Robustness evaluation of various Deep Neural Networks under adversarial attack

Junyi Guan
MBZUAI
Junyi.Guan@mbzuai.ac.ae

Xiangrui Ke
MBZUAI
xiangrui.ke@mbzuai.ac.ae

Xingyu Qu
MBZUAI
xingyu.qu@mbzuai.ac.ae

Zhengyu Hu
MBZUAI
zhengyu.hu@mbzuai.ac.ae

Abstract

Despite the efficiency and scalability that have dramatically improved, including state-of-the-art neural networks, most machine learning models are vulnerable to adversarial attacks. Well-trained models misclassify samples that are only slightly different from correctly classified ones where the differences are indistinguishable to humans. Recently, with the success of advancing natural language processing, transformer-based models are expected to bring revolutionary changes to computer vision. Do transformers also possess good adversarial robustness? In this work, we conducted a study to examine the adversarial robustness and compare it with CNN baselines, which focus on a classical computer vision task —- image classification. In particular, our attention is mainly focused on how to conduct a convincing empirical analysis through experiments, instead of simply validating an idea. Guided by some related works, we compare the robustness between CNNs and vision transformers as well as the adversarial performances among some well-known attack methods in various ways. At last, we try to utilize some insights gained from our experiments to enhance the performance of our models.
Keywords: Robustness Evaluation, Adversarial Attack, Deep Neural Networks

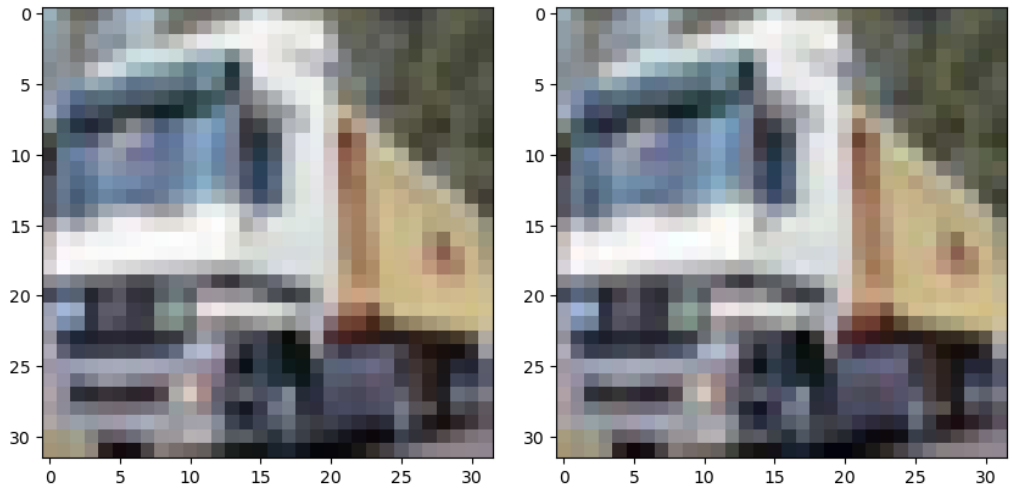
1 Introduction

The rapid development of deep learning over the past decade has led to a surge in human capabilities in certain application domains, such as in the computer vision field [16, 23, 14]. In the very beginning, convolution is at the heart of the computer vision solutions[11, 18], which owns the ability to extract useful features from extremely high-dimensional image information. However, deep neural networks are still generally considered as a black-box model, and thus subject to erratic performance in real-world applications. Later works have shown that the convolutional neural networks (CNNs) based model can be very vulnerable when encountering attacks [25, 13]. To address this issue, many adversarial training strategies are derived. The most famous of which is the generative adversarial networks (GANs) [12].

With the great success of the self-attention based mechanism and transformers [27] in the area of natural language processing [8, 1], multiple works have tried transferring this technique into the computer vision domain. Some earlier works focus on combining the CNN-based structure and the self-attention [28], while recent works show that the pure transformer (*a.k.a.* vision transformer) model can even perform better than the former [9, 20].

A natural question arises here: does the vision transformer also possess a good robustness behavior? In just two years, there are already many efforts put into this puzzle where the majority of which claims

Preprint. Under review.



(a) True Truck (b) False Frog

Figure 1: 1a is a photo of a truck from the CIFAR-10 dataset [15]. We perform the FGSM on a pre-trained Resnet [14] model using this picture. Before the attack, the model can correctly identify the truck. However, after adding some minor noises generated by FGSM into the original picture, the model classifies it as a frog, as shown in 1b.

that transformer is more robust than the CNN-based classifier [31, 10]. At the same time, however, recent work shows some different results [22]. A core reason that lies behind this contradiction is whether the experiments are conducted using a proper evaluation, which is ambiguous to adjust even nowadays [2].

Motivated by this, the objective of this paper is to revisit the adversarial robustness of CNNs and vision transformers on a classical computer vision task — image classification. In particular, our attention is mainly focused on how to conduct a convincing empirical analysis through experiments. Guided by some related works [2, 22], we compare the robustness between CNNs and vision transformers as well as the adversarial performances among some well-known attack methods in various ways. By comparing with previous work, we hope the results displayed in this paper can contribute to the future development of this area and to the evaluation of general machine learning paradigms.

Meanwhile, we try to explain the differences in robustness between different models. In particular, we study the Vapnik–Chervonenkis (VC) [26] dimension of neural networks which is used to measure the expressive power of networks on binary classification tasks. Inspired by VC dimension, we also define a new measure for the multi-classification network called Good Pattern Proportion(GPP), which is designed based on our another creation: Last Layer Activation Patterns(LLAPs), which inspired by activation patterns[30]. After that, we design 2 algorithms to roughly estimate good pattern proportion over different network topologies. Afterward, we explore if the GPP is consistent with the performance of the network. At last, we utilize some insights gained from the experiments to enhance the performance of our model.

2 Related Work

Adversarial Attacks. As stated above, the performance of deep learning models can be easily degraded under attacks. One of the early investigations is shown in [25]. In this paper, the authors formalize a box-constrained optimization problem to generate adversarial samples which will be misclassified by the model while they stay nearly the same as images that the model can verify correctly. A surprising fact is that those adversarial samples generated by one model also can fool other completely different models trained on different datasets with different hyper-parameters, which reveals some inherent flaws of the vanilla deep learning methods. An illustration of a successful attack using the Fast Gradient Sign Method (FGSM) [13] is shown in Figure 1.

Roughly speaking, current adversarial attacks can be divided into three categories: gradient based, score based, and decision based according to the information provided [19]. The majority of attacks proposed up to now belong to the first class, where the adversarial examples are generated using the gradients of the loss w.r.t the input data. Besides the primitive FGSM, this class of methods includes the famous and efficient Projected Gradient Descent Attack (PGD) [21], the Basic Iterative Method (BIM) [17], and the C&W attack [5]. In this paper, we will mostly focus on this type of attack.

As for the gradient based adversarial attacks, the process of generating an adversarial example can be viewed as solving an optimization problem. Different kinds of optimization formulations lead to distinct ways to generate adversarial examples. Two common approaches formulate the problem as a constraint optimization problem or a regularized optimization problem [19]. We will cover both in this paper.

Robustness Evaluation In this paper, we focus on the adversarial robustness evaluation of the model. The adversarial robustness evaluation is often used to validate the performance of some defense methods. A defense method refers to the strategy that strengthens the robustness of machine learning algorithms against some adversaries. Contrary to the vigorous development of research on attack methods, a long-last concern stands in the domain of defense research is the improper evaluation of the proposed defense method [2, 5, 3, 4]. This issue is a very critical factor preventing the area from expanding smoothly. In order to tackle this problem, some useful guidelines are summarized and stated in [2].

Later in the paper, we will view the advanced model structures like the vision transformer as a "defense method" evolved from early models such as the CNNs. Therefore, we can apply similar evaluation schemes to justify the real effect brought by the structures or other techniques.

Last Layer Activation Patterns VC dimension was first introduced by Vladimir Vapnik and Alexey Chervonenkis[26]. Eduardo D. Sontag defined the VC dimension for Neural Networks and provided some methods to estimate VC dimensions for some networks[24]. Huan Xiong, defined activation patterns and the number of linear regions[30] of ReLU CNN. LLAP is a special case of general activation patterns. One of the ideas of estimating LLAP is inspired by the idea used to estimate the number of linear regions[6].

3 Methodology

In this section, we introduce some methods or tools adopted in later experiments and analyses.

3.1 Adversarial Attacks

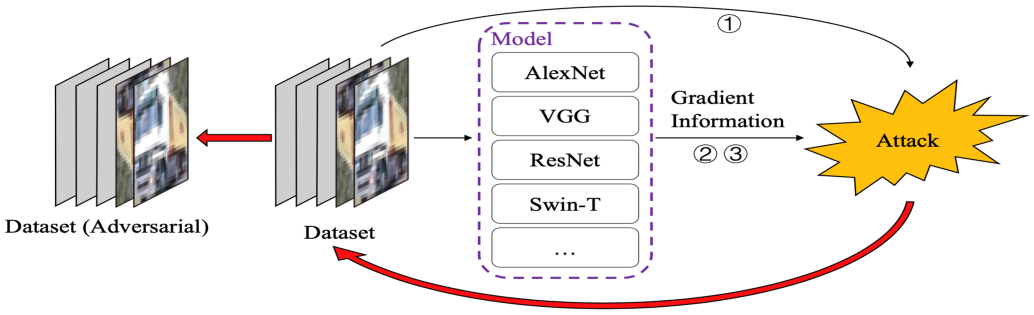


Figure 2: Three attack methods were conducted on ten models in our experiment.

3.1.1 Gaussian Noise

When all the network information including the architectures, weights and gradients is unknown, the most direct perturbation is to add Gaussian noise for each pixel. This kind of attacks are also called a

'Black Box' attack since the attacker knows nothing about the network. The update rule is shown in (1)

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + Gaussian(\mathbf{0}, \sigma^2), \tag{1}$$

where $\boldsymbol{x}$ is the image data and the noises added for each pixel are independent Gaussian random variables with mean 0 and standard deviation σ. It is also worth mentioning that this type of attack is not very efficient for some models, so we need to try other types of attacks.

3.1.2 Fast Gradient Sign Method

When all the information of the network is known, the attackers can use this information to perturb the input. FGSM is one of the simplest gradient-based adversarial approaches——a 'White Box' attack. It generates adversarial samples based on the sign of gradients of the loss function w.r.t the image data. A budget ϵ controls the degree of perturbation. The update rule is shown in (2).

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{w}; \boldsymbol{x}, y)\right) \tag{2}$$

where $\boldsymbol{w}$ represents the parameters of the network, $\boldsymbol{x}$ is the image data, and y is the ground truth label. form (2), it is clear that the direction of the perturbation and gradient form an acute angle, which means the perturbation will increase the cost function. Thus, moving in that direction will increase the chance to change the class prediction.

The fast gradient sign method is a popular choice for generating adversarial examples because it is fast and easy to implement. It can be used to quickly test the robustness of a model without requiring a lot of computational resources. Additionally, the adversarial examples generated using this method are often effective at fooling the model, making it a useful tool for evaluating the model's robustness.

3.1.3 Projected Gradient Descent

Projected Gradient Descent(PGD) is also a 'White Box' attack which only allows the perturbation within a $B_\infty(0)$ ball and repeats the updated rule of FGSM several times.

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \mathcal{P}(\epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{w}; \boldsymbol{x}, y)\right)) \tag{3}$$

$\mathcal{P}$ is a function that projects the perturbation back to the constraint.

Projected gradient descent is an algorithm that can be used to generate adversarial examples. It works by iteratively modifying the input to a machine learning model in a way that is intended to cause the model to make a mistake. At each step, the algorithm calculates the gradient of the loss function with respect to the input and uses this gradient to update the input. The input is then projected back onto the space of valid inputs, so that it remains within the constraints of the problem. This process is repeated until the desired level of adversarial perturbation is achieved.

It is often used in conjunction with the fast gradient sign method to produce more effective adversarial examples. This combined approach can be more effective at fooling the model than either method alone.

3.2 Last Layer Activation Patterns(LLAP)

3.2.1 Definition

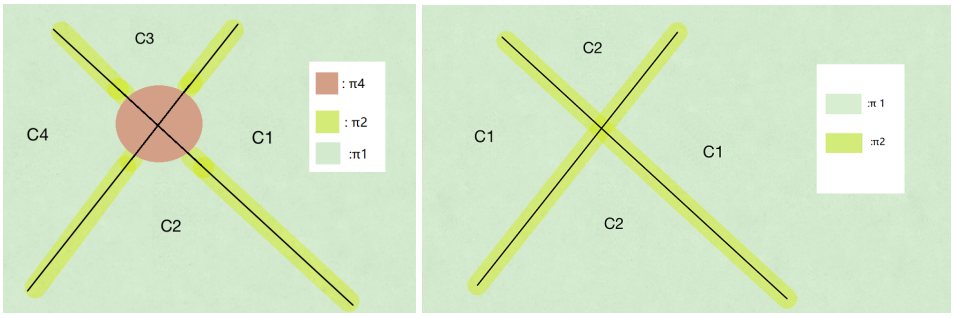
Definition 1. *Let $\mathcal{N}$ is a neural network for k classification and LLAP of $\mathcal{N}$ is a function f from a finite set $\mathbb{Z}_k$ to the set $\{-1, 1\}$.*

Definition 2. *Let $Linput(n, \theta)$ be the input of the nth neuron in the last layer and θ is the parameter of $\mathcal{N}$. We say $\mathcal{N}$ has LLAP f $(f \triangleleft \mathcal{N}) \Leftrightarrow \exists \theta \in \Theta, \forall n \in \mathbb{Z}_k, f(n) \cdot Linput(n) > 0$ For example, if $Linput(n, \theta)$ with 3 classes are $1, 5, -6$, we say $f \triangleleft \mathcal{N}$ and $f(0) = 1, f(1) = 1, f(2) = -1$*

Definition 3. *The number of LLAP for a neural network is defined by $\mathcal{L}_\mathcal{N} := \#\{f | \exists \theta \in \Theta, f \triangleleft \mathcal{N}_\theta\}$*

Remark 1. *By combinatorics, $\mathcal{L}_\mathcal{N} \leq 2^k$.*

Definition 4. *Let the pattern type $\pi_n := \{f | the\ number\ of\ positive\ output\ of\ f\ is\ exactly\ n\}$. For example, let $f(0) = 1, f(1) = 1, f(2) = -1 \implies f \in \pi_2$.*



(a) LLAPs Illustration (b) LLAPs Illustration

3.2.2 Relation Between Robustness

How LLAPs related to the model robustness? In order to illustrate the relation between LLAPs and robustness, We can view the input spaces according to 2 types of partition. The first one is partitioning the whole input spaces into data classes and more precisely, partition boundaries are the decision boundaries. The second one is that partitioning according to the type of the LLAPs. We can visualize these 2 partitions from Figure 3a. This figure is just an illustration. Why we use the areas of coloring to visualize different types of LLAPs is that if the output of some neurons at the last layer has positive values for an input image, we can infer that the model finds the input is similar to some classes. So slightly perturbing the input might result in changing the model prediction. So we can use the area near decision boundaries to informally represent this situation. It is clear that if the input is in $C_1 \cap \pi_1$. A small perturbation adding around to this point will not affect the class prediction of the model. So we can claim that pattern π_k is the most robust pattern when $k = 1$. However if the input is in $C_1 \cap \pi_4$. A small perturbation will have a high possibility to bring the point to other 3 classes. Thus, we can say pattern π_k is not robust for large k. Finally, how about π_k and π_j if $k \geq j \geq 1$. We can study it using the idea of entropy. If the input is in pi_2 and moves it slightly, it will result in 2 possibilities. However, If the input is in π_4, the destination of the perturbation is much more uncertain than π_2, in other words, larger entropy. For example, when the input is at one of the decision boundaries (in π_2) but not the intersection, the entropy of the perturbation is 1. when the input is at the intersection between 2 decision boundaries (in π_4), the entropy of the perturbation is 2. In conclusion as k increase, we claim that the pattern π_k will become less robust. So The motivation to design LLAPs is that it relates to robustness and decision boundary. Decision boundary for a network with high dimensional input spaces is hard to study, however, LLAPs is easy to compute.

3.2.3 Compare Model Robustness

After finding out the relation between LLAPs and robustness, we need to decide on a method to compare the robustness between two models. There may be many different type of LLAPs for a model, so we can manually choose a threshold to define which type of patterns are good(robust). From last section, we know that for a pattern π_k the robustness will decrease as k increase. So we can define the good patterns as follow.

Definition 5. *Let* $\Pi_n := \{f | f \in \pi_k, 0 \leq k \leq n\}$ *be a set of good patterns. For example, if* $f(0) = 1, f(1) = 1, f(2) = -1 \implies f \in \pi_2 \subset \Pi_2 \subset \Pi_3$

After specifying what should be good pattern for the models, we can compare the robustness between two models according to the size of the good pattern region. Assuming Figure 3a and Figure 3b are model 1 and 2. We can say the model2 is more robust than 1, if we define good pattern as Π_3. Since the area of the good region in Figure 2 is larger than that in Figure 1. The difference of region is the area in π_4.

How to choose k in for the good pattern? We need to choose a relatively small k for Π_k, since only small the k for π_k is robust. And also, Π_n is meaningless if n is equal to the total number of classes, Since the area of the good pattern is the same for all models. For example, in a ternary classification task, k is only meaningful when k is 1 or 2.

3.2.4 Algorithm

It is difficult to compare the size of areas of a good region, but we can use a probability approach to represent the size of area. So the new measure is Good Pattern Proportion(GPP), which is the probability that a randomly choose input locate at good pattern. we propose 2 algorithms to approximate two different probabilities and each of them has its own purpose.

p-good-pattern1

p-good-pattern1 is the first algorithm.In the pseudo code, X is the data, $\mathcal{N}_\theta$ is the model,n is the number of classes,s is the number of data we choose from the data X. k is the threshold to be regards as good pattern. This algorithm is fully depend on the data, we fist learn a fix weights θ from X and defining a suitable k for good patterns Π_k.And the algorithm will return the probability that a randomly chosen real data point is in good region. The implementation of the algorithm is simple, we just need to randomly choose some data point in X and calculate the ratio between the number of occurrences of data in Π_k and size of the data.

Algorithm 1 p-good-pattern1

procedure LLAP-COUNT1($X, \mathcal{N}, n, s, k$)
  $GoodPatterCounter \leftarrow 0$
  for $i = 1 : s$ **do**
    $output \leftarrow \mathcal{N}(X_i)$
    if The number of positive entries of the $n - dimension$ output vector is between $1 : k$ **then**
      $GoodPatterCounter \leftarrow GoodPatterCounter + 1$
    end if
  end for
  $ProGood \leftarrow GoodPatterCounter/s$
  return $ProGood$
end procedure

p-good-pattern2

p-good-pattern2 is the second algorithm. The algorithm randomly generates m different set of weights of the same model. And for each model, randomly generate s data and estimate the proportion for data is in good pattern Π_k. Finally, the algorithm returns the average proportions for all m models.

Algorithm 2 p-good-pattern2

procedure P-GOOD-PATTERN2($X, \mathcal{N}, n, m, s, k$)
  for $i = 1 : m$ **do**
    Sample $\theta_i \sim Gaussian(0, I)$
    $GoodPatterCounter_i \leftarrow 0$
    for $j = 1 : s$ **do**
      Sample $x_j \sim Gaussian(0, I)$
      $output_j \leftarrow \mathcal{N}(X_j)$
      if The number of positive entries of the $n - dimension\ output_j$ vector is between $1 : k$ **then**
        $GoodPatterCounter_i \leftarrow GoodPatterCounter_i + 1$
      end if
    end for
    $ProGood_i \leftarrow GoodPatterCounter_i/s$
  end for
  return average $ProGood_i$
end procedure

3.2.5 Algorithm Comparison

p-good-pattern1 tend to find a robust network architecture for a particular task and the distribution of the real data set. For example, searching which network architecture is the most robust according

to the CIFAT-10 dataset. However, p-good-pattern2 tend to find a robust network architecture for a general classification task. For example finding the most robust network for binary classification task. The advantage of p-good-pattern1 it is fast and easy to find a suitable network for a very specific classification task. p-good-pattern2 is very slow but has potential to find a ideal structure which perform well in many different tasks.

4 Experiments

4.1 Datasets and Preprocessing

We choose the CIFAR-10 dataset [15] as the basic dataset to conduct our experiments. It's a standard dataset to evaluate model performance on image classification. The dataset consists of 60k images from 10 classes, each of which owns an equal proportion. Originally, the dataset is divided into a training set including 50k samples and a test set containing the rest 10k images.

In the following, we always follow the standard preprocessing procedures adopted from the original work when fine-tuning some pre-trained model on the dataset,

Considering the limited time, we only choose a small subset of the entire dataset to run experiments for the midway report. In detail, we uniformly select 1k samples from the training set and the test set, respectively.

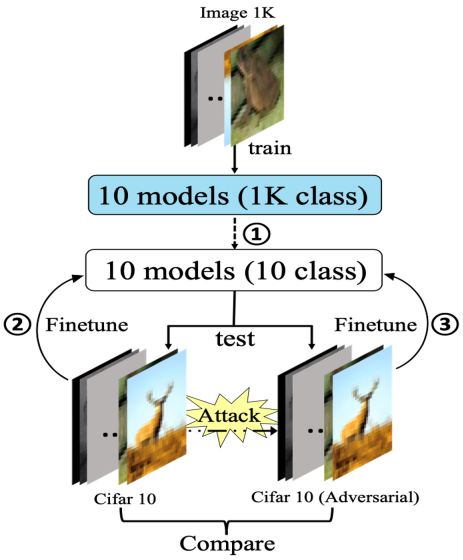


Figure 4: Evaluation Process

4.2 Models

We choose ten commonly used models from the CNN-based class and the vision transformer class to run experiments, each of which consists of five models. All models are pre-trained on the ImageNet-1K dataset [7]. Specifically, the CNN-based models include the AlexNet [16], VGG [23], and the ResNet [14]. The other class comprises the ViT [9] and the SwinT [20]. For most of the models, we utilize several variants different in the size of the inner architecture, *e.g.* ResNet18 and ResNet50 from the ResNet class.

4.3 Report on the Clean Dataset

Following the guide in [2], we first report the model performances on the clean dataset. For each model, we evaluate the the classification accuracy on the test dataset. The results are shown in Table 2. The second column refers to the accuracy using the original pre-trained model without fine-tuning on our test set. To manage this, the model firstly classify each sample as one of the primitive 1k categories. Then we map each category to one of the CIFAR-10 classes based on the predictions. The class is selected as the majority ground truth among the images predicted as this category. In this way, we can calculate the accuracy of the original model. Then we fine tune the model. We change the final fully connected linear layer to make the dimension of the output vector equal to 10. After we randomly initializing the new layer, the model is then trained using the training set. The rest columns in Table 2 displays the test accuracy of the model after each epoch. Although we only run 5 epochs for the consideration of limited time, we can also gain some basic understanding from the results. A rough illustration of the evaluation process is shown in Figure 4.

Model	Suc. Rate
AlexNet	96.19%
VGG11	98.78%
VGG13	98.09%
ResNet18	99.51%
ResNet50	92.91%
ViT_B_32	98.28%
ViT_B_16	99.67%
Swin_T	95.84%
Swin_S	97.18%
Swin_B	97.74%

Table 1: The probability that the model discriminates the sample into the same class before and after the attack.

It's clear that all the transformer models outperform the CNN models regarding the best accuracy. The vision transformer model owns a competitive performance even without fine-tuning, which owns

	Original	E = 1	E = 2	E = 3	E = 4	E = 5
AlexNet	54.9%	63.8%	59.2%	62.8%	63.5%	**65.9%**
VGG11	59.7%	67.7%	62.8%	68.6%	72.2%	**74.8%**
VGG13	58.2%	64.3%	66%	72.8%	72.9%	**75.5%**
ResNet18	59.8%	50.6%	71.1%	73.7%	**79.2%**	78.5%
ResNet50	68.7%	45.4%	63.2%	68.7%	77.6%	**80.7%**
ViT_B_32	81.2%	75.2%	74.7%	76.7%	**82.5%**	81.2%
ViT_B_16	84%	82%	**89.3%**	84.5%	88.5%	80.3%
Swin_T	75.7%	71.8%	76.9%	82.9%	**87.2%**	85.1%
Swin_S	82.3%	72.4%	85.1%	86.1%	**90%**	88.9%
Swin_B	83.8%	83.6%	85.2%	88.5%	**90.8%**	90.1%

Table 2: Clean Accuracy. The best results in each row are in bold face. The row order is based on the best accuracy of each model, where the first five turn out to be CNNs.

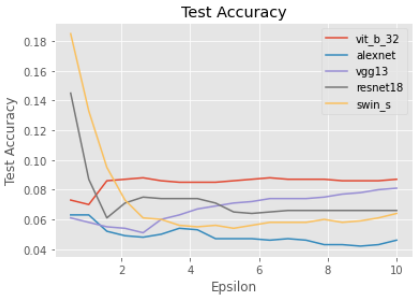


Figure 5: Adversarial Accuracy. Each curve tracks the test accuracy of the model as ϵ grows from 0.011 to 10. We choose one variant from each model class to compare their performances.

Models:	AlexNet	VGG1	VGG13	ResNet18	ResNet50
Probability:	10.6%	18.2%	16.2%	3.8%	17.6%
Models:	ViT_B_32	ViT_B_16	Swin_T	Swin_S	Swin_B
Probability:	17%	13.8%	20%	26.2%	48%

Table 3: Result of test for GGP measure.

to the pre-training on a much larger dataset. On the other hand, VGG and ResNet variants can also achieve a higher capability after fine-tuning for several rounds. Indeed, after training for 25 rounds on the entire CIFAR-10 training set in the experiments, we notice that the performances of the VGG and ResNet variants can attain a similar level of the transformer models. Therefore, the comparison w.r.t the adversarial robustness is more appealing.

4.4 Insights From the FGSM

We first adopt the simplest FGSM to attack our models where the algorithm is implemented from scratch. Each model is trained for 5 epochs and chosen as the one with the best generalization performance. To investigate the effect of the attack budget ϵ, we uniformly select 10 values from the interval [0, 1] and set ϵ to those values when attacking. Part of the results is reported in Table 4. Consistent with previous works, the two tables show that the mere addition of a perceptible perturbation ($\epsilon = 0.011$) can destroy the classifiers. We illustrate an example in Figure 6.

	$\epsilon = 0$	$\epsilon = 0.011$	$\epsilon = 0.033$	$\epsilon = 0.055$	$\epsilon = 0.077$	$\epsilon = 0.1$
AlexNet	65.9%	14.3%	3.6%	2.6%	2.5%	2.7%
VGG11	74.8%	11.2%	4%	4.2%	5.3%	6.5%
VGG13	75.5%	6.3%	2.8%	4.7%	6.5%	9.5%
ResNet18	79.2%	3.2%	2.5%	5.9%	9.6%	13.3%
ResNet50	80.7%	17.4%	25.7%	34.9%	36.1%	33.2%
ViT_B_32	82.5%	32.6%	14.7%	12%	10.2%	9.3%
ViT_B_16	89.3%	26.9%	16.4%	14.1%	13.4%	13.5%
Swin_T	87.2%	30.8%	18.5%	15.5%	14.8%	14.2%
Swin_S	90%	32.8%	19.5%	18%	17.5%	17.3%
Swin_B	90.8%	36.9%	24%	22.3%	21.2%	21.2%

Table 4: Robustness. This table shows the test accuracy under different degrees of attacks which can reflect the robustness of the model.

Another interesting fact is related to how the performance varies as ϵ grows. All transformer models seem to be much more robust than CNNs when encountering a tiny attack, which we can see in the third column of Table 4. However, the performance of VGG and ResNet increases when ϵ growing while transformer models become fragile. It's then natural asking whether this phenomenon reveals some inherent property of the models. One conjecture is that transformer models produce more smooth predictions than CNNs, which is consistent with the claim stated in earlier works. Therefore, the degrading in accuracy indicates the classifier manages to learn the underlying data distribution. We test this guess using a simple experiment. Instead of the FSGM, we generate the perturbation randomly from the normalized Gaussian distribution multiplied by $\epsilon = 0.011$. We repeat the attack 10 times for each sample and evaluate the performance of the models on them. The results is shown in Table 1 which, however, is not sufficient to support this conjecture. This table also shows some weird facts when combining it with Table 4. Therefore, we need more carefully conducted experiments to further explore the principle behind. Meanwhile, the increasing accuracy of CNNs can't be explained in this way.

Finally, we explore whether adding the perturbation budget ϵ can efficiently strengthen the adversarial capability. We uniformly select 20 values from [0, 10] as the choices of ϵ. The test accuracy is plotted in Figure 5. It indicates a small perturbation is enough to weaken the classifier and a much larger budget can not bring further improvement.

4.5 Comparison Under Different Attacks and Settings

Finally, we conduct a comprehensive experiment using all three attack methods stated in previous sections[1]. The results are displayed in Table 5. Each model is listed with three values representing its robustness under the FGSM, PGD, and Gaussian noise in every experiment setting. As shown in the column names, we design three experiment settings in total. Firstly, we test the robustness of the original pre-trained models on our CIFAR-10. Since the pre-trained modes are trained on ImageNet-1k, we use a standard transfer learning trick here. Specifically, we create a mapping $\mathcal{F}$ from the classes of the ImageNet-1k dataset to classes of CIFAR-10. We create $\mathcal{F}$ by first running the model on CIFAR-10 and recording the predicted classes of each sample. Then we set the value of $\mathcal{F}(x)$ as the class in CIFAR-10 to which most samples classified into class x truly belong. In this way, we can test the robustness of original pre-trained models and the results are shown in the first column of Table 5.

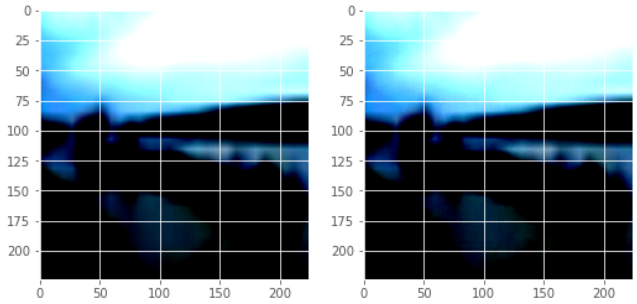
Then we follow the standard paradigms in transfer learning and adversarial training to design the rest two experiment settings. We change the final linear layer of each model to finetune it on CIFAR-10. We train the model in 5 iterations and again test the robustness which is displayed in the second column of Table 5. Finally, we adopt the adversarial training on those finetuned models using the simple FGSM for 5 iterations. The robustness of models after adversarial training are listed in the last column of Table 5.

4.6 Reasoning the Robustness

Although it needs to be further verified, Table 5 provides massive information to us. Firstly, it is easy to find that transformer models own better robustness than CNN-based models. This is consistent with observations stated in some previous works, where the researchers guess the reason might be that features learned by transformers contain less low-level information and benefit adversarial robustness. They are also less sensitive to high-frequency adversarial perturbations. We see taht the increase in the proportion of transformer blocks leads to better robustness when the model consists of both transformer and CNN blocks.

We can also analyze the effect of three adversarial attacks. Obviously, PGD is the most effective to degrade the performance of a model while adding the random Gaussian noise can hardly fool it. The latter is somewhat surprising to us since it means that the prediction of the models still owns some smoothness, which is contrary to many early works. Hence we need more investigations to figure out the reason causing this difference, which can be a future direction of this work. Another insight from here is that pre-training on large datasets for transformers does not significantly improve the robustness against adversarial attacks, while adversarial training does this job very well. Although

[1]Code can be checked at: https://github.com/Jessogreat/Adver-Project.



(a) True Ship (b) False Airplane

Figure 6: 6a is a picture of a ship from the test set after preprocessing and 6b is the same picture added by a perceptible perturbation. They do not appear to be any different, but the classifier predicts the latter as an airplane while correctly identifying the former.

	Pre-trained models(without fine-tune)	Pre-trained models(with fine-tune)	Adversarial trained models
AlexNet	40.13%/32.69%/99.35%	79.21%/75.99%/100%	100%/100%/100%
VGG11	23.05%/15.27%/98.44%	66.22%/58.24%/100%	100%/100%/100%
VGG13	25.16%/20.81%/96.89%	56.59%/45.66%/100%	100%/100%/100%
ResNet18	26.27%/19.62%/98.73%	47.68%/34.47%99.76%	96%/96%/100%
ResNet50	41.34%/31.01%/97.49%	43.78%/13.18%99.75%	90.38%/90.38%/100%
Swin-B	47.38%/37.81%/99.77%	88.33%/86.78%/100%	96.68%/96.46%/100%
Swin-S	47.37%/38.67%/98.16%	83.3%/81.29%/100%	91.86%/91.35%/99.75%
Swin-T	40.9%/32.17%/99.25%	80.11%/78.43%/100%	98.08%/98.08%/100%
ViT-B-16	51.15%/44.95%/99.31%	83.87%/83.13%/100%	92.7%/92.36%/100%
ViT-B-32	55.56%/52.25%/100%	91.26%/90.75%/100%	95%/95%/100%

Table 5: The rest accuracy of models after attacks, $\epsilon = 0.003$.

there are still many more insights can be gained from the results, we only highlight the above for the sake of simplicity.

4.7 Test for GGP Measure

In order to test our new measure for robustness, we do the following two experiments for p-good-pattern1 and p-good-pattern2.

4.7.1 p-good-pattern1

The first experiment is running the algorithm p-good-pattern1 for $k = 3$ on the 10 models after finetune. Results are shown in Table 3.

Compare to Table 5, it is clear that the figures are consistent for Swin and ResNet families. So our new measure is useful.

4.7.2 p-good-pattern2

The second experiment we want to explore how could we improve the robustness of a binary classification network, widening or deepening? The original network is a fully connected neural network with 2 hidden layers and 5 neurons in each hidden layer. The second network is adding 2 hidden layers based on the original network. The third network is doubling the number of neurons in the hidden layers of the original network. For each network, we run p-good-pattern2 100 times and according to the average performance, we find that 55% of times increase the wide structure is more robust than the deep structure which is consistent with some previous empirical results[29].

5 Conclusions

In this work, we conduct massive experiments to rethink the adversarial robustness of deep learning models on image classification. We focus on the comparison between CNN-based models and transformer models, the effect of different attacks, and training settings. Some of the results are consistent with previous works while the others are different and require further exploration. We also define two algorithms to predict the robustness of models which empirically works well through the experiments.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[2] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

[3] Nicholas Carlini and David Wagner. Defensive distillation is not robust to adversarial examples. *arXiv preprint arXiv:1607.04311*, 2016.

[4] 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.

[5] 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.

[6] Wuyang Chen, Xinyu Gong, and Zhangyang Wang. Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective. *arXiv preprint arXiv:2102.11535*, 2021.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[10] Stanislav Fort, Jie Ren, and Balaji Lakshminarayanan. Exploring the limits of out-of-distribution detection. *Advances in Neural Information Processing Systems*, 34:7068–7081, 2021.

[11] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

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

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[17] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018.

[18] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[19] Yao Li, Minhao Cheng, Cho-Jui Hsieh, and Thomas CM Lee. A review of adversarial attack and defense for classification methods. *The American Statistician*, pages 1–17, 2022.

[20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[21] 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.

[22] Francesco Pinto, Philip Torr, and Puneet K Dokania. Are vision transformers always more robust than convolutional neural networks? 2021.

[23] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[24] Eduardo D Sontag et al. Vc dimension of neural networks. *NATO ASI Series F Computer and Systems Sciences*, 168:69–96, 1998.

[25] 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.

[26] Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer, 2015.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[28] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[29] Boxi Wu, Jinghui Chen, Deng Cai, Xiaofei He, and Quanquan Gu. Do wider neural networks really help adversarial robustness? *Advances in Neural Information Processing Systems*, 34:7054–7067, 2021.

[30] Huan Xiong, Lei Huang, Mengyang Yu, Li Liu, Fan Zhu, and Ling Shao. On the number of linear regions of convolutional neural networks. In *International Conference on Machine Learning*, pages 10514–10523. PMLR, 2020.

[31] Chongzhi Zhang, Mingyuan Zhang, Shanghang Zhang, Daisheng Jin, Qiang Zhou, Zhongang Cai, Haiyu Zhao, Xianglong Liu, and Ziwei Liu. Delving deep into the generalization of vision transformers under distribution shifts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7277–7286, 2022.