Appendices

A. Related works

Generalization and robustness. Robustness in machine learning models is a large field. We review some more works that analyze robustness from the statistical perspective. The majority of works that study adversarial robustness from the generalization perspective study the generalization behaviors of machine learning models under *adversarial risk*. The works that study adversarial risk include Attias et al. (2018); Schmidt et al. (2018); Cullina et al. (2018); Yin & Bartlett (2018); Khim & Loh (2018); Sinha et al. (2018). The bounds obtained under the setting of adversarial risk characterize the risk gap introduced by adversarial examples. Thus, it is intuitive that a larger risk gap would be obtained for a larger allowed perturbation limit ϵ, which is roughly among the conclusions obtained in those bounds. That is to say, the conclusion normally leads to a larger generalization error as an algorithm is asked to handle more adversarial examples, for that it focuses on characterizing the error of adversarial examples, not that of natural examples. However, adversarial risk is not our focus. In this paper, we study when a classifier needs to accommodate adversarial examples, what is the influence that the accommodation has on generalization behaviors of empirical risk of natural data.

Hard and soft adversarial robust regularization. We study the behaviors of NNs that are trained in the way that adversarial examples are required to be classified correctly. We note that the adversarial robustness required can also be built in NNs in a soft way by adding a penalty term in the risk function. Relevant works includes Lyu et al. (2015) and Miyato et al. (2018). This line of works is not our subject of investigation. They focus on increasing test performance instead of defense performance. The focus of our works is to study the behaviors that lead to standard performance degradation when a network is trained to has a reasonable defense ability to adversarial examples. For example, a 50% accuracy on adversarial examples generated by PGD methods (Madry et al., 2018) in fig. 12 is a defense ability that can serve as a baseline for a reasonable defense performance. It is natural that in the setting where the requirement to defend against adversarial examples is dropped, the regularization can be weakened (added as a penalty term) to *only* aim to improve the test performance of the network. In this case, no performance degradation would occur, but the defense performance is also poor.

Explicit regularization that increases robustness of NNs by imposing smoothness through a penalty term. The smoothing effect of adversarial training on the loss surface has been observed in contemporary works (Moosavi-Dezfooli et al., 2019; Qin et al., 2019). And based on such an observation, explicit regularization is formulated by adding a penalty term to the risk function to increase NNs' robustness. The message of this work is different from the insights of (Moosavi-Dezfooli et al., 2019; Qin et al., 2019) related to regularization. They (Moosavi-Dezfooli et al., 2019; Qin et al., 2019) show that if the output of NNs is explicitly smoothed through a penalty term thorough curvature regularization (Moosavi-Dezfooli et al., 2019), or local linearization (Qin et al., 2019), then a certain degree of adversarial robustness (AR) can be achieved. The penalty term works as a regularizer because it is explicitly formulated that way. It is not clear whether adversarial training, which is a different and arguably the most widely used technique, has the effect of a regularizer. This is the issue that is investigated in this work, and we show that adversarial training effectively regularizes NNs, which is not clear previously. In addition, this work has shown that adversarial training has a smoothing effect on features of all layers, instead of just the loss surface. Such a fine-grained analysis is possible because of the theoretical instruments developed in this work, and is absent previously.

B. Further empirical studies on adversarial robustness

B.1. Technique to build adversarial robustness

To begin with, we describe the technique that we use to build AR into NNs. As mentioned in the caption of fig. 1, we choose arguably the most well received technique, i.e., the adversarial training method (Madry et al., 2018). Specifically, we use l_∞-PGD (Madry et al., 2018) untargeted attack adversary, which creates an adversarial example by performing projected gradient descent starting from a random perturbation around a natural example. Then, NNs are trained with adversarial examples. NNs with different AR strength are obtained by training them with increasingly stronger adversarial examples. The adversarial strength of adversarial examples is measured in the l_∞ norm of the perturbation applied to examples. l_∞-norm is rescaled to the range of $0 - 255$ to present perturbations applied to different datasets in a comparable way; that means in fig. 1 fig. 3 fig. 4 fig. 5 and fig. 6 fig. 12, AR is measured in this scale. We use 10 steps of size 2/255 and maximum of = [4/255, 8/255, 16/255] respectively for different defensive strength in experiments. For example, a NN with AR strength 8 is a NN trained with adversarial examples generated by perturbations whose l_∞ norm are at most 8. Lastly, we note that although adversarial training could not precisely guarantee an adversarial robustness radius of ϵ, a larger l_∞ norm used in training would make NNs adversarially robust in a larger ball around examples. Thus, though the precise adversarial robustness radius is not known, we know that we are making NNs adversarially robust w.r.t. a larger ϵ. Consequently, it enables us to study the influence of ϵ-AR on NNs by studying NNs trained with increasing l_∞ norm.

B.2. Quantitative analysis of variance reduction in singular values

Here, we provide more quantitative analysis on fig. 5c and fig. 5d, as noted previously in section 4.2.2.

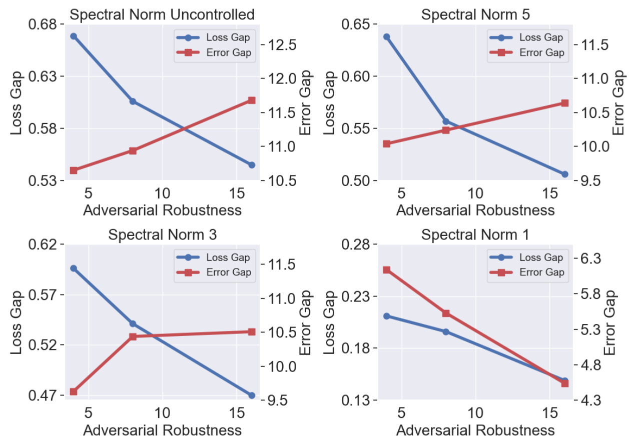
Quantitatively, we can look at the accumulated standard deviation (STD) difference in all layers. We separate the layers into two group: the group that the STD (denoted σ_i^4) of singular values of layer i (of the NN trained) with AR strength 4 that is larger than that (denoted σ_i^{16}) of AR strength 16; and the group that is smaller. In CIFAR10, for the first group, the summation of the difference/increments of STD of the two networks ($\sum_i \sigma_i^4 - \sigma_i^{16}$) is 4.7465, and the average is 0.1158. For the second groups, the summation ($\sum_i \sigma_i^{16} - \sigma_i^4$) is 0.4372, and the average is 0.0312. In CIFAR100, the summation of the first group is 3.7511, and the average is 0.09618; the summation of the second group is 0.4372, and the average is 0.1103. The quantitative comparison shows that the accumulated STD decrease in layers that have their singular value STDs decreased (comparing STD of the NN with AR strength 16 with STD of the NN with AR strength 4) is *a magnitude larger* the accumulated STD increase in the layers that have their singular value STDs increased. The magnitude difference is significant since the STDs of singular values of most layers are around 1.

B.3. Discrepancy between trends of loss and error rate gaps in large capacity NNs

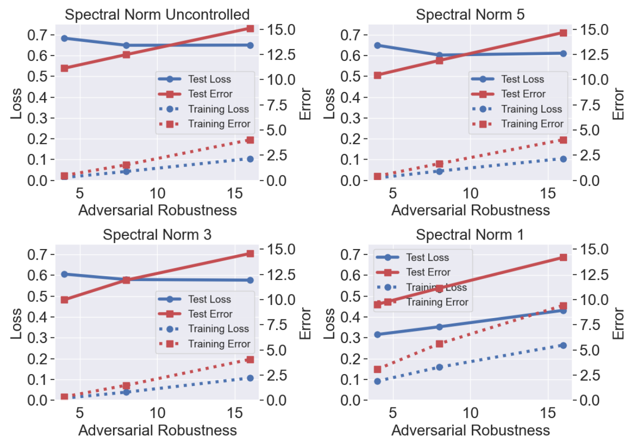
In section 4.1, we have noted an inconsistent behaviors of CIFAR10, compared with that of CIFAR100 and Tiny-ImageNet: the error gap reduces for CIFAR100 and Tiny-ImageNet, but increases for CIFAR10. It might suggest that AR does not effectively regularize NNs in the case of CIFAR10. However, we show in this section that the abnormal behaviors of CIFAR10 are derived from the same margin concentration phenomenon observed in section 4.2.1 due to capacity difference, and compared with the error gaps, the GE/loss gaps are more faithfully representatives of the generalization ability of the NNs. Thus, the seemingly abnormal phenomenon corroborate, not contradict, the *key results* present in section 1.

Using CIFAR10 and CIFAR100 as examples and evidence in the previous sections, we explain how the discrepancy emerges from AR's influence on margin distributions of the same network trained on tasks with different difficulties. Further evidence that the discrepancy arises from capacity difference would be shown at appendix B.3, where we run experiments to investigate GE gap of NNs with varied capacities on the same task/dataset.

1. *On CIFAR10, the margin distribution of training sets not only concentrate more around zero, but also skews towards zero.* As shown in the margin distribution on training sets of CIFAR10 in fig. 4c, we find that the large error gap is caused by the high training accuracy that is achieved with a high concentration of training samples just slightly beyond the decision boundary. This phenomenon does not happen in CIFAR100. Comparing margin distribution on the test set in fig. 4(a) in fig. 4a, the margin distribution on the training set in fig. 4c is highly skewed, i.e., asymmetrically distributed w.r.t. mean. While the margin distributions of CIFAR100 training set in fig. 4d is clearly less skewed, and looks much more like a normal distribution, as that of the margin distribution on the test set.
2. *The high skewness results from the fact that the NN trained on CIFAR10 is of large enough capacity to overfit the training set.* As known, CIFAR100 is a more difficult task w.r.t. CIFAR10 with more classes and less training examples in each class. Thus, relatively, even the same ResNet56 network is used, the capacity of the network trained on CIFAR10 is larger than the one trained on CIFAR100. Recall that NNs have a remarkable ability to overfit training samples (Zhang et al., 2016). And note that though AR requires in a ball around an example, the examples in the ball should be of the same class, since the ball is supposed only to include imperceptible perturbation to the example, few of the training samples are likely in the same ball. Thus, the ability to overfit the training set is not regularized by AR: if NNs can overfit all training samples, it can still overfit some more examples that are almost imperceptibly different. For CIFAR10, since NNs have enough capacity, the NN simply overfits the training set.
3. However, as shown in the observed overfitting phenomenon in fig. 4c, the high training accuracy is made up of correct predictions with relatively lower confidence (compared with NNs with lower AR), which is bad and not characterized by the error rate; and the low test accuracy are made up of wrong predictions with relatively lower confidence as well (as explain in section 4.2.1), which is good, and not characterized by error rate as well. *Thus, the error gap in this case does not characterize the generalization ability (measured in term of prediction confidence) of NNs well, while the GE gap more faithfully characterizes the generalization ability, and show that AR effectively regularizes NNs.* In the end, AR still leads to biased poorly performing solutions — since the overfitting in training set does not prevent the test margin distribution concentrating more around zero, which leads to higher test errors of CIFAR10 as shown in fig. 3b. It further suggests that the damage AR done to the hypothesis space is not recovered by increasing capacity, however the ability of NNs to fit arbitrary labels



(a) Gap Curves



(b) Error Rate & Loss Curves

Figure 6. The four plots from upper left to lower bottom (in each subfigure) are NNs with increasingly smaller spectral complexity, where "Spectral Norm 1" means for each weight matrix of the NN, its spectral norm is at most 1. **(a)** Plots of training/test loss gap (and error gap) against adversarial robustness strength. **(b)** Training/test losses and error rates against increased strength of adversarial robustness.

is not hampered by AR.

B.4. Further evidence of regularization effects on NNs with varied capacity

In previous sections, we observe AR consistently effectively regularizes NNs; meanwhile, we also observe that in the case where a NN has a large capacity, it can spuriously overfit training samples and lead to an increased error gap. In this section, we present additional results by applying AR to networks of varied capacities. The effects of adversarial training on a larger NNs, i.e., ResNet 110 is given in appendix B.4.1. Then, AR applied on NNs with controlled capacities through spectral normalization is given in appendix B.4.2. This is to ensure that our observations and analysis in previous sections exist not just at some singular points, but also in a continuous area in the hypothesis space.

B.4.1. Regularization effects on NNs with larger capacity

To preliminarily validate that the regularization effects observed in section 4.1 manifest in NNs with varied capacities, we investigate the regularization effects of AR on a larger NNs, i.e., ResNet 110. The results are shown in fig. 7. The observed phenomenon is the same with that of ResNet56 presented in section 4.1, and thus corroborates our results.

B.4.2. Regularization effects on NNs with controlled capacities

To control capacities of NNs quantitatively, we choose the measure based on spectral norm (Bartlett et al., 2017; Neyshabur et al., 2018a). In spectral norm based capacity measure bound (Bartlett et al., 2017; Neyshabur et al., 2018a), the NN capacity is normally proportional to a quantity called spectral complexity (SC), which is defined as follows.

Definition 7 (Spectral Complexity)**.** *Spectral complexity $SC(T)$ of a NN T is the multiplication of spectral norms of weight matrices of layers in a NN.*

$$SC(T) = \prod_{i=1}^{L} ||\boldsymbol{W}_i||_2$$

where $\{\boldsymbol{W}_i\}_{i=1\ldots L}$ denotes weight matrices of layers of the NN.

To control SC, we apply the spectral normalization (SN) (Sedghi et al., 2018) on NNs. The technique renormalizes the spectral norms of the weight matrices of a NN to a designated value after certain iterations. We carry out the normalization at the end of each epoch.

We train ResNet56 with increasingly strong AR and with increasingly strong spectral normalization. The results are shown in fig. 6.

As can be seen, as the capacity of NNs decreases (from upper left to bottom right in each sub-figure), the error gap between training and test gradually changes from an increasing trend to a decreasing trend, while the loss gap keeps a consistent decreasing trend. It suggests that the overfitting phenomenon is gradually prevented by another regularization techniques, i.e., the spectral normalization. As a result, the regularization effect of AR starts to emerge even in the error gap, which previously manifests only in the loss gap. The other curves corroborate our previous observations and analysis as well.

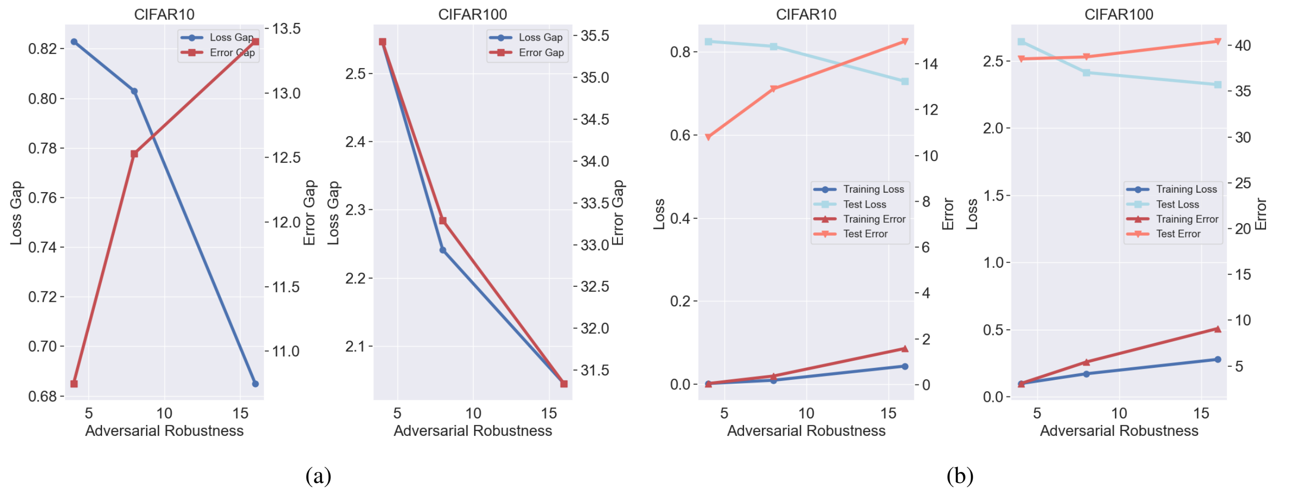


Figure 7. Experiment results on CIFAR10/100 with ResNet-110 (He et al., 2016). The unit of x-axis is the adversarial robustness (AR) strength of NNs, c.f. the beginning of section 4. **(a)** Plots of loss gap (and error rate gap) between training and test datasets v.s. AR strength. **(b)** Plots of losses (and error rates) on training and test datasets v.s. AR strength.

B.5. Further evidence on the smoothing effect of adversarial robustness

We quantitatively measure the smoothing effect around examples here by measuring the average maximal loss change/variation induced by the perturbation (of a fixed infinity norm) applied on examples. We found that the loss variation decreases as networks become increasingly adversarially robust. Note that the loss of an example is a proxy to the confidence of the example — it is the logarithm of the estimated probability (a characterization of confidence) of the NN classifier.

For a given maximal perturbation range characterized by the infinity norm, we generate adversarial examples within that norm for all test samples. For each example, the maximal loss variation/change of the adversarial example w.r.t. the natural example is computed for networks with different adversarial strength. To obtain statistical behaviors, we compute the average and standard deviation of such maxima of all test samples. The results are shown in fig. 8. The exact data can be found in table 1.

We can see that the average loss variation decreases with adversarial robustness. The standard deviation decreases with network adversarial robustness as well. The phenomenon that the standard deviation is comparably large with the mean might need some explanation. This is because different examples have different losses, thus the loss varies in relatively different regimens — the more wrongly classified examples vary in a larger magnitude, and vice versa for more correctly classified examples. This phenomenon leads to the large standard deviation of the loss variation.

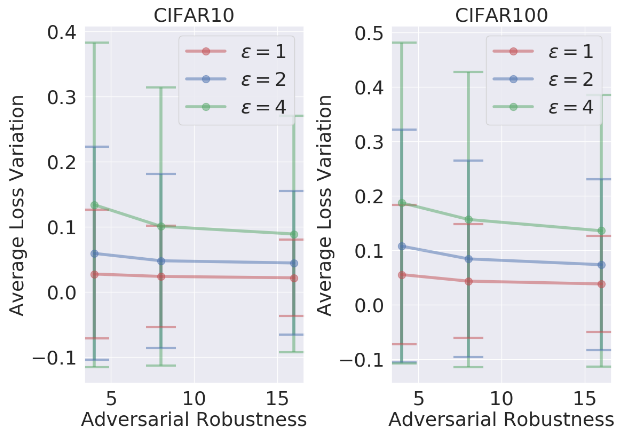


Figure 8. Average maximal loss variation induced by adversarial examples in networks with increasing adversarial robustness. The experiments are carried on CIFAR10/100. ϵ represents the maximal perturbation can be applied on natural test examples to generate adversarial examples. It is measured in the infinity norm. The larger the ϵ, the stronger the perturbation is. The error bars represent standard deviation.

B.6. Further experiments on using FGSM in adversarial training to build adversarial robustness

We explain the choice of PGD as the representative of adversarial training techniques here. Various adversarial training methods are variant algorithms that compute first order approximation to the point around the input example that minimizes the label class confidence. The difference is how close the approximation is. Recent works on adversarial examples exclusively only use PGD in experiments

Table 1. Data of the smoothing effect of PGD adversarial training in fig. 8.

Dataset	Attack Strength	Defensive Strength 4	Defensive Strength 8	Defensive Strength 16
CIFAR10	$\epsilon = 1$	0.0273 ± 0.0989	0.0236 ± 0.0778	0.0215 ± 0.0588
	$\epsilon = 2$	0.0590 ± 0.1637	0.0477 ± 0.1337	0.0443 ± 0.1102
	$\epsilon = 4$	0.1337 ± 0.2494	0.1006 ± 0.2137	0.0888 ± 0.1816
CIFAR100	$\epsilon = 1$	0.0550 ± 0.1276	0.0430 ± 0.1043	0.0379 ± 0.0886
	$\epsilon = 2$	0.1072 ± 0.2138	0.0839 ± 0.1802	0.0732 ± 0.1568
	$\epsilon = 4$	0.1868 ± 0.2946	0.1563 ± 0.2712	0.1355 ± 0.2494

(Kannan et al., 2018; Schmidt et al., 2018; Xie et al., 2019; Ilyas et al., 2019; Wang & Zhang, 2019). It is also a very strong multi-step attack method that improves over many of its antecedents: NNs trained by FGSM could have no defense ability to adversarial examples generated by PGD, as shown in Table 5 in Madry et al. (2018); multi-step methods prevent the pitfalls of adversarial training with single-step methods that admit a degenerate global minimum (Tramèr et al., 2017). Thus, we believe the observations in this work is representative for various adversarial training techniques. Yet, even in the worst case, this work at least makes a first step to understand a representative approach of the approximation.

To corroborate the analysis, we also use FGSM (Goodfellow et al., 2015) in the adversarial training to build adversarial robustness into NNs. The results are consistent with the results obtained using PGD. The experiments are carried on CIFAR10/100. We present key plots that support the results obtained in the main con- tent here. All the setting are same with that described in appendix B.1 of PGD, except that we replace PGD with FGSM.

Adversarial robustness reduces generalization gap and standard test performance. In section 4.1, we find that NNs with stronger adversarial robustness tend to have smaller loss/generalization gap between training and test sets. Consistent phenomenon has been observed in networks adversarially trained with FGSM on CIFAR10/100, as shown in fig. 9a. Consistent standard test performance degradation has been observed in adversarially trained with FGSM on CIFAR10/100 as well, as shown in fig. 9b. The exact data can be found in table 2.

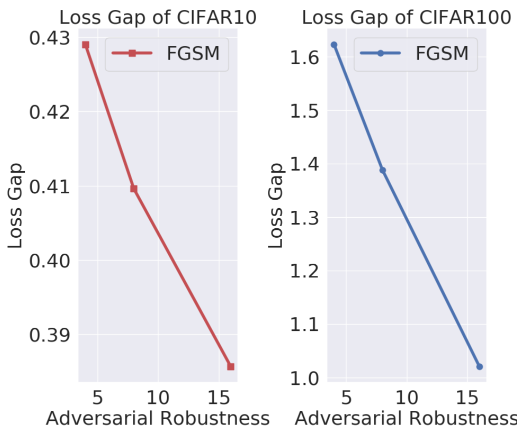
Adversarial robustness concentrates examples around decision boundaries. In section 4.2.1, we find that the distributions of margins become more concentrated around zero as AR grows. The phenomenon has been observed consistently in networks adversarially trained with FGSM on CIFAR10/100, as shown in fig. 10. Phenomenon in fig. 5a and fig. 5b are also reproduced consistently in fig. 11a and fig. 11b. Please refer to section 4.2.1 for the analysis of the results. Here we mainly present counterparts of the results analyzed there.

Adversarial robustness reduces the standard deviation of singular values of weight matrices in the network. In section 4.2.2, we find that for NNs with stronger adversarial robustness, the standard deviation of singular values of weight matrices is smaller in most layers. The phenomenon has been consistently observed in NNs trained with FGSM on CIFAR10/100, as shown in fig. 11c and fig. 11d. Please refer to section 1.1 and section 4.2.2 for the analysis of the results. Here we mainly present counterparts of the results analyzed there.

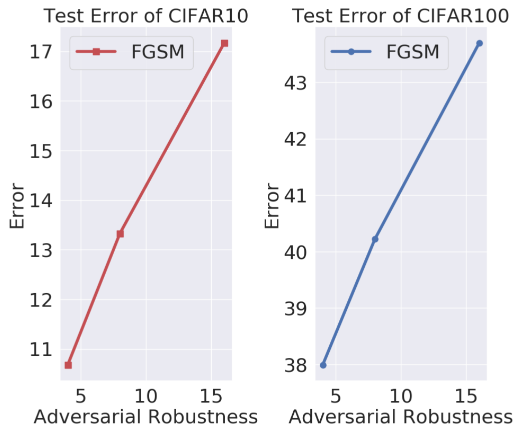
In conclusion, all key empirical results have been consistently observed in NNs trained with FGSM.

Table 2. Data of fig. 9

Dataset		Defensive Strength 4	Defensive Strength 8	Defensive Strength 16
CIFAR10	Test Acc.	89.32	86.67	82.83
	Trn Loss	0.038	0.086	0.252
	Test Loss	0.467	0.495	0.637
	Δ Loss	0.429	0.409	0.385
CIFAR100	Test Acc.	62.01	59.78	56.30
	Trn Loss	0.469	0.656	0.822
	Test Loss	1.776	1.723	1.797
	Δ Loss	1.307	1.067	0.975

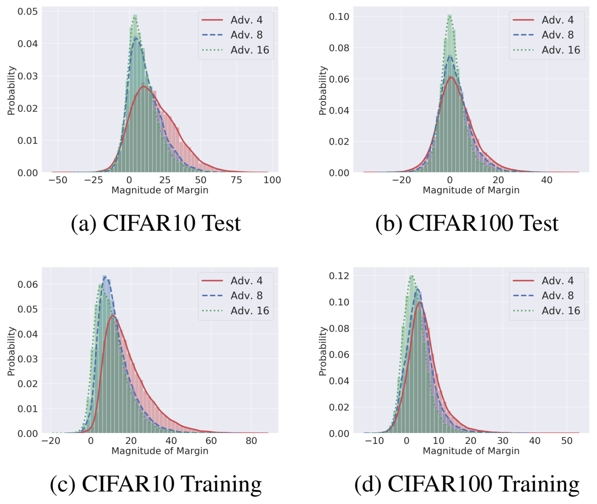


(a)



(b)

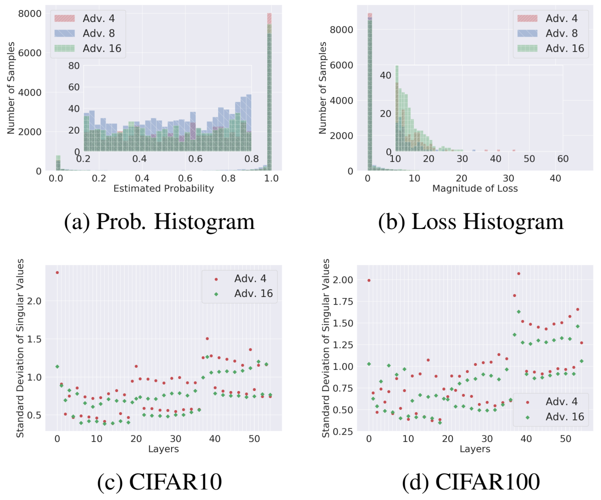
Figure 9. Experiment results on CIFAR10/100. The network is ResNet-56 (He et al., 2016). The unit of x-axis is the adversarial robustness (AR) strength of NNs, c.f. the beginning of section 4. **(a)** Plots of loss gap between training and test datasets v.s. AR strength. **(b)** Plots of error rates on training and test datasets v.s. AR strength.



(a) CIFAR10 Test (b) CIFAR100 Test

(c) CIFAR10 Training (d) CIFAR100 Training

Figure 10. Margin distributions of NNs with AR strength 4, 8, 16 on Training and Test sets of CIFAR10/100.



(a) Prob. Histogram (b) Loss Histogram

(c) CIFAR10 (d) CIFAR100

Figure 11. **(a)(b)** are histograms of estimated probabilities and losses respectively of the test set sample of NNs trained AR strength 4, 8, 16. We plot a subplot of a narrower range inside the plot of the full range to show the histograms of examples that are around the middle values to show the change induced by AR that induces more middle valued confidence predictions. **(c)(d)** are standard deviations of singular values of weight matrices of NNs at each layer trained on CIFAR10/100 with AR strength 4, 16. The AR strength 8 is dropped for clarity.

C. Proof of theorem 3.1

C.1. Algorithmic Robustness Framework

In order to characterize the bound to the GE, we build on the *algorithmic robustness* framework (Xu & Mannor, 2012).

We introduce the framework below.

Definition 8 ($(K, \epsilon(\cdot))$-robust)**.** *An algorithm is $(K, \epsilon(\cdot))$ robust, for $K \in \mathbb{N}$ and $\epsilon(\cdot) : \mathcal{Z}^m \mapsto \mathbb{R}$, if $\mathcal{Z}$ can be partitioned into K disjoint sets, denoted by $\mathcal{C} = \{C_k\}_{k=1}^K$, such that the following holds for all $s_i = (\boldsymbol{x}_i, y_i) \in S_m, z = (\boldsymbol{x}, y) \in \mathcal{Z}, C_k \in \mathcal{C}$:*

$$\forall s_i = (\boldsymbol{x}_i, y_i) \in C_k, \forall z = (\boldsymbol{x}, y) \in C_k$$
$$\implies |l(f(\boldsymbol{x}_i), y_i) - l(f(\boldsymbol{x}), y)| \leq \epsilon(S_m).$$

The gist of the definition is to constrain the variation of loss values on test examples w.r.t. those of training ones through local property of the algorithmically learned function f. Intuitively, if $s \in S_m$ and $z \in \mathcal{Z}$ are "close" (e.g., in the same partition C_k), their loss should also be close, due to the intrinsic constraint imposed by f.

For any algorithm that is robust, Xu & Mannor (Xu & Mannor, 2012) proves

Theorem C.1 (Xu & Mannor (Xu & Mannor, 2012))**.** *If a learning algorithm is $(K, \epsilon(\cdot))$-robust and $\mathcal{L}$ is bounded, a.k.a. $\mathcal{L}(f(\boldsymbol{x}), y) \leq M \ \forall z \in \mathcal{Z}$, for any $\eta > 0$, with probability at least $1 - \eta$ we have*

$$GE(f_{S_m}) \leq \epsilon(S_m) + M\sqrt{\frac{2K\log(2) + 2\log(1/\eta)}{m}}. \quad (6)$$

To control the first term, an approach is to constrain the variation of the loss function. Covering number (Shalev-Shwartz & Ben-David, (Shalev-Shwartz & Ben-David, 2014), Chapter 27) provides a way to characterize the variation of the loss function, and conceptually realizes the actual number K of disjoint partitions.

For any regular k-dimensional manifold embedded in space equipped with a metric ρ, e.g., the image data embedded in $L^2(\mathbb{R}^2)$, the square integrable function space defined on $\mathbb{R}^2$, it has a covering number $\mathcal{N}(\mathcal{X}; \rho, \epsilon)$ of $(C_{\mathcal{X}}/\epsilon)^k$ (Verma, 2013), where $C_{\mathcal{X}}$ is a constant that captures its "intrinsic" properties, and ϵ is the radius of the covering ball. When we calculate the GE bound of NNs, we would assume the data space is a k-dimensional regular manifold that accepts a covering.

Adversarial robustness makes NNs a $(K, \epsilon(\cdot))$-robust algorithm, and is able to control the variation of loss values on test examples. Building on covering number and theorem C.1, we are able to prove theorem 3.1.

C.2. Proof

Proof of lemma 3.1 . By theorem 3 in Sokolic et al. (2017), we have

$$||I_l(\boldsymbol{x}) - I_l(\boldsymbol{x}')|| = \int_0^1 \boldsymbol{J}(\boldsymbol{x} - t(\boldsymbol{x}' - \boldsymbol{x}))dt(\boldsymbol{x} - \boldsymbol{x}') \quad (7)$$

where $\boldsymbol{J}(\boldsymbol{x})$ denotes the Jacobian of $I_l(\boldsymbol{x})$ at $\boldsymbol{x}$.

By lemma 3.2 in Jia et al. (2019), when we only have max pooling layers and ReLU as nonlinear layer in NNs, $\boldsymbol{J}(\boldsymbol{x})$ is a linear operator at a local region around $\boldsymbol{x}$. For terminology concerning regions, we follow the definitions in Jia et al. (2019). More specifically, we have

$$\boldsymbol{J}(\boldsymbol{x}) = \prod_{i=1}^{l} \boldsymbol{W}_i^{\boldsymbol{x}}$$

where $\boldsymbol{W}_i^{\boldsymbol{x}}$ is the linear mapping (matrix) induced by $\boldsymbol{J}(\boldsymbol{x})$ at $\boldsymbol{x}$. It is a matrix obtained by selectively setting certain rows of $\boldsymbol{W}_i$ to zero. For the more concrete form of $\boldsymbol{W}_i^{\boldsymbol{x}}$, refer to lemma 3.2 in Jia et al. (2019). In Jia et al. (2019), it is noted as $\boldsymbol{W}_i^q$, where q is a region where $\boldsymbol{x}$ is in.

Suppose that from $\boldsymbol{x}$ to $\boldsymbol{x}'$, the line segment $\boldsymbol{x} - \boldsymbol{x}'$ passes through regions $\{q_j\}_{j=1,\ldots,n}$. The line segment is illustrated in fig. 2b as the boldest black line segment at the upper half of the figure. In the illustration, $\boldsymbol{x} - \boldsymbol{x}'$ passes through three regions, colored coded as gray, dark yellow, light blue respectively. The line segment is divided into three sub-segments. Suppose $\boldsymbol{l}(t) = \boldsymbol{x} + t(\boldsymbol{x}' - \boldsymbol{x})$. Then the three sub-segments can be represented by $\boldsymbol{l}(t)$ as $\boldsymbol{l}(s_1)$ to $\boldsymbol{l}(e_1)$, $\boldsymbol{l}(s_2)$ to $\boldsymbol{l}(e_2)$, and $\boldsymbol{l}(s_3)$ to $\boldsymbol{l}(e_3)$ respectively, as noted on the line segment in the illustration. Originally, the range of the integration in eq. (7) is from 0 to 1, representing the integration on the line segment $\boldsymbol{l}(0)$ to $\boldsymbol{l}(1)$ in the instance space. Now, since for each of these regions trespassed by the line segment, the Jacobian $\boldsymbol{J}(\boldsymbol{x})$ is a linear operator, denoted as $\boldsymbol{W}_i^{q_j}$, the integration in eq. (7) from 0 to 1 can be decomposed as a summation of integration on segments $l(s_1)$ to $l(e_1)$ etc. In each of these integration, the Jacobian $\boldsymbol{J}(\boldsymbol{x})$ is the multiplication of linear matrices $\boldsymbol{W}_i^{q_j}$, i.e., $\prod_{i=1}^{l} \boldsymbol{W}_i^{q_j}$. Thus, eq. (7) can be written as

$$\sum_{j=1}^{n} \int_{s_j}^{e_j} \prod_{i=1}^{l} \boldsymbol{W}_i^{q_j} dt(\boldsymbol{x} - \boldsymbol{x}')$$

where s_j, e_j denotes the start and end of the segment $[s_j, e_j] \subset [0, 1]$ of the segment $[0, 1]$ that passes through the region q_j.

□

In the cases that a linear operator is applied on the feature map $I_l(\boldsymbol{x})$ without any activation function, we can also obtain a similar conclusion. Actually, such cases are just degenerated cases of feature maps that have activation functions.

Corollary C.1. *Given two elements $\boldsymbol{x}, \boldsymbol{x}'$, and $I_l(\boldsymbol{x}) = \boldsymbol{W}_l g(\boldsymbol{W}_{l-1} \dots g(\boldsymbol{W}_1 \boldsymbol{x}))$, we have*

$$||I_l(\boldsymbol{x}) - I_l(\boldsymbol{x}')|| = \sum_{j=1}^{n} \int_{s_j}^{e_j} \boldsymbol{W}_l \prod_{i=1}^{l-1} \boldsymbol{W}_i^{q_j} dt (\boldsymbol{x} - \boldsymbol{x}')$$

where symbols are defined similar as in Proof of lemma 3.1.

Now, we are ready to prove theorem C.1.

Proof of theorem C.1. Similar with the proof of theorem C.1, we partition space $\mathcal{Z}$ into the ϵ-cover of $\mathcal{Z}$, which by assumption is a k-dimension manifold. Its covering number is upper bounded by $C_{\mathcal{X}}^k/\epsilon^k$, denoting $K = C_{\mathcal{X}}^k/\epsilon^k$, and $\hat{C}_i$ the ith covering ball. For how the covering ball is obtained from the ϵ-cover, refer to theorem 6 in Xu & Mannor (2012). We study the constraint/regularization that adversarial robustness imposes on the variation of the loss function. Since we only have ϵ-adversarial robustness, the radius of the covering balls is at most ϵ — this is why we use the same symbol. Beyond ϵ, adversarial robustness does not give information on the possible variation anymore. Let T' denotes the NN without the last layer.

First, we analyze the risk change in a covering ball C_i. The analysis is divided into two cases: 1 all training samples in C_i are classified correctly; 2) all training samples in C_i are classified wrong. Note that no other cases exist, for that the radius of C_i is restricted to be ϵ, and we work on ϵ-adversarial robust classifiers. It guarantees that all samples in a ball are classified as the same class. Thus, either all training samples are all classified correctly, or wrongly.

We first study case 1). Given any example $z = (\boldsymbol{x}, y) \in C_i$, let $\hat{y} = \arg\max_{i \neq y} \boldsymbol{w}_i^T T' \boldsymbol{x}$. Its ramp loss is

$$l_\gamma(\boldsymbol{x}, y) = \max\{0, 1 - \frac{1}{\gamma}(\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T T' \boldsymbol{x}\}.$$

Note that within C_i, $(\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T T' \boldsymbol{x} \geq 0$, thus $l_\gamma(\boldsymbol{x}, y)$ is mostly 1, and we would not reach the region where $r > 0$ in definition 4. Let $u(\boldsymbol{x}) := (\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T T' \boldsymbol{x}$, and $u_{\text{min}}^i = \min_{\forall \boldsymbol{x} \in C_i} u(\boldsymbol{x})$. We have

$$l_\gamma(\boldsymbol{x}, y) \leq \max\{0, 1 - \frac{u_{\text{min}}^i}{\gamma}\} \leq \max\{0, 1 - \frac{u_{\text{min}}}{\gamma}\},$$

where u_{min} denotes the smallest margin among all partitions.

The inequality above shows adversarial robustness requires that $T'\boldsymbol{x}$ should vary slowly enough, so that in the worst case, the loss variation within the adversarial radius should satisfy the above inequality. The observation leads to the constraint on the loss difference $\epsilon(\cdot)$ defined earlier in definition 8 in the following.

Given any training example $z := (\boldsymbol{x}, y) \in C_i$, and any element $z' := (\boldsymbol{x}', y') \in C_i$, where C_i is the covering ball that covers $\boldsymbol{x}$, we have

$$\begin{aligned}
&|l_\gamma(\boldsymbol{x}, y) - l_\gamma(\boldsymbol{x}', y')| \\
=&|\max\{0, 1 - \frac{u(\boldsymbol{x})}{\gamma}\} - \max\{0, 1 - \frac{u(\boldsymbol{x}')}{\gamma}\}| \\
\leq&\max\{0, 1 - \frac{u_{\text{min}}}{\gamma}\}. \qquad (8)
\end{aligned}$$

Now we relate the margin to the margin in the instance space.

Given $z := (\boldsymbol{x}, y) \in \mathcal{Z}$, and z', of which $\boldsymbol{x}'$ is the closest points to $\boldsymbol{x}$ (measured in Euclidean norm) on the decision boundary, we can derive the inequality below.

$$\begin{aligned}
u(\boldsymbol{x}) &= u(\boldsymbol{x}) - u(\boldsymbol{x}') && (9)\\
&= \int_0^1 \boldsymbol{J}(\boldsymbol{x} - t(\boldsymbol{x} - \boldsymbol{x}')) dt (\boldsymbol{x} - \boldsymbol{x}') && (10)\\
&= \int_0^1 (\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T \prod_{i=1}^{L-1} \boldsymbol{W}_i^{\boldsymbol{x} - t(\boldsymbol{x} - \boldsymbol{x}')} dt (\boldsymbol{x} - \boldsymbol{x}') \\
&= \int_0^1 (\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T \prod_{i=1}^{L-1} \boldsymbol{W}_i^{\boldsymbol{x} - t(\boldsymbol{x} - \boldsymbol{x}')} dt (\boldsymbol{x} - \boldsymbol{x}') && (11)\\
&= \sum_{j=1}^{n} \int_{s_j}^{e_j} (\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T \prod_{i=1}^{L-1} \boldsymbol{W}_i^{q_j} dt (\boldsymbol{x} - \boldsymbol{x}') && (12)\\
&\geq \min_{y, \hat{y} \in \mathcal{Y}, y \neq \hat{y}} ||\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}}||_2 \prod_{i=1}^{L-1} \sigma_{\text{min}}^i ||\boldsymbol{x} - \boldsymbol{x}'||_2 \int_0^1 dt && (13)\\
&\geq \min_{y, \hat{y} \in \mathcal{Y}, y \neq \hat{y}} ||\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}}||_2 \prod_{i=1}^{L-1} \sigma_{\text{min}}^i ||\boldsymbol{x} - \boldsymbol{x}'||_2
\end{aligned}$$

where $\boldsymbol{J}(\boldsymbol{x})$ denotes the Jacobian of $I_l(\boldsymbol{x})$ at $\boldsymbol{x}$. eq. (10) can be reached by theorem 3 in Sokolic et al. (2017). eq. (11) can be reached because $(\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}}) \boldsymbol{W}_i^{\boldsymbol{x} - t(\boldsymbol{x} - \boldsymbol{x}')} (\boldsymbol{x} - \boldsymbol{x}')$ is the actually classification score $u(\boldsymbol{x}), u(\boldsymbol{x}')$ difference between $\boldsymbol{x}, \boldsymbol{x}'$, and by assumptions assumption 3.1, they are positive throughout. eq. (12) is reached due to corollary C.1 — in this case, the matrix $\boldsymbol{W}_l$ in corollary C.1 is of rank one.

To arrive from eq. (12) to eq. (13), we observe that $\boldsymbol{x}'$ is the closest point to $\boldsymbol{x}$ on the decision boundary. Being the closest means $\boldsymbol{x} - \boldsymbol{x}' \perp \mathcal{N}((\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}}) T')$. If the difference $\boldsymbol{x}' - \boldsymbol{x}$ satisfies $\boldsymbol{x} - \boldsymbol{x}' \not\perp \mathcal{N}(T')$, we can always remove the part in the $\mathcal{N}(T')$, which would identify a point that is closer to $\boldsymbol{x}$, but still on the decision boundary, which would be a contradiction. Then if $\boldsymbol{x} - \boldsymbol{x}'$ is orthogonal to the null space, we can bound the norm using the least singular values. We develop the informal reasoning above formally in the following.

Similarly in lemma 3.4 in Jia et al. (2019), by Cauchy interlacing law by row deletion, assuming $\boldsymbol{x} \perp \mathcal{N}(\prod_{i=1}^{L-1} \boldsymbol{W}_i^{q_j})$ ($\mathcal{N}$ denotes the null space; the math statement means $\boldsymbol{x}$ is orthogonal to the null space of $\boldsymbol{J}(\boldsymbol{x})$), we have

$$||\prod_{i=1}^{L-1} \boldsymbol{W}_i^{q_j} \boldsymbol{x}||_2 \geq \prod_{i=1}^{L-1} \sigma_{\min}^i ||\boldsymbol{x}||_2 \tag{14}$$

where $\sigma_{\min}^i$ is the smallest singular value of $\boldsymbol{W}_i$. Then conclusion holds as well for multiplication of matrices $\prod_{i=1}^{L-1} \boldsymbol{W}_i^{q_j}$, since the multiplication of matrices are also a matrix.

Notice that in each integral in eq. (12), we are integrating over constant. Thus, we have it equates to

$$\sum_{j=1}^{n} (e_j - s_j) \ (\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T \prod_{i=1}^{L-1} \boldsymbol{W}_i^{q_j} (\boldsymbol{x} - \boldsymbol{x}') \ .$$

Now we show that in each operand, $\boldsymbol{x} - \boldsymbol{x}' \perp \mathcal{N}((\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T \prod_{i=1}^{L-1} \boldsymbol{W}_i^{q_j})$. Denote T_{q_j} as $\mathcal{N}((\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T \prod_{i=1}^{L-1} \boldsymbol{W}_i^{q_j})$. Suppose that it does not hold. Then we can decompose $\boldsymbol{x} - \boldsymbol{x}'$ into two components $\boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2$, where $\boldsymbol{\Delta}_1 \perp T_{q_j}, \boldsymbol{\Delta}_2 \not\perp T_{q_j}$. We can find a new point $\boldsymbol{x}'' = \boldsymbol{x} + \boldsymbol{\Delta}_1$ that is on the boundary. However, in this case

$$||\boldsymbol{x} - \boldsymbol{x}''||_2 = ||\boldsymbol{\Delta}_1||_2 \leq ||\boldsymbol{\Delta}_1||_2 + ||\boldsymbol{\Delta}_2||_2 = ||\boldsymbol{x} - \boldsymbol{x}'||_2$$

Recall that $\boldsymbol{x}'$ is the closest point to $\boldsymbol{x}$ on the decision boundary. This leads to a contradiction. Repeat this argument for all $j = 1, \ldots, n$, then we have $\boldsymbol{x} - \boldsymbol{x}'$ be orthogonal to all $\mathcal{N}(T_{q_j})$. Thus, by the inequality eq. (14) earlier, we can arrive at eq. (13) — notice that $\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}}$ is a matrix with one column, thus also satisfies the above reasoning.

Through the above inequality, we can transfer the margin to margin in the instance space. Let $v(\boldsymbol{x})$ be the shortest distance in $||\cdot||_2$ norm from an element $\boldsymbol{x} \in \mathcal{X}$ to the decision boundary. For a covering ball C_i, let $v_{\min}^i$ be $\min_{\boldsymbol{x} \in C_i} v(\boldsymbol{x})$. Let $v_{\min}$ be the smallest $v_{\min}^i$ among all covering balls C_i that contain at least a training example. We have that

$$u_{\min} \geq \min_{y, \hat{y} \in \mathcal{Y}, y \neq \hat{y}} ||\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}}||_2 \prod_{i=1}^{L-1} \sigma_{\min}^i v_{\min}$$

Consequently, we can obtain an upper bound of eq. (8) parameterized on $v_{\min}$, as follows

$$\max\{0,1 - \frac{u_{\min}}{\gamma}\} \leq \max\{0, \\ 1 - \frac{\min_{y,\hat{y} \in \mathcal{Y}, y \neq \hat{y}} ||\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}}||_2 \prod_{i=1}^{L-1} \sigma_{\min}^i v_{\min}}{\gamma}\}.$$

Notice that only because ϵ_0-adversarial robustness, we can guarantee that $v_{\min}$ is non-zero, thus the bound is influenced by AR.

Then, we study case 2), in which all training samples $z \in C_i$ are classified wrong. In this case, for all $z \in C_i$, the $\hat{y}$ given by $\hat{y} = \arg\max_{i \neq y} \boldsymbol{w}_i^T T' \boldsymbol{x}$ in the margin operator is the same, for that $\hat{y}$ is the wrongly classified class. Its ramp loss is

$$l_\gamma(\boldsymbol{x}, y) = \max\{0, 1 - \frac{1}{\gamma} (\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T T' \boldsymbol{x}\}.$$

Note that in the case 1), it is the y that stays fixed, while $\hat{y}$ may differ from example to example; while in the case 2), it is the $\hat{y}$ stays fixed, while y may differ.

Similarly, within C_i as required by adversarial robustness, $(\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T T' \boldsymbol{x} \leq 0$, thus we always have $1 - \frac{1}{\gamma}(\boldsymbol{w}_y - \boldsymbol{w}_{\hat{y}})^T T' \boldsymbol{x} \geq 1$, implying

$$l_\gamma(\boldsymbol{x}, y) = 1.$$

Thus, $\forall z = (\boldsymbol{x}, y), z' = (\boldsymbol{x}', y') \in C_i$

$$|l_\gamma(\boldsymbol{x}, y) - l_\gamma(\boldsymbol{x}', y')| = 0. \tag{15}$$

Since only these two cases are possible, by eq. (8) and eq. (15), we have $\forall z, z' \in C_i$

$$|l_\gamma(\boldsymbol{x}, y) - l_\gamma(\boldsymbol{x}', y')| \leq \max\{0, 1 - \frac{u_{\min}}{\gamma}\}. \tag{16}$$

The rest follows the standard proof in algorithmic robust framework.

Let N_i be the set of index of points of examples that fall into C_i. Note that $(|N_i|)_{i=1 \ldots K}$ is an IDD multimonial random variable with parameters m and $(|\mu(C_i)|)_{i=1\dots K}$. Then

$$\begin{aligned}
&|R(l \circ T) - R_m(l \circ T)| \\
=&|\sum_{i=1}^{K} \mathbb{E}_{Z\sim\mu}[l(TX,Y)]\mu(C_i) - \frac{1}{m}\sum_{i=1}^{m} l(T\boldsymbol{x}_i, y_i)| \\
\leq&|\sum_{i=1}^{K} \mathbb{E}_{Z\sim\mu}[l(TX,Y)]\frac{|N_i|}{m} - \frac{1}{m}\sum_{i=1}^{m} l(T\boldsymbol{x}_i, y_i)| \\
&+|\sum_{i=1}^{K} \mathbb{E}_{Z\sim\mu}[l(TX,Y)]\mu(C_i) - \sum_{i=1}^{K} \mathbb{E}_{Z\sim\mu}[l(TX,Y)]\frac{|N_i|}{m}| \\
\leq&|\frac{1}{m}\sum_{i=1}^{K}\sum_{j\in N_i}\max_{z\in C_i}|l(T\boldsymbol{x},y) - l(T\boldsymbol{x}_j,y_j)| \qquad (17) \\
&+|\max_{z\in\mathcal{Z}}|l(T\boldsymbol{x},y)|\sum_{i=1}^{K}|\frac{|N_i|}{m} - \mu(C_i)||. \qquad (18)
\end{aligned}$$

Remember that $z = (\boldsymbol{x}, y)$.

By eq. (16) we have eq. (17) is equal or less than $\max\{0, 2(1 - \frac{u_{\min}}{\gamma})\}$. By Breteganolle-Huber-Carol inequality, eq. (18) is less or equal to $\sqrt{\frac{\log(2)2^{k+1}C_{\mathcal{X}}^{k}}{\gamma^k m} + \frac{2\log(1/\eta)}{m}}$.

The proof is finished. $\square$

D. Implementation Details

We summarize the details of the experiments in this section. The experiments are run with PyTorch (Pfeiffer, 2017).

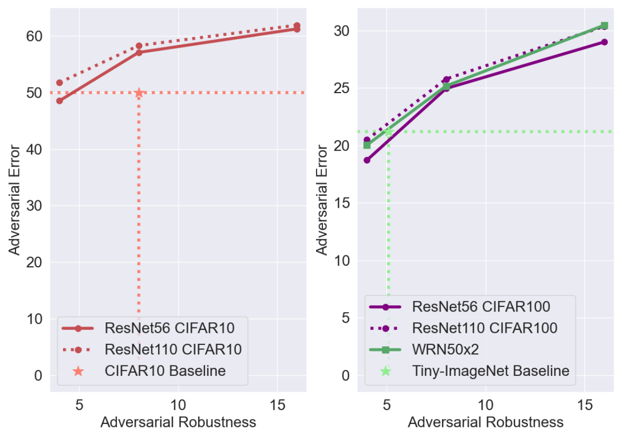


Figure 12. The plot of accuracy on *adversarial examples* v.s. adversarial defense strength built in NNs. The dotted line of which the intersections are marked by stars are adversarial accuracy in Madry et al. (2018) (CIFAR10), in Li et al. (2018) (Tiny ImageNet) under similar adversarial attack strength.

D.1. Datasets

CIFAR10/100. Each CIFAR dataset consists of $50,000$ training data and $10,000$ test data. CIFAR-10 and CIFAR-100 have 10 and 100 classes respectively. Our data augmentation follows the standard manner in Lee et al. (2015): during training, we zero-pad 4 pixels along each image side, and sample a 32×32 region cropped from the padded image or its horizontal flip; during testing, we use the original non-padded image.

Tiny-ImageNet. Tiny-ImageNet is a subset of ImageNet dataset, which contains 200 classes rather than $1,000$ classes. Each class has 500 training images and 50 validation images. Images in the Tiny-ImageNet dataset are of 64×64 pixels, as opposed to 256×256 in the full ImageNet set. The data augmentation is straightforward: an input image is 56×56 randomly cropped from a resized image using the scale, aspect ratio augmentation as well as scale jittering. A single 56×56 cropped image is used for testing.

D.2. Experiments in section 4.1

CIFAR10/100 Models and Training. The models for CIFAR10/100 are the same as the ones in appendix B.3, except that we do not use spectral normalization anymore. CIFAR100 has 100 output neurons instead of 10.

Tiny-ImageNet Model. For Tiny ImageNet dataset, we use

Table 3. Raw data of CIFAR10 dataset for plots in fig. 3, fig. 7 and fig. 12.

Method		Defensive Strength		
		4	8	16
ResNet-56 + Adv Trn	Trn Acc.	99.51	98.45	95.97
	Test Acc.	88.86	87.51	84.89
	ΔAcc.	10.65	10.94	11.08
	Trn Loss	0.014	0.043	0.105
	Test Loss	0.683	0.649	0.650
	ΔLoss	0.669	0.606	0.545
	PGD	65.92	65.24	72.16
ResNet-110 + Adv Trn	Trn Acc.	99.95	99.62	98.42
	Test Acc.	89.20	87.09	85.02
	ΔAcc.	10.75	12.53	13.40
	Trn Loss	0.002	0.010	0.044
	Test Loss	0.825	0.813	0.729
	ΔLoss	0.823	0.803	0.685
	PGD	58.02	66.94	72.40

Table 4. Raw data of CIFAR100 dataset for the plot in fig. 3, fig. 7 and fig. 12.

Method		Defensive Strength		
		4	8	16
ResNet-56 + Adv Trn	Trn Acc.	88.73	86.97	82.17
	Test Acc.	61.31	60.87	59.43
	ΔAcc.	27.42	26.10	22.74
	Trn Loss	0.357	0.413	0.570
	Test Loss	2.063	2.106	1.978
	ΔLoss	1.706	1.693	1.408
	PGD	30.52	40.99	48.81
ResNet-110 + Adv Trn	Trn Acc.	96.91	94.55	90.90
	Test Acc.	61.48	61.26	59.56
	ΔAcc.	35.43	33.29	31.34
	Trn Loss	0.098	0.171	0.278
	Test Loss	2.645	2.413	2.323
	ΔLoss	2.547	2.241	2.045
	PGD	33.33	42.08	50.99

50-layered wide residual networks with 4 groups of residual layers and $[3, 4, 6, 3]$ bottleneck residual units for each group respectively. The 3×3 filter of the bottleneck residual units have $[64 \times k, 128 \times k, 256 \times k, 512 \times k]$ feature maps with the widen factor $k = 2$ as mentioned in Zagoruyko & Komodakis (2016). We replace the first 7×7 convolution layer with 3×3 filters with stride 1 and padding 1. The max pooling layer after the first convolutional layer is also removed to fit the 56×56 input size. Batch normalization layers are retained for this dataset. The weights of convolution layers for Tiny ImageNet are initialized with Xavier uniform (Glorot & Bengio, 2010). Again, all dropout layers are omitted.

Tiny-ImageNet Training. The experiments on the Tiny-ImageNet dataset are based on a mini-batch size of 256 for 90 epochs. The initial learning rate is set to be 0.1 and decayed at 10 at 30 and 60 epochs respectively. All experiments are trained on the training set with stochastic gradient descent with the momentum of 0.9.

Results. The data for fig. 3 and fig. 7 are given in table 3, table 4 and table 5. More specifically, the data on CIFAR10 are given in table 3. The result on CIFAR100 are given in table 4. The result on Tiny-ImageNet are given in table 5.

Adversarial Robustness Attack Method. The adversarial accuracy is evaluated against l_∞-PGD (Madry et al., 2018) untargeted attack adversary, which is one of the strongest white-box attack methods. When considering adversarial attack, they usually train and evaluate against the same perturbation. And for our tasks, we only use the moderate adversaries that generated by 10 iterations with steps of size 2 and maximum of 8. When evaluating adversarial robustness, we only consider clean examples classified correctly originally, and calculate the accuracy of the adversarial examples generated from them that are still correctly classified. The adversarial accuracy is given in table 3 table 4 table 5, the row named "PGD", and plotted in fig. 12.

Table 5. Raw data of Tiny-ImageNet dataset for the plot in fig. 3, fig. 7 and fig. 12.

Method		Defensive Strength			
		0	4	8	16
Wide ResNet + Adv Trn	Trn Acc.	79.12	73.71	66.17	60.73
	Test Acc.	63.43	62.09	61.09	57.36
	ΔAcc.	15.69	11.62	5.08	3.37
	Trn Loss	0.874	1.080	1.384	1.641
	Test Loss	1.561	1.637	1.689	1.806
	ΔLoss	0.687	0.557	0.305	0.165
	PGD	0.00	32.26	41.20	53.12

D.3. Experiments in appendix B.3

Models. We use ResNet-type networks (Zhang et al., 2018). Given that we need to isolate factors that influence spectral complexity, we use ResNet without additional batch normalization (BN) layers. To train ResNet without BN, we rely on the fixup initialization proposed in Zhang et al. (2018). The scalar layers in Zhang et al. (2018) are also omitted, since it changes spectral norms of layers. Dropout layers are omitted as well. Following Sedghi et al. (2018), we clip the spectral norm every epoch rather than every iteration.

Training. The experiments on CIFAR10 datasets are based on a mini-batch size of 256 for 200 epochs. The learning rate starts at 0.05, and is divided by 10 at 100 and 150 epochs respectively. All experiments are trained on training set with stochastic gradient descent based on the momentum of 0.9.

Table 6. Raw data for fig. 6. SP denotes spectral norm.

Strength of Spectral Normalization		Defensive Strength 4	8	16
SP 1	Trn Acc.	96.91	94.38	90.58
	Test Acc.	90.47	88.87	85.82
	ΔAcc.	6.44	5.51	4.76
	Trn Loss	0.092	0.159	0.265
	Test Loss	0.316	0.353	0.432
	ΔLoss	0.224	0.194	0.168
	PGD	57.93	69.98	75.98
SP 3	Trn Acc.	99.65	98.51	95.94
	Test Acc.	90.02	88.07	85.43
	ΔAcc.	9.63	10.44	10.51
	Trn Loss	0.010	0.039	0.107
	Test Loss	0.606	0.580	0.577
	ΔLoss	0.596	0.541	0.470
	PGD	56.83	67.73	73.41
SP 5	Trn Acc.	99.57	98.33	95.96
	Test Acc.	89.53	88.09	85.32
	ΔAcc.	10.04	10.24	10.64
	Trn Loss	0.012	0.045	0.105
	Test Loss	0.649	0.602	0.611
	ΔLoss	0.638	0.557	0.506
	PGD	54.91	65.96	72.37
SP Uncontrolled	Trn Acc.	99.51	98.45	95.97
	Test Acc.	88.86	87.51	84.89
	ΔAcc.	10.65	10.94	11.08
	Trn Loss	0.014	0.043	0.105
	Test Loss	0.683	0.649	0.650
	ΔLoss	0.669	0.606	0.545
	PGD	65.92	65.24	72.16

Results. The data for fig. 6 are given in table 6.