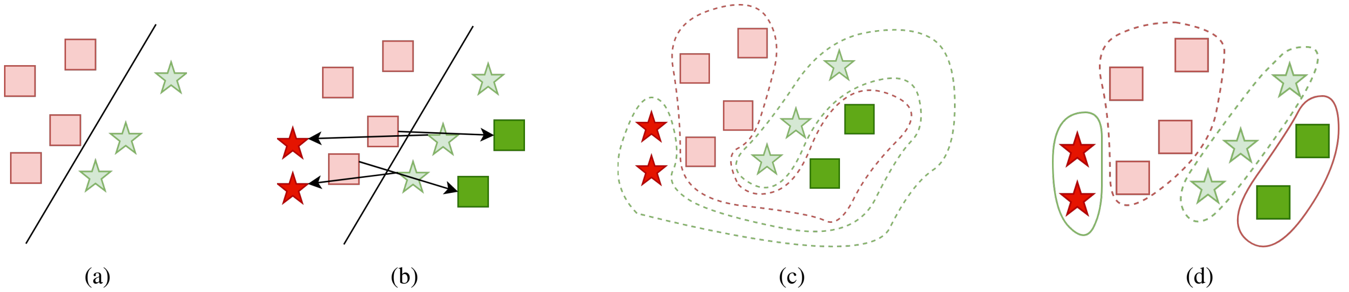


Figure 1: Conceptual illustration of our method compared to standard adversarial training. (a) standard training decision boundary (b) generation of adversarial examples that cross the decision boundary (c) standard adversarial training that tries to learn boundaries for both clean and adversarial examples of each class (d) "splitting the difference" (our method). The dashed lines represent the original classes' learned boundaries. The solid lines represent the adversarial classes learned by DBAT. Standard adversarial training learns complex shared boundaries for the two classes, while our method learns four much simpler boundaries.

while also achieving the highest reported natural accuracy, with a significant margin across different datasets. This optimal or near-optimal natural accuracy makes our method applicable for real-world applications (autonomous vehicles, face recognition systems, healthcare monitoring, and diagnosis, etc.) that cannot sacrifice natural accuracy for robustness. We stress that our aim is not to compete with the state-of-the-art models on robustness, but rather a general-purpose technique for endowing a classifier with a significant level of robustness, while only incurring a minor degradation of natural accuracy. [1]

- In §4, we provide a rudimentary plausibility argument to shed some theoretical light on the statistical trade-off presented by DBAT: an increased number of classes to learn, but with much simpler boundaries.

2 Related work

Since the discovery of adversarial examples by [68], a wide range of defenses were proposed to enhance robustness. Among these, adversarial training [29, 45] emerged as one of the most successful methods to train robust models. Madry *et al.* [45] proposed a technique, commonly referred to as standard Adversarial Training (AT), to minimize the cross entropy loss only on adversarial examples with respect to the original class labels. Throughout the years, standard adversarial training was enhanced in various ways [7] – with changes in the regularization terms [28, 38, 40, 42, 44, 80, 96], model ensemble [50, 72, 90], adversarial training with adaptive attack budget [15, 23], curriculum adversarial training [9, 79, 97], utilizing out-of-distribution data [43], applying Stochastic Weight Averaging (SWA) [36] to flatten the adversarial loss landscape [14, 31], adapting adversarial training to model weights using Adversarial Weight Perturbation (AWP) [73, 85], and combining adversarial training with data augmentation techniques [32, 56, 57] and synthetically generated data [49, 61, 81, 89].

Other lines of research include theoretically certified approaches [16, 30, 53, 54, 63, 82, 84], computationally efficient adversarial training [2, 62, 67, 83, 94], robust overfitting and possible mitigations [58], semi/un-supervised adversarial training [12, 75, 93], adversarial self-training and pretraining [13, 37], incorporating domain adaptation alongside adversarial training [44, 66], and robust model architecture and custom building blocks [77, 86, 87, 95]. Specifically, normalizer-free robust training (NoFrost) [77] suggested removing all batch normalization (BN) layers from the network during AT, but this approach was shown to have a negative effect on the robustness against stronger attacks. [2]

Some well-known methods include [96], who proposed the method TRADES, which uses the Kullback-Leibler (KL) divergence as a regularization term to push the decision boundary away from the data.

Most related methods to DBAT are ones that suggest changes to the regularization terms of adversarial training (AT) with the goal of reducing natural accuracy degradation in AT. A recent work by [20], named *LBGAT*, used an additional Mean Square Error (MSE) regularization term between the logits of a natural model, alongside the robust. In [15], authors suggested a work named Customized Adversarial Training (*CAT*) which adaptively customizes the perturbation level and the corresponding label for each training sample, but was later shown to suffer from obfuscated gradients [64]. Another work [52] suggested Helper Adversarial Training (*HAT*), which attempts to mimic the discriminative features learned by standard trained networks to improve the accuracy of clean samples with the goal of improving natural accuracy. Recently, Universal Inverse Adversarial Training (*UIAT*) was suggested by [24] to encourage the model to produce similar output probabilities for an adversarial example and its "in-

[1] Our source code is available on Github.

[2] See: AutoAttack reduces accuracy of NoFrost.

verse adversarial" counterpart, where the counterpart is generated by maximizing the likelihood in the neighborhood of the natural example. Additionally, the authors of [78] recently suggested *Generalist*, which consists of two base learners separately trained within their respective fields and a global learner that aggregates the parameters of base learners during the training process. The parameters of base learners are collected and combined to form a global learner at intervals during the training process.

In contrast to all of the aforementioned methods, our work suggests a fundamentally different approach. While other methods treated the generated adversarial examples of a given class as additional instances of that same class when learning the class boundaries (or even used only the adversarial examples), our method acknowledges the fact that adversarial examples induce additional class distributions on the source dataset, which essentially doubles the number of classes in the dataset, and therefore these examples should be treated as additional classes of the dataset. We underline that our goal is not to improve robust accuracy compared to the current state-of-the-art in robustness, but rather to equip models with a significant level of robustness, while keeping their natural accuracy as high as possible.

3 Double Boundary Adversarial Training

In this section, we introduce our approach for training robust models, *Double Boundary Adversarial Training*, DBAT. A conceptual illustration is presented in Figure 1, and is supported empirically in Appendix D, where we test if such a case is possible in real-world scenarios by calculating the distance histogram of random examples to the decision boundary.

3.1 Motivation behind Double Boundary Adversarial Training

Tsipras *et al*. [74] argued that robustness may be at odds with natural accuracy, and usually the trade-off is inherent. We concur that this indeed is typically the case when adversarial examples are assigned to the same class as the natural examples they were generated from. However, when separating the adversarial examples from their source, and generating new parallel adversarial classes, we may be able to maintain natural accuracy while still achieving significant robustness.

Therefore, we suggest an alternative approach to learning robust models. Our main hypothesis is that natural examples and their adversarial counterparts should not necessarily be assigned to the same class. Instead, for each class, we learn an additional counterpart adversarial class, which will be assigned to the adversarial examples. In essence, the number of classes in the dataset is doubled.

In other words, instead of learning shared boundaries for both natural examples and their adversarial counterparts, where adversarial examples are expected to reside in the same class as their natural counterparts, we suggest treating adversarial examples as additional classes in the dataset.

While other methods can be thought of as modifying existing boundaries, DBAT learns boundaries for completely new dynamically generated adversarial classes. We hypothesize that this behavior creates a trade-off, where on one hand, DBAT does not induce significant changes to existing boundaries for natural classes in terms of complexity, and keeps them smoother (as we empirically demonstrate on the synthetic dataset experiment in §3.4 and Figure 2) - which mitigates the drop in natural accuracy. But on the other hand - it's a more challenging task to learn completely new classes, which in turn can impact robustness in some tasks.

We highlight that our aim is not to compete with the state-of-the-art on robustness, but rather to find a general-purpose technique that equips classifiers with a significant level of robustness with only minor or neglectable degradation of their natural accuracy.

3.2 Training procedure

During the training process, our goal is to learn additional classes, one for each in the original class set. Given a dataset $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{n}$ with $\mathcal{C}$ classes $\mathcal{Y} = \{0, 1, ..., \mathcal{C}-1\}$, we define a new class space $\mathcal{Y}_{DBAT} = \{0, 1, ..., \mathcal{C}-1, \mathcal{C}, \mathcal{C}+1, ..., \mathcal{C}\cdot 2-1\}$ where class label $\mathcal{C}+k, k \geq 0$ is the label of the adversarial examples generated for class k.

For each natural example (x_i, y_i), DBAT generates an adversarial example x_i' using targeted-PGD with a random target. *Then, the adversarial example is assigned with the adversarial class* $y_i + \mathcal{C}$ *which corresponds to the natural class* y_i *of the natural example*. To summarize, for each natural example (x_i, y_i), we generate adversarial example and assign it the corresponding adversarial class, $(x_i', y_i + \mathcal{C})$. Algorithm 1 describes the training procedure.

Remark. We note that targeted-PGD typically does not outperform untargeted-PGD when used with standard adversarial training methods. That said, we argue that this observation is not applicable to our setting. First, since we aim to learn *new* generic adversarial classes, it stands to reason that class diversity will be conducive to generalization. Therefore, using random targeted-PGD mitigates the scenario where adversarial examples generated by untargeted attacks for a given class focus on small/specific regions of the manifold. Additionally, using untargeted PGD, attacks can potentially be directed to the adversarial class corresponding to the natural one. To avoid the later, one can use untargeted PGD only on the original classes, by adding a projection back to the original classes during the optimization. Finally, we also experimented with using Least-Likely targeted-PGD. See Appendix B for results comparison.

Algorithm 1 DBAT Training

```
**Input:** $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{n}$ with $\mathcal{C}$ classes, and model $f_\theta$
**Parameters:** Batch size $m$, perturbation size $\varepsilon$, attack step size $\tau$, current iteration index $k$ (zero-initialized), and learning rate $\alpha$
repeat
    Fetch mini-batch $X_s = \{x_j\}_{j=1}^{m}$, $Y_s = \{y_j\}_{j=1}^{m}$
    Initialize $X^{'} = \{\}, Y^{'} = \{\}$
    for $j = 1$ to $m$ (in parallel) do
        # Generate an adv. example
        $y'_j =$ Select random label uniformly from $\{0, 1, ..., \mathcal{C} - 1, \mathcal{C}, ..., \mathcal{C} \cdot 2 - 1\} / \{j, j + \mathcal{C}\}$
        $x'_j =$ targeted-PGD$(x_j, y'_j, \varepsilon, \tau, f_\theta)$
        # Save the adv. example with the adv. class label
        $X^{'} = X^{'} \cup \left\{x'_j\right\}$
        $Y^{'} = Y^{'} \cup \{y_j + \mathcal{C}\}$
    end for
    $\theta = \theta - \alpha \cdot \nabla_\theta \ell(X_s \cup X^{'}, Y_s \cup Y^{'})$
    $\theta' = \dfrac{\theta' \cdot k + \theta}{k + 1}$
    $k = k + 1$
until stopping criterion is met
```

3.3 Inference procedure

At inference time, the model will output a probability vector v of size $|v| = 2 \cdot \mathcal{C}$ which corresponds to the double number of classes used during training. However, the dataset originally has only $\mathcal{C}$ classes. Therefore, as our final class prediction, we use the following formula:

$$v^* = (\max(v_0, v_{\mathcal{C}}), ..., \max(v_{\mathcal{C}-1}, v_{2 \cdot \mathcal{C}-1})), \quad (1)$$

$$\text{predicted class} = \underset{0 \le i \le \mathcal{C}}{\operatorname{argmax}} \, v_i^*. \quad (2)$$

In other words, the final class prediction is taken as the class with the maximum probability. If this class is one of the adversarial classes, we return to its natural counterpart.

3.4 Illustrating DBAT's Decision Boundaries using a Synthetic Dataset

In Figure 2 we illustrate how DBAT can learn simpler and smoother decision boundaries by applying it to a synthetic dataset. We exhibit the decision boundaries for standard AT and DBAT. The dataset is composed of isotropic Gaussian blobs with a cluster standard deviation of 0.1 and two features generated using the make_blobs from [51]. The number of samples for each blob is 10,000. The adversary was given a budget of $\varepsilon = 1.2$ optimized for six steps with a step size of 0.2. This enables some of the samples to cross the decision boundary. As can be seen in Figure 2c, our method learns much smoother and simpler decision boundaries as compared to standard adversarial training in Figure 2b.

4 Theoretical analysis

We provide a theoretical plausibility argument for the empirical success of our approach, in the following somewhat idealized setting. We identify a phenomenon, which we term the *DBAT advantage*, which, when applicable, justifies the use of our technique.

Here we assume familiarity with the basic notions of PAC learning, such as the *sample error* of a hypothesis, $\widehat{\text{err}}(h)$, its *generalization error*, $\text{err}(h)$, and the Vapnik-Chervonenkis (VC) dimension of a concept class; these may all be found, e.g., in [3]. Suppose that one trains a k-multiclass classifier by reducing it to k binary classification problems via the standard 1-vs-all method (i.e., a separate in-class/out-class binary classifier is trained for each of the k classes). Suppose for simplicity that each of the classifiers is trained using the same concept class H of VC-dimension V. If h_i is the classifier trained for the ith class on a sample of size n with sample error $\widehat{\text{err}}(h_i)$, then the agnostic PAC bound [3, Theorem 4.9] implies that with probability at least $1 - \delta$,

$$\text{err}(h_i) - \widehat{\text{err}}(h_i) \quad \le \quad c\left(\sqrt{(V + \log(1/\delta))/n}\right), \quad (3)$$

where $\text{err}(h_i)$ is the generalization error and $c > 0$ is a universal constant.

Claim. The following form of (3) to holds for all of the k classifiers, with probability at least $1 - \delta$, simultaneously:

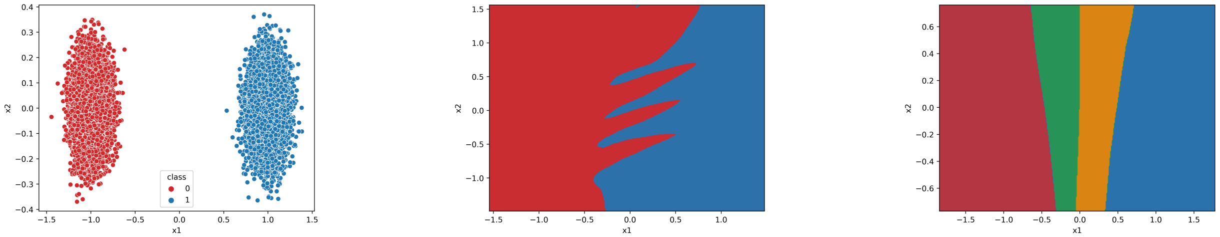
$$\max_{1 \le i \le k} \text{err}(h_i) - \widehat{\text{err}}(h_i) \le c\left(\sqrt{(V + \log(k/\delta))/n}\right) \quad (4)$$

Proof. One sets $\delta' = \delta/k$, which guarantees, with probability at least $1 - \delta'$, a generalization error of at most $c\sqrt{(V + \log(1/\delta'))/n}$ for each class individually, and hence, by a union bound, for all classes simultaneously, with probability of at least $1 - k\delta' = 1 - \delta$. ■

But now suppose that we can express each $h \in H$ as a union of two simpler concepts: $h = h_1 \cup h_2$, where $h_1, h_2 \in H'$, and the latter has VC-dimension, say, $V/2$. In this case, we can formulate the learning problem as a $2k$-multiclass classification problem, over the concept class H'. By assigning $V/2$ and $2k$, the corresponding bound in (4) will now behave as:

$$\sqrt{(V/2 + \log(2k/\delta))n} \quad (5)$$

— which, for constant δ and large V, constitutes considerable savings in sample complexity. The improvement in sample complexity will be even more significant as we consider ℓ-fold (rather than just 2-fold) unions of basic concepts: $h = h_1 \cup h_2 \cup \ldots \cup h_\ell$, $h_i \in H'$. We will refer to this phenomenon — in which decreasing hypothesis complexity while increasing the number of classes reduces the overall sample complexity — as the **DBAT advantage**, and discuss it in greater detail below.



(a) Isotropic Gauss. blobs (boundary $x_1 = 0$)

(b) Standard AT decision boundary

(c) **DBAT** decision boundaries

Figure 2: Synthetic dataset viz. on 2-classes dataset (a) of two 2D features each. Adversary: 6-step ℓ_∞-PGD, $\varepsilon = 1.2$, $\delta = 0.2$.

We will illustrate this phenomenon in some detail on the natural example of halfspaces and Euclidean balls in $\mathbb{R}^d$. Since providing the requisite background on Vapnik-Chervonenkis (VC) theory (in particular: shattering, VC-dimension) is beyond the scope of the paper, we refer the reader to [46].

Halfspaces.

Claim 1. *If H' is the collection of all homogeneous (going through the origin) halfspaces $\mathbb{R}^d$, then the VC-dimension of H' is $d+1$.*

Proof. It is shown in [46, Example 3.2] that the VC-dimension of *general* halfspaces in R^d is $d+1$. The restriction that the halfspace contain the origin can be ensured by translating any shattered set $\{x_1, \ldots, x_{d+1}\}$ by x_1 to obtain the shattered set $\{x_2 - x_1, \ldots, x_{d+1} - x_1\}$ of size d. This shows that homogeneous hyperplanes have VC-dimension 1 less than the general ones, i.e., d. □

Claim 2. *For H' as above (the collection of all homogeneous halfspaces in $\mathbb{R}^d$), the set of all 2-fold unions of concepts from H' will have VC-dimension at least twice that of H.*

Proof. For the lower bound, it suffices to find d points in the positive orthant shattered by a set of 2^d homogeneous halfspaces $H_1 \subset H'$, as well as another set of d points in the negative orthant shattered by another set of 2^d homogeneous halfspaces $H_2 \subset H'$, such that each $h \in H_1$ labels the negative orthant negative, while each $h \in H_2$ labels the positive orthant positive. Evidently, the set of pairwise unions of $h_1 \in H_1$ and $h_2 \in H_2$ shatters the combined set of $2d$ points. □

This example illustrates that 2-fold unions of simple classifiers can double the VC dimension of the hypothesis class. In the more general case of ℓ-fold unions of hyperplanes, it is known [19] that the VC-dimension is $\Omega(\ell d \log \ell)$, so the increase in sample complexity is even more significant. Moreover, [19] showed that this continues to be true for many other kinds of Boolean aggregations: intersections, XORs etc.

Euclidean balls.

Claim 3. *The VC-dimension of Euclidean balls in $\mathbb{R}^d$ is $d+1$.*

Proof. This is a well-known fact, which we prove for completeness and also because the argument will be useful in the sequel.

Both the upper and lower bounds on the VC-dimension of balls rely on the fact that locally, these act like halfspaces: any two finite sets separated by a halfspace can also be separated by a ball of large enough radius (see Figure 4).

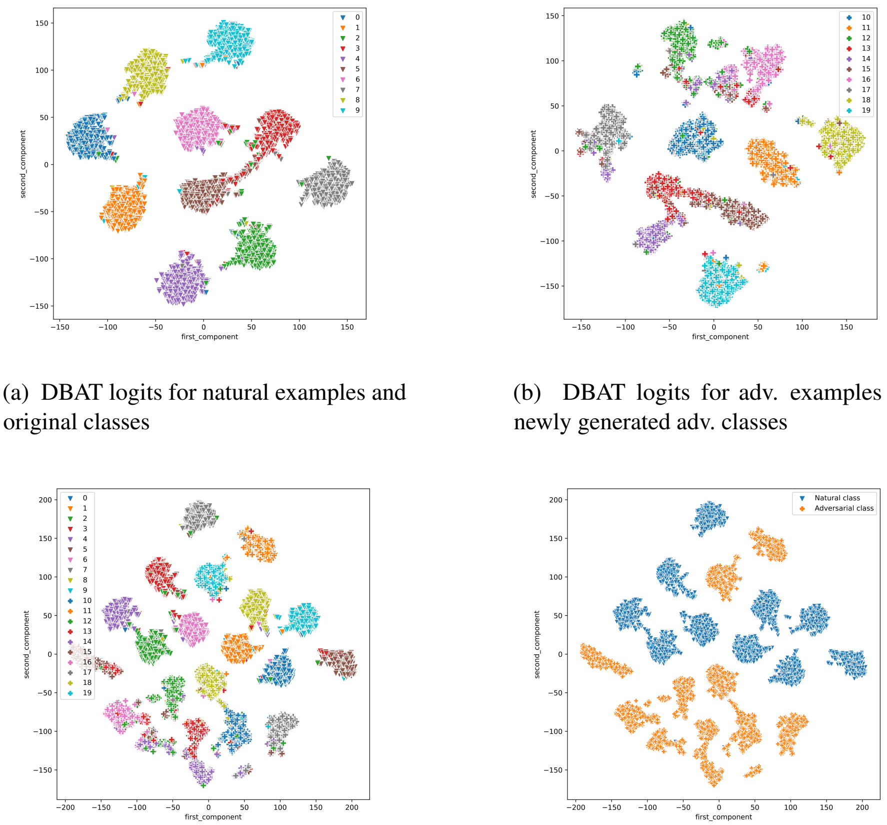
This argument is enough to establish the lower bound: any set that is shattered by general halfspaces is also shattered by Euclidean balls (see Figure 5), and we know from [46, Example 3.2] that such a set can be as large as $d+1$.

For the upper bound, we invoke Radon's theorem [46, Theorem 3.4]: Any set S of $d+2$ points in $\mathbb{R}^d$ can be partitioned into two subsets S_1 and S_2 such that the convex hulls of S_1 and S_2 intersect. Such a partition will be called a *Radon partition*. Suppose, for a contradiction, that the Euclidean balls shatter some set S of $d+2$ points. Then there exists a Radon partition of these into S_1 and S_2. But shattering means that some ball B_1 contains S_1 and not S_2, while another ball B_2 contains S_2 but not S_1.

This means that S_1 and S_2 must be separable by a hyperplane. We conclude that the Euclidean balls cannot shatter any more points that the halfspaces, which is at most $d+1$. □

Claim 4. *If H' is the collection of the Euclidean balls in $\mathbb{R}^d$ (with VC-dimension $d+1$, as shown above), and H is the set of all 2-fold unions of concepts from H', then the VC-dimension of H is at least $2d$*

Proof. The argument proceeds by the same reduction from balls to halfspaces employed in the proof of the lower bound in Claim 3: any set that can be shattered by halfspaces can also be shattered by balls. Now, as in the proof of Claim 2, we construct two disjoint sets shattered by homogeneous halfspaces, consisting of d points each, in the positive and negative orthants, respectively. Each is also shattered by balls, and by dilating the points sufficiently far from the origin, we can ensure that the 2^d balls shattering the positive-orthant set are



(a) DBAT logits for natural examples and original classes

(b) DBAT logits for adv. examples on newly generated adv. classes

(c) DBAT logits for both natural and adv. examples on all classes

(d) DBAT logits in two colors for natural (blue) and adv. examples (orange).

Figure 3: Visualizing the logits of DBAT on CIFAR-10 test set using T-SNE [76] with two components on the model output for (a) natural examples (b) adversarial examples with their new generated adversarial classes (c) combined 2-D visualization of both natural and adversarial examples with all 20 classes (same color for natural class and its adversarial counterpart). (d) same plot as in c, but colored in two colors to observe the separation between natural and adversarial examples. We can observe the strong separation between classes obtained by DBAT, for both original and newly generated classes. Interestingly, adversarial and natural examples almost don't mix, and the majority of mismatches within each cluster are from the same domain (adversarial/natural).

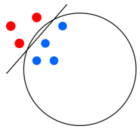


Figure 4: Two finite sets separated by a halfspace can also be separated by a ball of large enough radius.

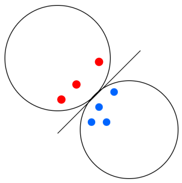


Figure 5: A finite set that is shattered by balls is also shattered by halfspaces.

disjoint from their counterparts shattering the negative-orthant set. Thus, the 2-fold unions of Euclidean balls shatter a set of size $2d$. □

The above discussion was more of a proof-of-concept illustration, since VC-dimension is not a particularly practical tool in analyzing deep neural networks with a large number of weights. In Appendix C, we show that the thrust of our point continues to hold for the Rademacher complexity as well, which is far more practical as far as providing finite-sample generalization bounds [8, 91]. Using the analysis of [26], we show that the Rademacher complexity of ℓ-fold unions grows with ℓ roughly as $\sqrt{\ell}$.

An additional qualification of our plausibility argument is that adversarial loss is distinct from the 0-1 loss discussed above. This is indeed a limitation of our analysis, although in some instances it is possible to control adversarial risk via a VC-type analysis [5, Theorem 2]. Finally, an implicit assumption we have made above is that the adversarial perturbations are *non-adaptive*: the adversary has fixed a (possibly, stochastic) perturbation function in advance of seeing any data — e.g., a neural network trained on a hold-out set, similar to black-box settings. This lets us argue that the examples continue to be iid, under a new (unknown, perturbed) distribution. This assumption, while not entirely realistic, is often made to facilitate analysis [5].

Modulo these qualifications, the above discussion provides evidence that when training k classifiers from a concept class with high complexity, it may be advantageous to decompose them into unions of simpler classifiers. The blow-up of the number of classifiers is more than compensated in the reduction of classifier complexity.

5 Experiments

To emphasize the advantage of Double Boundary Adversarial Training, we conduct extensive evaluations. The evaluation process of DBAT includes white-box and black-box settings, Auto-Attack, natural corruptions [35], unforeseen adversaries, and ablation studies. All results are averaged over 5 runs while omitting one standard deviation. These evaluations demonstrate that the results obtained are not a consequence of what is commonalty referred to as *obfuscated gradients* [4].

We compare our method to some of the most well-known adversarial training methods – Standard AT [45], and TRADES [96], alongside related work – LBGAT [73], Generalist [78], CAT [15], HAT [52], and UIAT [24]. Our evaluation starts with the common CIFAR-10 benchmark. In §5.4 and §5.6, we demonstrate the generalization of our method to other datasets by experiments on CIFAR-100 [39] and SVHN [48]. We use the WRN-34-10 [92] architecture for CIFAR-10 and CIFAR-100, and the PreAct ResNet-18 for SVHN. As suggested in [57], we combine Stochastic Weight Averaging (SWA) [36], and Cutout [21] with window length eight. We used "concatenated batches" as suggested by [65]. Attacks are generated using ℓ_∞-PGD with $\varepsilon = 8/255$, and perturbation step size 1/255 for 10 attack steps. Full experiment settings are detailed in Appendix A.

5.1 Threat model

Our trained model outputs a vector whose dimension is twice the number of classes in the dataset. That is, one-half of the coordinates corresponds to the original classes, while the second half corresponds to the new adversarial classes. Therefore, when considering the adversary's capabilities, specifically for untargeted white-box attacks, we need to explicitly define how the optimization is done. Recall that the aggregation/projection described in §3.3 takes place only at inference time. Moreover, since the projection is not part of the computation graph, the defender can switch it to any desired metric (max, mean, median, log, exp, etc.) at any time during inference, without updating the network. Therefore, there are three possible adversaries:

First, the most basic adversary is one who does not utilize any projection function while attacking.

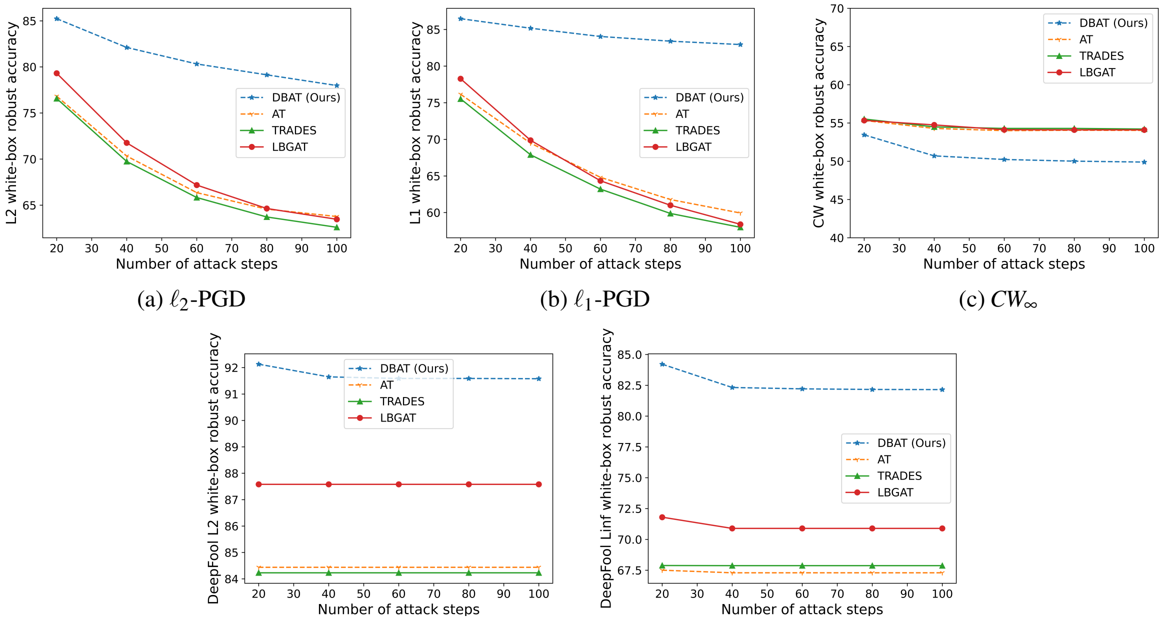
Second, a more advanced (and perhaps most realistic) adversary, is one who knows that the defender is utilizing a projection function, but does not have inference time access to the defender (e.g., a model was published at model zoos), and needs to conjecture the projection function while attacking.

Third, and most powerful adversary (although the least realistic one), is one who can access not only the entire network parameters but also real-time access to the defender's system and projection function at any given time during inference.

Table 1: Natural, PGD , and Auto-Attack (AA) robust accuracy against the fully adaptive white-box perfect knowledge adversary (i.e., "Inference real-time access"). The attack is an ℓ_∞ attack on CIFAR-10 with WRN-34-10. Similar to our method, the presented results are for models that **do not utilize additional data** during training. The Natural method refers to a model trained using standard training under the same hyper-parameters settings. Green represents the best natural accuracy among the robust models. Orange represents the second-best natural accuracy among the robust models. The range alongside the green arrow represents the natural accuracy improvement compared to the other methods.

Method	Natural Acc.	PGD	AA
DBAT (Ours)	**95.01** (↑4–10.1%)	54.61	40.08
AT	85.10	54.46	51.52
TRADES	84.92	55.56	53.08
LBGAT	88.22	54.31	52.86
Generalist	91.03	56.92	52.91
HAT	84.86	52.30	48.85
UIAT	85.01	54.63	49.11
CAT	89.61	73.38	34.78
Natural	95.43	0	0

Throughout the paper, we compare against the most powerful adversary. In §5.8, we present additional experiments to demonstrate how different access to the inference time projection function affects the adversary's strength (i.e., the attacker's ability to degrade robust accuracy). As for black-box attacks analysis, we evaluate against two types of adversaries: naturally trained surrogate models, and other adversarially trained models. For natural corruptions, the corruptions are generated independently from the trained model.



(a) ℓ_2-PGD

(b) ℓ_1-PGD

(c) CW_∞

(d) ℓ_2-DeepFool

(e) ℓ_∞-DeepFool

Figure 6: Robustness against **unforeseen** (a) ℓ_2 PGD adversary (b) ℓ_1 PGD adversary (c) CW_∞ adversary on CIFAR-10.

5.2 White/Black-box and Auto-Attack Evaluation

White-box/Black-box PGD Robustness. We present DBAT's l_∞-PGD white-box and black-box results compared to a variety of adversarial training methods. Attacks are generated with $\varepsilon = 8/255$, and perturbation step size 1/255 and 10 attack steps. On CIFAR-10, DBAT's results are in line with the SOTA methods under black-box attacks. For PGD white-box attacks, DBAT achieves significant PGD robustness (e.g., 54.25% with PGD_{1000}), similar to the other methods, with near-optimal natural accuracy of 95.01% (compared to 95.43% for a naturally trained model). Additionally, in Figure 3 we visually present the strong class separation obtained by DBAT for the original classes, the newly generated adversarial classes, and the combination of all the 20 classes for CIFAR-10.

Auto-Attack Evaluation. We evaluate DBAT on Auto-Attack, an ensemble of diverse attacks: APGD, APGD-DLR [18], Square [1], and FAB [17]. As described in Table 1, our method reaches near-optimal natural accuracy (compared to a naturally-trained model) while still maintaining significant robustness when tested against AA. We note that Auto-Attack results are not as good as PGD results. We ascribe the difference to the adversarial classes that were generated using ℓ_∞-PGD and are therefore oriented towards PGD adversaries. It can be empirically evidenced in the "unforeseen attacks" (Figure 6), where our results on attacks such as C&W are good, but our results on the different PGD adversaries with different norms ($\ell_\infty, \ell_2, \ell_1$) are better.

5.3 Unforeseen Adversaries Robustness

To further demonstrate that our method does not suffer from false robustness, we test it against different adversaries that were not observed during training, including ℓ_2-PGD, ℓ_1-PGD, ℓ_∞-DeepFool, and ℓ_2-DeepFool [47] implemented by Foolbox [55], and CW_∞ [11]. We applied white-box attacks, with common attack budgets of 12 for ℓ_1-PGD, 0.5 for ℓ_2-PGD, 0.02 overshoot for DeepFool, and $8/255$ for CW_∞. Results are visualized in Figure 6. Our method significantly improves results (except for CW_∞) , even on unforeseen adversaries. *DBAT improves ℓ_2-PGD by up to 14%, ℓ_1-PGD by up to 20%, ℓ_2-DeepFool by up to 10%, and ℓ_∞-DeepFool by up to 16%.*

Feature Adversaries. We demonstrate the effectiveness of DBAT to so-called "adaptive adversaries": those that try to circumvent our defense by ignoring the projection function during the optimization process [71], and show that they cannot evade our defense. To do so, we tested DBAT on CIFAR-10 against feature and logit-level adversaries: Kullback–Leibler divergence (KLD) attack on the probabilities vectors [96], l_2 logit-matching attack [71] on adversarial examples and their corresponding natural examples, and lastly, a feature adversary suggested in [60]. We used $\varepsilon = 8/255$, $\delta = 1/255$, and ran for 500 iterations. Results are presented in Table 2. DBAT presents notably impressive and strong results against feature and logit-level adversaries. We also tried attacking the inner layers (and combinations of layers), in addition to attacking the feature representation layer, but we noticed it did not improve the attack success rate. These results also support our claim in §5.1, that an adaptive attacker will benefit from using the projection function in the attack optimization.

Table 2: CIFAR-10 results against feature and logit level adversaries.

Adversary	Robust Accuracy
KLD	85.9
l_2 Logit Matching	84.5
Feature Adversary [60]	86.8

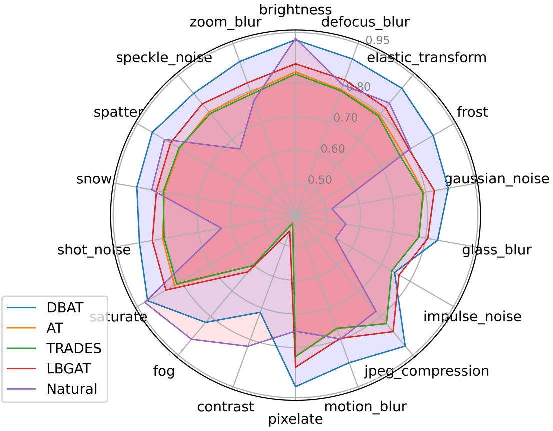


Figure 7: CIFAR-10C accuracy comparison results between different methods over all 18 natural corruption types, including noise, blur, weather, and digital categories.

5.4 Natural Corruptions Robustness

We demonstrate the effectiveness of DBAT when facing natural corruptions, as proposed by [35]. This corruptions benchmark dataset consists of 18 diverse corruption types. It covers noise, blur, weather, and digital categories. As the researchers claimed, research that improves performance on this benchmark should indicate general robustness gains, as the corruptions are varied and great in number. These corruptions each have five different levels of severity. To test DBAT, we use the CIFAR-10-C and CIFAR-100C corruptions benchmarks. Note that the corruptions are model-independent. As demonstrated in Figures 7 and 8, our method outperforms the other methods by a significant margin on all corruption types.

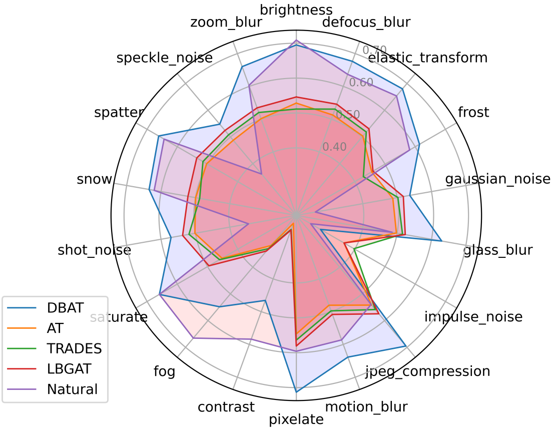


Figure 8: CIFAR-100C accuracy comparison results between different methods over all 18 natural corruption types, including noise, blur, weather, and digital categories.

CIFAR-10C results. When compared to the second-best performing method, *DBAT obtains an average improvement of 7.96% across all corruption types, and a maximum improvement of up to 35.19%.*

CIFAR-100C results. When compared to the second-best performing method, *DBAT obtains an average improvement of 10.82% across all corruption types, and a maximum improvement of up to 25.75%.*

5.5 Clean vs. Robust Accuracy Trade-off

Clean and robust accuracy trade-off. Originally, the clean and adversarial classes were equally weighted during training. Meaning, given that λ is the weighting factor for the adversarial classes, we set $\lambda = 1$ in our experiments.

In the following experiment, we run an extensive evaluation to show how the trade-off between natural and robust accuracy changes as we weigh the loss on the natural and adversarial classes differently, i.e., how the natural and robust accuracy changes as we change the values of λ. We use CIFAR-10 with the same experiment settings described above. We report Auto-Attack (AA) results, as well as natural accuracy results. In Figure 9 and Table 3, we plot DBAT's and TRADES's trade-off between natural and Auto-Attack robust accuracy as the weighting factor, λ, varies the trade-off between the natural and adversarial classes. For DBAT, we compare against the fully adaptive white-box perfect knowledge adversary (i.e., "Inference real-time access"). Not surprisingly, as we increase λ, clean accuracy decreases while robust accuracy increases, and vice-versa. However, as can be seen, the changes in natural accuracy for DBAT are relatively small, even though λ

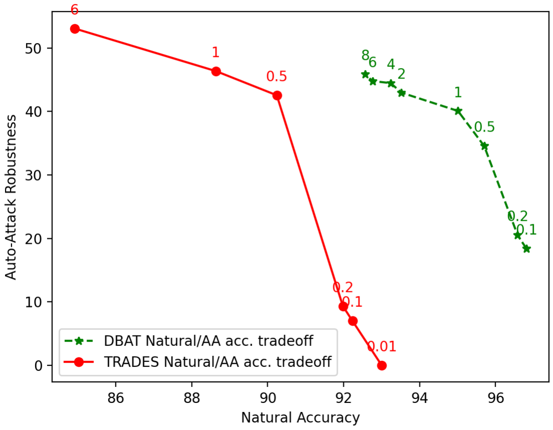


Figure 9: Natural and AutoAttack robust accuracy trade-off, for DBAT and TRADES on CIFAR-10, as we vary the hyper-parameter λ that controls the weight we put on the natural and adversarial classes. The numbers on the graph represent the value of λ for the specific trade-off.

was changed between a wide range of 0.1 and 8.

Table 3: DBAT's natural and Auto-Attack accuracy trade-off on CIFAR-10 as the weighting factor λ varies between 0.1 and 8 (where 1 is the default value used in the experiments).

λ	Natural	Auto-Attack
0.1	**96.81**	18.40
0.2	96.58	20.50
0.5	95.70	34.56
1	95.01	40.08
2	93.52	42.97
4	93.24	44.47
6	92.76	44.80
8	92.56	**45.91**

We also noticed that when increasing λ beyond a value of 10, the model started to diverge. We attribute this behavior to the fact that we are over-weighting dynamically newly generated classes, which in turn affects the model's ability to converge. We noticed that a simple warm start of λ can help to some extent. As for TRADES, we've noticed that as we decrease λ below 0.1, TRADES was not able to learn robust models.

Overall, we've demonstrated how reducing the trade-off parameter λ, TRADES was not able to match DBAT's clean accuracy without losing robust accuracy almost entirely. *This is another empirical evidence of DBAT's unique ability to learn models with optimal or near-optimal natural accuracy and a significant level of robustness.*

F1-robust. To further demonstrate that DBAT's robust-natural trade-off is indeed good compared to other methods, we adopt the recently proposed metric, F1-robust, suggested by [44] which was specifically designed as a balanced measurement for robust and natural accuracy. Results are presented in Table 4.

Table 4: Results on CIFAR-10 using the recently proposed F1-robust metric specifically designed as a balanced measurement for robust and natural accuracy. Higher is better.

Defense	F1-Robust
DBAT	**0.710**
AT	0.657
TRADES	0.659
LBGAT	0.670
Generalist	0.685
HAT	0.622
UIAT	0.645

5.6 Generalization Across Datasets

To check the generalization of our approach to different datasets, we evaluate DBAT on SVHN and CIFAR-100. We present Auto-Attack results in Table 5. As presented in Table 5, our method reaches optimal (for SVHN) and near-optimal (for CIFAR-100) natural accuracy on the different datasets while still maintaining significant robustness even against Auto-Attack.

SVHN evaluation. For SVHN, DBAT achieves an improvement of 3-5% under black-box attacks, When tested against white-box PGD attacks, DBAT achieves significant PGD robustness of 53.40% (PGD_{20}), similar to other popular AT methods. Additionally, DBAT reaches a natural accuracy of 96.86%, compared to 96.85% for a naturally trained model under the same architecture and settings. Meaning, ***DBAT exhibits no reduction in natural accuracy***, *while also achieving significant robust accuracy under various strong adversaries.*

CIFAR-100 evaluation. For CIFAR-100, DBAT achieves an improvement of 6-12% under black-box attacks. When tested in PGD white-box attacks, DBAT still achieves significant PGD robustness, e.g., 29.95% with PGD_{20}, similar to other popular AT methods. We note that Auto-Attack robustness is lower than the other methods, possibly due to the greater diversity in the dataset and the small number of examples in each class, which makes it more difficult to learn new adversarial class boundaries. DBAT achieves significant Auto-Attack robustness . Moreover, DBAT reaches a natural accuracy of 75.18%, compared to 79.30% for a naturally

Table 5: Natural, PGD , and Auto-Attack (AA) robust accuracy against the fully adaptive white-box perfect knowledge adversary (i.e., "Inference real-time access"). The attack is an ℓ_∞-PGD attack on SVHN and CIFAR-100. Similar to our method, the presented results are for models that **do not utilize additional data** during training. The Natural method refers to a model trained using standard training under the same hyper-parameters settings. Green represents the best natural accuracy among the robust models. Orange represents the second-best natural accuracy among the robust models. The range alongside the green arrow represents the natural accuracy improvement compared to the other methods.

Dataset	Method	Natural Acc.	PGD	AA
CIFAR-100	DBAT	**75.18** (↑12.2–18.5%)	27.22	18.17
	AT	56.73	28.45	24.12
	TRADES	58.24	29.70	24.90
	LBGAT	60.64	34.84	29.33
	Generalist	62.97	29.49	23.96
	HAT	58.73	27.92	23.34
	UIAT	59.55	30.81	25.73
	CAT	62.84	-	16.82
	Natural	79.30	0	0
SVHN	DBAT	**96.86** (↑2.8–6.8%)	49.31	40.49
	AT	89.90	49.45	45.25
	TRADES	90.35	54.13	49.50
	LBGAT	91.80	63.38	40.83
	Generalist	94.11	55.29	45.41
	HAT	92.06	57.35	52.06
	UIAT	93.28	58.18	52.45
	Natural	96.85	0	0

trained model under the same settings. *Compared to the other methods, DBAT improves natural accuracy by 14.5-18.5%.*

5.7 Ablation Studies

DBAT core components. We demonstrate the performance gain obtained by our method by removing the two parts that are not at the core of DBAT – SWA and Cutout. We use the CIFAR-10 dataset, with WRN-34-10, and report the Auto-Attack (AA) results when removing SWA and Cutout. When removing SWA and Cutout we observe that their total contribution to DBAT is 2.21% in natural accuracy and 3.55% in robust accuracy. Additionally, in Table 7 we present the results using different aggregation functions (sum and mean). Another study that tests the effect of training with targeted versus untargeted PGD is presented in Appendix B. Altogether, we conclude that the majority of the gain in natural and robust accuracy is obtained by DBAT.

Numerical instability. The loss function calculates the log over the max on the Softmax probabilities. This additional log may cause numerical instability, as discussed in Appendix G.2 of [71]. To demonstrate that our method does not suffer from numerical instability, we conducted two additional experiments:

- We replaced the max with LogSumExp (LSE) which should be more stable. Changes in results were within a standard deviation of ∓ 0.22 from the original reported results.
- We ran both AA and PGD-20 with 5 random restarts (within epsilon) and calculated mean and std. All results were within a standard deviation of ∓ 0.2.

Model complexity and training time overhead. We acknowledge the fact that DBAT presents additional complexity to the model. However, keeping in mind that only the output of the last final fully connected layer is doubled, the additional model complexity is minor in most of the cases. Specifically, with WRN-32-10 on CIFAR-100, DBAT introduces 64k additional parameters, which sums up to an additional 0.13% of the total parameters. Additionally, we also analyzed the training time overhead of our approach compared to the well-known TRADES. Overall, DBAT has a minor overhead of up to 2-3% for models with class numbers ranging from 10 to 100.

5.8 Adaptive vs. Non-adaptive Attacks

In §5.1 we stated that the most powerful white-box adversary is one who has access to both model parameters and to the projection function at any given time during inference. This adversary, which we used throughout the paper and termed *inference real-time access* adversary, possesses all possible capabilities and may also be thought of as a “Perfect-Knowledge Adversary”. That is, the adversary has access to the model parameters, and more importantly, to the projection function the defender is using at each given time during inference — and thus can utilize the same projection function while attacking. Throughout the paper, we compare primarily against this unrealistically powerful adaptive adversary. Although this adversary is not the most realistic one, here we wish to demonstrate that this kind of adversary is indeed the most powerful adversary. To do so, we compare the results with two different white-box adversaries. The first adversary will be referred to as the *model parameters* adversary. It is assumed to possess access to the model parameters, but not to the inference projection function — meaning that the adversary can optimize the network parameters, but is not utilizing the inference projection function (e.g., max) in the attack optimization process. The *model parameters* adversary illustrates how the attack success rate is influenced by the attacker’s adaptive knowledge (or lack thereof) about the defender’s projection function. The second, most realistic adversary, termed *model parameters + projection function access* adversary, knows that the defender is utilizing a projection function, but does

Table 6: Natural and Auto-Attack (AA) ℓ_∞ robustness on CIFAR-10, CIFAR-100 and SVHN, against adversaries with different capabilities. *Model parameters + projection function access* refers to the adversary capabilities to access both model parameters and inference projection function. *inference real-time access* refers to the adversary that can access not only to the entire network parameters but also to the defender's system and projection function at any given time during inference. *model parameters* refers to an adversary that gains access only to the model parameters, but not to the inference time projection function.

Dataset	Adversary access capabilities	Natural Acc.	Robust Acc.
CIFAR-10	Model parameters access	95.01	**50.31**
	Model parameters + projection function access		47.82
	Inference real-time access (model params. + inference-time access to projection func.)		40.08
CIFAR-100	Model parameters access	75.18	**23.16**
	Model parameters + projection function access		20.87
	Inference real-time access (model params. + inference-time access to projection func.)		18.17
SVHN	Model parameters access	96.86	**56.60**
	Model parameters + projection function access		48.58
	Inference real-time access (model params. + inference-time access to projection func.)		40.49

Table 7: Natural and Auto-Attack (AA) results against the strongest, *inference real-time access* adversary, using two additional aggregation functions: sum and mean.

Agg. function	Acc.	CIFAR-10	SVHN	CIFAR-100
Sum	Natural	95.01	96.85	75.07
	AA	39.81	36.00	18.08
Mean	Natural	95.01	96.85	75.07
	AA	39.93	35.84	18.49

not have inference-time access to the defender's choices at any given time during inference, and therefore has to conjecture the projection function from common projection functions (mean, softmax, maximum, sum, etc.) while attacking. Both adversaries — *model parameters* and *model parameters + projection function access* — are more realistic than the *inference real-time access* adversary, since the projection function is not part of the optimization. For example, the trained model can be published at model zoos, while the projection function does not. Alternatively, the defender can randomize or change the projection function at any time during inference.

Table 6 presents results under the three adversaries' settings. As can be seen, since the *model parameters* adversary does not utilize the knowledge about the projection function, the attack against DBAT becomes much less effective, and as a consequence the attack success rate decreases, i.e., model robustness increases.

6 Conclusion

In this paper, we demonstrate the advantage of treating the clean and adversarially perturbed examples as belonging to separate classes, instead of insisting that the classifier "stretch" a single class to accommodate them both. With this new idea in mind, we proposed Double Boundary Adversarial Training (DBAT). Our extensive evaluation illustrates the ability of DBAT to achieve state-of-the-art results under various tasks such as black-box PGD attacks, natural corruptions robustness, and unforeseen adversaries (e.g., ℓ_2-PGD, ℓ_1-PGD, and DeepFool). That said, our aim is not to compete with the state-of-the-art in robustness across the board. Rather, we wish to equip models with a significant level of robustness, while only incurring a minor or negligible degradation to their original natural accuracy. Therefore, the main benefit of DBAT is its ability to reach optimal or near-optimal natural accuracy while achieving significant robustness, even against strong adversaries. This ability makes DBAT applicable for real-world applications (e.g., healthcare, autonomous vehicles, and security systems) that cannot sacrifice much of their natural accuracy.

7 Ethical Considerations

The existence of adversarial examples points to a basic weakness of deep neural networks. With the deployment of AI in safety-critical systems, such as security systems, medical diagnosis, and autonomous driving, it is at a premium to build systems that are robust, at least to some extent, against such attacks. However, the gain of robustness is usually at the expense of the systems' natural accuracy. In order for real-world, safety-critical systems to adopt robust models, we need to make sure that the degradation in the natural accuracy is minor. For this reason, we suggested DBAT, which achieved a significant level of robustness without sacrificing much of the natural accuracy, and hope that it will help real-world applications adopt robust models.

Having said that, DBAT still has its limitations: adversarial training is an expensive training method that requires extra computations when compared to vanilla training. Moreover, DBAT achieves significant level of robustness, but it does not eliminate it completely, and one should take it under consideration when deploying such methods.

References

[1] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision*, pages 484–501. Springer, 2020.

[2] Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. *arXiv preprint arXiv:2007.02617*, 2020.

[3] Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, Cambridge, 1999.

[4] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283. PMLR, 2018.

[5] Idan Attias, Aryeh Kontorovich, and Yishay Mansour. Improved generalization bounds for robust learning. In *Algorithmic Learning Theory*, pages 162–183. PMLR, 2019.

[6] Idan Attias, Aryeh Kontorovich, and Yishay Mansour. Improved generalization bounds for robust learning, 2021.

[7] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent advances in adversarial training for adversarial robustness. *arXiv preprint arXiv:2102.01356*, 2021.

[8] Peter L. Bartlett, Dylan J. Foster, and Matus Telgarsky. Spectrally-normalized margin bounds for neural networks. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6240–6249, 2017.

[9] Qi-Zhi Cai, Min Du, Chang Liu, and Dawn Song. Curriculum adversarial training. *arXiv preprint arXiv:1805.04807*, 2018.

[10] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14, 2017.

[11] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[12] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, Percy Liang, and John C Duchi. Unlabeled data improves adversarial robustness. *arXiv preprint arXiv:1905.13736*, 2019.

[13] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 699–708, 2020.

[14] Tianlong Chen, Zhenyu Zhang, Sijia Liu, Shiyu Chang, and Zhangyang Wang. Robust overfitting may be mitigated by properly learned smoothening. In *International Conference on Learning Representations*, 2020.

[15] Minhao Cheng, Qi Lei, Pin-Yu Chen, Inderjit Dhillon, and Cho-Jui Hsieh. Cat: Customized adversarial training for improved robustness. *arXiv preprint arXiv:2002.06789*, 2020.

[16] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

[17] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pages 2196–2205. PMLR, 2020.

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

[19] Mónika Csikós, Nabil H. Mustafa, and Andrey Kupavskii. Tight lower bounds on the vc-dimension of geometric set systems. *J. Mach. Learn. Res.*, 20:81:1–81:8, 2019.

[20] Jiequan Cui, Shu Liu, Liwei Wang, and Jiaya Jia. Learnable boundary guided adversarial training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15721–15730, 2021.

[21] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[22] Luc Devroye and Gábor Lugosi. *Combinatorial methods in density estimation*. Springer series in statistics. Springer, 2001.

[23] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. *arXiv preprint arXiv:1812.02637*, 2018.

[24] Junhao Dong, Seyed-Mohsen Moosavi-Dezfooli, Jianhuang Lai, and Xiaohua Xie. The enemy of my enemy is my friend: Exploring inverse adversaries for improving adversarial training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24678–24687, 2023.

[25] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

[26] Dylan J. Foster and Alexander Rakhlin. ℓ_∞ vector contraction for rademacher complexity. *CoRR*, abs/1911.06468, 2019.

[27] Dylan J Foster and Alexander Rakhlin. ℓ_∞vector contraction for rademacher complexity. *arXiv preprint arXiv:1911.06468*, 2019.

[28] Micah Goldblum, Liam Fowl, Soheil Feizi, and Tom Goldstein. Adversarially robust distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3996–4003, 2020.

[29] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[30] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

[31] Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020.

[32] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. Improving robustness using generated data. *Advances in Neural Information Processing Systems*, 34, 2021.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[34] Warren He, Bo Li, and Dawn Song. Decision boundary analysis of adversarial examples. In *International Conference on Learning Representations*, 2018.

[35] Dan Hendrycks and Thomas G Dietterich. Benchmarking neural network robustness to common corruptions and surface variations. *arXiv preprint arXiv:1807.01697*, 2018.

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

[37] Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. Robust pre-training by adversarial contrastive learning. In *NeurIPS*, 2020.

[38] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018.

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

[40] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[41] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2016.

[42] Saehyung Lee, Hyungyu Lee, and Sungroh Yoon. Adversarial vertex mixup: Toward better adversarially robust generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 272–281, 2020.

[43] Saehyung Lee, Changhwa Park, Hyungyu Lee, Jihun Yi, Jonghyun Lee, and Sungroh Yoon. Removing undesirable feature contributions using out-of-distribution data. *arXiv preprint arXiv:2101.06639*, 2021.

[44] Matan Levi, Idan Attias, and Aryeh Kontorovich. Domain invariant adversarial learning. *arXiv preprint arXiv:2104.00322*, 2021.

[45] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[46] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

[47] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[48] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[49] Tianyu Pang, Min Lin, Xiao Yang, Jun Zhu, and Shuicheng Yan. Robustness and accuracy could be reconcilable by (proper) definition. In *International Conference on Machine Learning*, pages 17258–17277. PMLR, 2022.

[50] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning*, pages 4970–4979. PMLR, 2019.

[51] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[52] Rahul Rade and Seyed-Mohsen Moosavi-Dezfooli. Reducing excessive margin to achieve a better accuracy vs. robustness trade-off. In *International Conference on Learning Representations*, 2021.

[53] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

[54] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Semidefinite relaxations for certifying robustness to adversarial examples. *arXiv preprint arXiv:1811.01057*, 2018.

[55] Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. In *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017.

[56] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan A Calian, Florian Stimberg, Olivia Wiles, and Timothy Mann. Fixing data augmentation to improve adversarial robustness. *arXiv preprint arXiv:2103.01946*, 2021.

[57] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan Andrei Calian, Florian Stimberg, Olivia Wiles, and Timothy A Mann. Data augmentation can improve robustness. *Advances in Neural Information Processing Systems*, 34, 2021.

[58] Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pages 8093–8104. PMLR, 2020.

[59] Jérôme Rony, Luiz G Hafemann, Luiz S Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger. Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4322–4330, 2019.

[60] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. *arXiv preprint arXiv:1511.05122*, 2015.

[61] Vikash Sehwag, Saeed Mahloujifar, Tinashe Handina, Sihui Dai, Chong Xiang, Mung Chiang, and Prateek Mittal. Robust learning meets generative models: Can proxy distributions improve adversarial robustness? *arXiv preprint arXiv:2104.09425*, 2021.

[62] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

[63] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifiable distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2, 2017.

[64] Chawin Sitawarin, Supriyo Chakraborty, and David Wagner. Sat: Improving adversarial training via curriculum-based loss smoothing. In *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, pages 25–36, 2021.

[65] Chawin Sitawarin, Arvind Sridhar, and David Wagner. Improving the accuracy-robustness trade-off for dual-domain adversarial training. *UDL*, 2021.

[66] Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Improving the generalization of adversarial training with domain adaptation. *arXiv preprint arXiv:1810.00740*, 2018.

[67] Gaurang Sriramanan, Sravanti Addepalli, Arya Baburaj, and Venkatesh Babu Radhakrishnan. Towards efficient and effective adversarial training. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[68] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[69] Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 426–433. IEEE, 2016.

[70] Florian Tramer. Detecting adversarial examples is (nearly) as hard as classifying them. *arXiv preprint arXiv:2107.11630*, 2021.

[71] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *Advances in neural information processing systems*, 33:1633–1645, 2020.

[72] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[73] Yu-Lin Tsai, Chia-Yi Hsu, Chia-Mu Yu, and Pin-Yu Chen. Formalizing generalization and robustness of neural networks to weight perturbations. *arXiv preprint arXiv:2103.02200*, 2021.

[74] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[75] Jonathan Uesato, Jean-Baptiste Alayrac, Po-Sen Huang, Robert Stanforth, Alhussein Fawzi, and Pushmeet Kohli. Are labels required for improving adversarial robustness? *arXiv preprint arXiv:1905.13725*, 2019.

[76] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[77] Haotao Wang, Aston Zhang, Shuai Zheng, Xingjian Shi, Mu Li, and Zhangyang Wang. Removing batch normalization boosts adversarial training. In *International Conference on Machine Learning*, pages 23433–23445. PMLR, 2022.

[78] Hongjun Wang and Yisen Wang. Generalist: Decoupling natural and robust generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20554–20563, 2023.

[79] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *ICML*, volume 1, page 2, 2019.

[80] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*, 2019.

[81] Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. Better diffusion models further improve adversarial training. *arXiv preprint arXiv:2302.04638*, 2023.

[82] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295. PMLR, 2018.

[83] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

[84] Eric Wong, Frank R Schmidt, Jan Hendrik Metzen, and J Zico Kolter. Scaling provable adversarial defenses. *arXiv preprint arXiv:1805.12514*, 2018.

[85] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33, 2020.

[86] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 501–509, 2019.

[87] Cihang Xie and Alan Yuille. Intriguing properties of adversarial training at scale. *arXiv preprint arXiv:1906.03787*, 2019.

[88] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2730–2739, 2019.

[89] Yuancheng Xu, Yanchao Sun, Micah Goldblum, Tom Goldstein, and Furong Huang. Exploring and exploiting decision boundary dynamics for adversarial robustness. *arXiv preprint arXiv:2302.03015*, 2023.

[90] Huanrui Yang, Jingyang Zhang, Hongliang Dong, Nathan Inkawhich, Andrew Gardner, Andrew Touchet, Wesley Wilkes, Heath Berry, and Hai Li. Dverge: diversifying vulnerabilities for enhanced robust generation of ensembles. *arXiv preprint arXiv:2009.14720*, 2020.

[91] Dong Yin, Ramchandran Kannan, and Peter Bartlett. Rademacher complexity for adversarially robust generalization. In *International Conference on Machine Learning*, pages 7085–7094. PMLR, 2019.

[92] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[93] Runtian Zhai, Tianle Cai, Di He, Chen Dan, Kun He, John Hopcroft, and Liwei Wang. Adversarially robust generalization just requires more unlabeled data. *arXiv preprint arXiv:1906.00555*, 2019.

[94] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. *arXiv preprint arXiv:1905.00877*, 2019.

[95] Haichao Zhang and Jianyu Wang. Defense against adversarial attacks using feature scattering-based adversarial training. *Advances in Neural Information Processing Systems*, 32:1831–1841, 2019.

[96] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482. PMLR, 2019.

[97] Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In *International Conference on Machine Learning*, pages 11278–11287. PMLR, 2020.

A Full Experimental Setup Details

We conduct our experiments on CIFAR10, CIFAR100 [39] and SVHN [48]. We use the wide residual network (WRN-34-10) [92] architecture for CIFAR-10 and CIFAR-100, and the PreAct ResNet-18 [33] for SVHN. Following [58], we apply early-stopping based on a validation set. The batch size is set to 128, weight decay is set to $7e^{-4}$. We train the model for 200 epochs with an initial learning rate of 0.1. For CIFAR datasets, the learning rate is decayed by a factor of 10 at iterations 50 and 150. For SVHN, the learning rate is decayed by a factor of 10 at iterations 50 and 75. Natural images are padded with 4-pixel padding with 32-random crop and random horizontal flip. Furthermore, all methods are trained using SGD with momentum 0.9. We combine Stochastic Weight Averaging (SWA) [36], exponential moving average on the model weights during training steps, within our training process. SWA was shown to be effective in training robust models [14,57], due to its temporal ensemble effect, and the ability to smooth the weights.

B Training with Targeted vs. Untargeted PGD

As previously stated, using targeted-PGD on random target labels helps to better generalize the adversarial classes. In Table 8 , we show the comparison of the results between random targeted PGD (T-PGD), Least-Likely (least likely label based on the model decision) targeted PGD, and untargeted PGD. Using Least-Likely T-PGD achieves optimal natural accuracy, at the cost of a decrease in robust accuracy. Using untargeted PGD, we experience better AA robustness, at the cost of a decrease in natural accuracy.

Table 8: Natural accuracy and Auto-Attack robust accuracy against the strongest adversary, *Inference real-time access*, on CIFAR-10.

Attack type	Natural Acc.	AA
Random T-PGD	95.01	40.08
Least-Likely T-PGD	**95.67**	34.42
Untargeted PGD	92.18	**44.13**

Finally, we conducted a case study experiment on CIFAR-10 for two other training methods, AT [45] and TRADES [96], in order to test if random targeted PGD can help improve other methods' results as much as it helped our method. For AT, Auto-Attack robust accuracy was 45.94%, a decrease of 5.58% in robustness, with a small improvement of almost 3% in natural accuracy. For TRADES, Auto-Attack robust accuracy was 52.32%, a decrease of 0.76% in robustness, with a small improvement of 1.07% in natural accuracy. We can conclude that random targeted PGD degrades robustness for the other tested methods with only marginal improvement in natural accuracy.

C Rademacher Analysis

Rademacher complexity. As mentioned in Section 3.4, the VC-dimension is impractical in analyzing deep neural networks with a large number of weights. We will now argue that the thrust of our point continues to hold for the *Rademacher complexity* as well, which is far more practical as far as providing finite-sample generalization bounds [8,91]. We assume a basic familiarity with this notion and refer the reader to [46] for background. For a brief recap, if F is a collection of functions mapping some set Ω to $\mathbb{R}$, and $X_1,\ldots,X_n$ is sampled iid from some distribution on Ω, then the (empirical) Rademacher complexity is defined by

$$R_n(F;X_1,\ldots,X_n) = \mathop{\mathbb{E}}_{\sigma} \sup_{f\in F} \frac{1}{n}\sum_{i=1}^{n} \sigma_i f(X_i), \qquad (6)$$

where expectation is over the σ_i, which are iid *Rademacher* variables (i.e., $\mathbb{P}(\sigma_i = 1) = \mathbb{P}(\sigma_i = -1) = 1/2$). It is a classic fact [46, Theorem 3.5] that the Rademacher complexity upper-bounds the generalization error: if $X_1,\ldots,X_n$ is an iid sample, then

$$\mathrm{err}(h) \leq \widehat{\mathrm{err}}(h) + R_n(H; X_1, \ldots, X_n) + 3\sqrt{\frac{\log(2/\delta)}{2n}} \qquad (7)$$

holds uniformly over all $h \in H$ with probability[3] at least $1 - \delta$.

It is always the case [22, Theorem 3.2, Theorem 4.3] that

$$R_n(H; X_1, \ldots, X_n) \leq c\sqrt{\frac{V}{n}}, \qquad (8)$$

where V is the VC-dimension of H and c is a universal constant. Observe that the left-hand side of (8) is sensitive to the sampling distribution of X_i (and can be arbitrarily small for very concentrated distributions), while the right-hand side is distribution-free — and hence the bound in (8) can be rather loose. In situations where the VC-dimension is very large, the more delicate Rademacher analysis gives much tighter bounds than VC theory [8, 91].

We will make use of a recent Rademacher ℓ_∞ vector contraction result:

Theorem 1. *[27] Let F be a $\mathbb{R}^\ell$-valued function class, such that the coordinate projection class is denoted by $F_j = \{w \mapsto f(w)_j \,|\, f \in F\}$, for $1 \leq j \leq \ell$. Let $(\phi_t)_{t \leq n}$ be a sequence of functions such that each ϕ_t is L-Lipschitz with respect to ℓ_∞ norm. For any $\alpha > 0$, there exists a constant $C_\alpha > 0$ such that if $|\phi_t(f(w))| \vee ||f(w)||_\infty \leq B$, then it holds for any sequence $\mathbf{w} = (w_1, \cdots, w_n)$,*

$$\begin{aligned} R_n(\phi \circ F|\mathbf{w}) &:= E_\sigma \sup_{f \in F} \frac{1}{n} \sum_{t=1}^{n} \sigma_t \phi_t(f_t(w_t)) \\ &\leq C_\alpha L\sqrt{\ell} \cdot \max_{i \in [\ell]} \sup_{\mathbf{a}=(a_1,\ldots,a_n)} R_n(F_i|\mathbf{a}) \cdot \\ &\quad \log^{\frac{3}{2}+\alpha}\left(\frac{Bn}{\max_{i \in [\ell]} \sup_{\mathbf{a}=(a_1,\ldots,a_n)} R_n(F_i|\mathbf{a})}\right). \end{aligned}$$

We observe, as in [6, Theorem 5.1], that $\max(x_1, \ldots, x_\ell)$ is a 1-Lipschitz function with respect to the ℓ_∞ norm. It follows (taking $\alpha = 1/2$ and specializing the argument of [6, Theorem 5.1] to our simpler case), that if $H^{\cup \ell}$ is the *ℓ-fold union* of H — that is, every $h' \in H^{\cup \ell}$ can be expressed as the union of some ℓ members of H — then we have:

$$R_n(H^{\cup \ell}; X_1, \ldots, X_n) \leq C\sqrt{\ell} \max_{i \in [\ell]} \bar{R}_n(H) \log^2 \frac{n}{\bar{R}_n(H)}, \qquad (9)$$

where C is a universal constant and $\bar{R}_n(H) := \sup_{(x_1,\ldots,x_n) \in \Omega^n} R_n(H; x_1, \ldots, x_n)$. Since the Rademacher complexity of the ℓ-fold union grows roughly as $\sqrt{\ell}$, it may often be advantageous to re-analyze complex hypotheses as unions of simple ones — just as we concluded for the VC-dimension in §4. Thus, we can apply (7) to the two competing approaches: (i) when learning ℓk "simple" classifiers from the class H, and (ii) when learning k "complex" classifiers from the class $H^{\cup \ell}$. Ignoring logarithmic factors, and treating δ as fixed, we can compare the bound of order roughly $R_n(H) + \sqrt{\log(\ell k)/n}$ when splitting the classifiers (the ℓk inside the log is from the union bound) and roughly $\sqrt{\ell} R_n(H) + \sqrt{\log(k)/n}$ without splitting (there are only k classes but each incurred a $\sqrt{\ell}$ factor from (9)). For simple hypothesis classes H (i.e., those with a low Rademacher complexity), this again demonstrates the advantage of splitting.

D Distance to Decision Boundary

In the following experiment, we wish to demonstrate that the conceptual illustration that we've drawn can exist in real datasets. To do so, we need to demonstrate that most clean examples are relatively close to the decision boundary, in a distance that is half of the perturbation size ($\epsilon/2 = 4/255$), so that it will support the way we've drawn the intuition in 1, where examples "switch" places after the attack. In the following experiment, we used SVHN with a naturally trained model using PreAct ResNet-18, and 1000 random examples from SVHN's test set. To calculate the distance, we followed the distance estimation suggested in [34]. We estimate the distance to a decision boundary in a sample of random directions in the model's input space, starting from a given input point. In each direction, we estimate the distance to a decision boundary by computing the model's prediction on perturbed inputs at points along the direction and increase the random directions by a magnitude factor (0.002) if the prediction does not change in any of the directions. We perform this search over a set of 1,000 random orthogonal directions. Results are present in Figure 10.

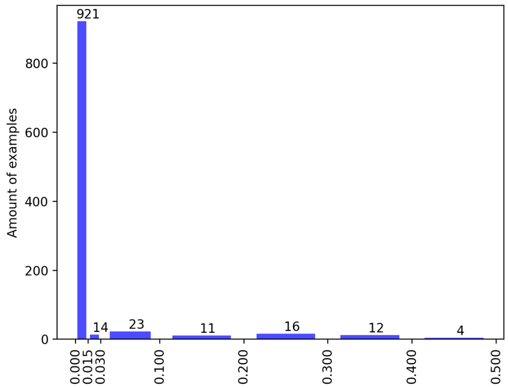


Figure 10: Histogram of the estimated distance from the decision boundary

[3] where the randomness is over the sample X_i