Defending Against Adversarial Attacks Using Random Forest

Yifan Ding[1] Liqiang Wang[1] Huan Zhang[2] Jinfeng Yi[3] Deliang Fan[1] Boqing Gong[4]
[1]University of Central Florida
yf.ding@knights.ucf.edu, lwang@cs.ucf.edu, dfan@ucf.edu
[2]University of California, Los Angeles [3]JD AI Research [4]Google Research
huanzhang@ucla.edu jinfengyi.ustc@gmail.com boqinggo@outlook.com

Abstract

As deep neural networks (DNNs) have become increasingly important and popular, the robustness of DNNs is the key to the safety of both the Internet and physical world. Unfortunately, some recent studies show that adversarial examples, which are hard to be distinguished from real examples, can easily fool DNNs and manipulate their predictions. Upon observing that adversarial examples are mostly generated by gradient-based methods, in this paper, we first propose to use a simple yet very effective non-differentiable hybrid model that combines DNNs and random forests, rather than hide gradients from attackers, to defend against the attacks. Our experiments show that our model can successfully and completely defend the white-box attacks, has a lower transferability, and is quite resistant to three representative types of black-box attacks; while at the same time, our model achieves similar classification accuracy as the original DNNs. Finally, we investigate and suggest a criterion to define where to grow random forests in DNNs.

1. Introduction

Despite being remarkably effective in solving many real-world problems such as the perception of images, videos, and texts, Deep neural networks (DNNs) are surprisingly vulnerable to adversarial attacks: one can easily fool the networks and manipulate their predictions by generating adversarial examples that are only slightly different from the real examples for which the networks usually give rise to correct predictions [29]. While at the same time, DNNs are widely applied to many critical real-life applications, such as self-driving cars, robotics, and Internet of Things, it is vital to improve their robustness against adversarial attacks.

In the state-of-the-art adversarial attacking methods, gradients are the key ingredient to perturb a normal example to an adversarial one. These methods include white-box attacks [14, 14], which assume that attackers have full knowledge of the DNNs being attacked, including the architecture and weights of the DNNs and, even the training data and gradients during training. In like manner, black-box attacking algorithms [28, 21] allow attackers access to nothing but the DNNs' outputs (*e.g.* classification prediction probabilities) which are usually used to estimate the gradients. Thus recent defense strategies focus on preventing the attackers from inferring the gradients. As summarized in [1], defense methods often use shattered gradients [6], design stochastic gradients, or intentionally vanishing/exploding gradients [31]. But since these methods are in the nutshell of training with back-propagation, it essentially opens the door to attackers. The gradient-based attack could still succeed as long as they find a way to obtain or approximate the correct gradients, which seems feasible in most cases [1].

In this paper, we open a novel direction for defending against gradient-based attacking models by introducing non-differentiable algorithms into DNNs. We propose a simple, intuitive yet effective method to transform a fragile DNN classifier into a more robust one. We present a hybrid approach that we transform the last few layers of DNNs classifiers into random forests that are non-differentiable at all so that the gradients cannot been approximated. Our proposed technique is very flexible for easily marrying the strength of the representation power of DNNs as feature extractor and the security property thanks to the non-differentiability of random forests.

Extensive experiments on several DNN classificaton models including AlexNet [18], VGG-16 [30], and MNIST-net [7] show the effectiveness of our method against some popular attacking algorithms such as C&W model [7] for white-box attacks and Zeroth Order Optimization (ZOO) [8] for black-box attacks. White-box attack methods fail to attack our hybrid deep model because the last few layers are not differentiable at all. Furthermore, most previous defense methods suffer from failing to defend the transferability of adversarial examples. We show that our approach using random forests leverages the problem. On the other hand, black-box attacks also find it hard to attack such hybrid model because the final prediction is not a single continuous-valued probabilistic output but an ensemble output. Details of experiments will be discussed latter.

We further provide technical study of our approach. We investigate the performance of building the random forests from different layers of the DNNs. We find that the relative ℓ_2 distance between the real examples and the corresponded adversarial examples serves as good indicator to determine the depth of DNN layers that the random forests best grow upon. With careful examination and ablative study, experiments show our models not only defend against the attacks successfully but also achieve a high classification accuracy on the original test instances.

Below we summarize the key contributions:

- We present a simple, flexible, and intuitive technique to enhance the security of DNN classifiers against adversarial attacks.
- We propose a novel direction for defending from gradient-based attacks. We utilize random forests to transform pure DNNs into non-differentiable classifiers thus prevent the gradient from being inferred which is the strategy that state-of-the-art attacking models typically rely on.
- Extensive experiments show the effectiveness of our model against some strong attackers in both white-box and black-box literature.

2. Related work

Since the concept of adversarial samples comes up, many attack and defense methods have been proposed. In this section, we describe recent developments related to this topic, including both adversarial attacks and adversarial defense approaches.

2.1. Adversarial attacks

White-box attacks. White-box means attackers have the full access of the architecture, parameters and weights of the model [32, 7, 14]. One of the first methods is proposed in [32], known as box-constrained L-BFGS, which minimizes the additive perturbation based on the classification constrains. Then, Goodfellow *et al.* proposed an approach called Fast Gradient Sign Method (FGSM) [14] to conquer the inefficiency. Later, Basic Iterative Method (BIM) method [19] is introduced to add perturbations iteratively. In 2017, a very strong attack [7] (*a.k.a.* C&W) method is proposed to find the minimal L_0, L_2, and L_∞ distance. There are also generative network based attacking methods. For example, [2] generates adversarial samples through generative adversarial networks [13]. It is noteworthy that Kantchelian proposes an attack method on tree ensemble classifiers [16]. However, the approach handles regression trees while we use classification trees, while the authors also propose that the attack of tree classifiers can be easily solved with adversarial training.

Black-box attacks. Compared to white-box attack methods that request the target neural networks to be differentiable, black-box attacks are introduced to deal with non-differentiable systems or systems whose parameters and weights cannot be reached. Zeroth Order Optimization (ZOO) [8] can directly estimate the gradients of the target network through zeroth order stochastic coordinate descent. Papernot *et al.* propose [28] to achieve decision boundaries learning based on transfer attack. And Liu *et al.* [21] use an ensemble of several pre-trained models as the source model to generate adversarial examples even when no query of probes are allowed.

2.2. Adversarial defenses

Defenses based on gradient masking. Gradient masking is one of the most popular defense methods that intentionally or unintentionally mask the gradient that is needed for computing perturbations by most white-box attackers. Buckman encodes input images using thermometer encoding to enable discrete gradient that cannot be attacked directly [6]. While Guo applies transformation to the inputs to shatter the gradients [15]. By adding noise to the logit outputs of neural networks, Nguyen introduces the masking based defense against C&W attacks [27].

Defenses through adversarial training. Adversarial training is one of the most straightforward and efficient strategies when defending against the adversarial attacks [32]. However, adversarial training only improves the robustness of some specific attacks [24, 4]. For some strong attack methods like [7], it is very hard to gain robustness via adversarial training because of large search space. It is observed that by adding noise before starting the optimizer for the attack [22], the over-fitting towards specific attacks can be reduced and ensembling adversarial samples generated by different models also improves robustness [33].

Other defenses. There are also other defense methods that fall into none of the two categories above. Metzen defends against attacks through refusing classification by detecting whether there are signs of tampering of the input [23]. Goodfellow uses shallow RBF that is robust to adversarial samples but has a much lower accuracy on the clean samples [14]. Some other methods try to apply preprocessing techniques towards the input to denoise before the classifier such as JPEG compression [10] and median filter [34].

3. Our Approach

To adversarially attack deep neural networks (DNNs), the most effective existing methods are almost gradients-based (*e.g.* [7, 14]) probably because the DNNs are usually trained by (stochastic) gradient descent. As a result, to defend DNNs against such attacks, gradient masking has become the main idea behind recent defense methods [6]. It means that although the gradients exist in the deep models, they could be hidden from the attackers. However, such defenses fail once attackers find some ways to approximate the gradients [1]. We first introduce our methdology, then summarize the the two streams of attack models we used to examine our method.

3.1. Model

We design a defense approach with a completely different strategy instead. By incorporating a standard CNN classifier with a random forest into a robust classifier, we integrate the merit of neural networks in extracting abstract and informative features and the advantage of non-gradient based classifiers. In practice, our hybrid model is achieved through a two-stage training procedure. We first train a DNN classifier, with back-propagation from cross-entropy loss, then discard the last few layers in the DNNs and replace them with a random forest (*e.g. built with classification trees*) to work as a non-differentiable classifier. Figure 1 illustrates our approach. It could be understood as attaching random forests upon a well-trained DNN feature extractor. Thanks to the powerful feature extractor and the expressive capabilities of random forests, our approach achieves almost the same accuracy as the original DNN classifier on the testing sets. Meanwhile, it disable white-box attacking methods to compute gradients through such a hybrid model. We show in experiments that our hybrid model is also robust to the transfer attacks which we proposes as the ultra white-box attack.

In formal, we introduce our hybrid model. To re-purpose pre-trained DNNs as feature extractors [12], a straightforward implementation is to use output from the penultimate layer. However, in practice directly stacking a random forest after the penultimate layer yields poor performance. To address this problem, we propose a passive strategy to identify the proper layer of the DNNs from which we grow the random forests. We find that after adding a small perturbation to the input, there is a difference for the relative ℓ_2 distance between the original samples and the corresponding adversarial samples generated by the C&W attack after each activation, which means that different layers in the DNNs have different amplifying ability towards the perturbation. The average relative ℓ_2 distance of the kth layer is calculated based on:

$$D_k = \frac{1}{|M|} \sum_{i=1}^{M} \frac{\|\boldsymbol{a}_i^{(k)} - \tilde{\boldsymbol{a}}_i^{(k)}\|_2}{\|\boldsymbol{a}_i^{(k)}\|_2} \tag{1}$$

where M denotes the number of images, $\boldsymbol{a}_i^{(k)}$ and $\tilde{\boldsymbol{a}}_i^{(k)}$ denote the activation of the ith original input x_i and the corresponding adversarial example $\tilde{x}_i$ at the kth layer of DNN $\mathcal{F}$, respectively.

Using ℓ_2 as the criterion, we keep as many layers as possible at the beginning from the input layer until we reach the one that causes significantly larger distortion than the other layers. Our experiments in Section 4 demonstrate that this criterion is effective, allowing the resulting hybrid model to achieve a high classification accuracy and meanwhile to be robust against the transfer attacks or black-box attacks.

Afterwards, we use the output of the $(k-1)$th layer of the DNN $\mathcal{F}$, while k denotes the layer with the largest relative ℓ_2 distance between the original samples and the adversarial samples, as the input of the random forests. We build our random forests with classification trees by following the approach in [9]. For each split in each tree, we compute the split score S through:

$$S = G(L) + G(R) \tag{2}$$

where L and R denote the left and right per-class probability count vector, respectively. The idea is to calculate the Gini impurity to give the lowest split score S. The Gini impurity is calculated by:

$$G(C) = 1 - \sum_{j}^{J} (\frac{C_j}{N_c})^2, N_c = \sum_{j}^{J} C_j \tag{3}$$

where there are J classes and C_j denotes the number of samples classified into the jth class. The problem can be solved through Hoeffding tree [11]. Let P_h denote the output of the hth classification tree and suppose there are H trees in the random forests. The final predictions of sample x in our hybrid model becomes the ensemble of the classification trees:

$$P_f(j|x) = \frac{1}{H} \sum_{h=1}^{H} P_h(j|\mathcal{F}_{k-1}(x)) \tag{4}$$

where $\mathcal{F}_{k-1}$ denotes the output of the $(k-1)$th layer in the DNN $\mathcal{F}$, j is the j^{th} class and H is the number of trees in the forest.

Next, we introduce the two categories of white-box and black-box attacks we mainly focus against in experiments.

3.2. Defending against white-box attacks and transfer attacks

White-box attacks are the most powerful adversarial attacks. However, by stacking the random forests, the hybrid model becomes non-differentiable and the conventional white-box attackers cannot get meaningful results by attacking the hybrid model because there is no way of back-propagating gradients through random forests. Besides, to further demonstrate the robustness of the hybrid model, we also propose a attack setting named *ultra white-box attack*. In the ultra white-box attack setting, not only our our hybrid

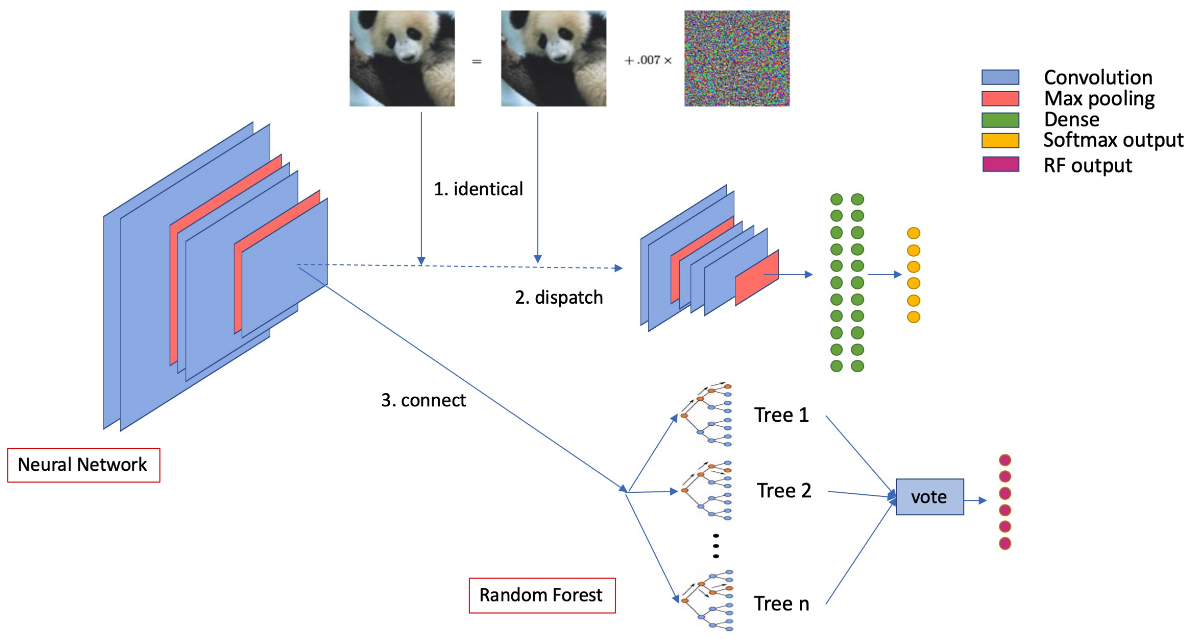

Figure 1. An overview of our approach. We build a hybrid model through three steps: First we identify the proper layer of growing the random forest by locating the kth layer in the DNN which has the biggest amplifying ability with Equation 1. Then we dispatch the DNN from the kth layer and connect a random forest to imitate the classification ability of the DNN and dissolve the gradients to avoid being attacked by the adversarial attackers.

model but also the original DNNs upon which our model is built are released to attackers. Therefore, the ultra-white-box attackers will be able to generate adversarial examples for our end-to-end differentiable DNNs through gradient methods and transfer the attacking results to our hybrid model. The setting is based on the observations that adversarial examples have remarkable transferabilities [21, 32]: given two neural networks trained for solving the same problem, the adversarial examples generated for one neural network can also successfully attack the other.

In the white-box attacks experiments, a strong white-box attack, C&W attack, is used to conduct the experiments. Specifically, we use their most efficient ℓ_2 attack.

3.3. Defending against black-box attacks

We also discuss the robustness of our model against the black-box attacks since black-box attackers should be able to attack any model without access to the gradients. As there are countless number of black-box attack methods and we would not be able to test all of them, we therefore categorize the existing black-box attacks into the following three representative groups. Then in each group, a representative and latest method is selected to test the effectiveness of our method.

Gradient estimation One of the most efficient category is gradient estimation based methods that use various techniques to estimate the unknown gradients of the target model [8, 25, 3]. We choose ZOO [8] out of this category in our experiments as it is one of the strongest black-box attack methods. ZOO achieves almost the same success rate as white-box attacks by estimating the gradients of the target network through the zeroth-order stochastic coordinate descent.It is worth pointing out that there is no ground-truth gradients at all in our hybrid model. However, ZOO still tries to attack it by the zeroth-order gradients. As a result, our approach is experimentally shown hard to be attacked by ZOO.

Decision-based attacks The decision-based attacks are black-box methods that solely rely on the final decision of the model [26, 4]. As a representative decision-based method, Boundary Attack [5] achieves a comparable attack performance with one of the best gradient-based attacks [7] in both targeted and untargeted scenarios which literately modifies a random sample from the target class to be visually similar to the original sample.Boundary Attack is hard to be defended because one of the pre-requisites of the algorithm is to maintain the target classification. Therefore, the attack success rate would always be 100% although the generated adversarial samples might be quite different from the original samples. During the experiment, we use the ℓ_2 distance to evaluate the performance of the attack and defense.

Table 1. The average relative ℓ_2 distance between adversarial and original samples

Activation	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
MNISTnet	0.03	0.05	0.08	0.18	0.22	0.30	**0.60**	-	-	-	-	-	-	-	-
AlexNet	0.01	0.03	0.08	0.17	0.17	**0.43**	0.50	0.62	-	-	-	-	-	-	-
VGG-16	0.01	0.02	0.03	0.07	0.12	0.14	**0.29**	0.43	0.45	0.51	0.53	0.48	0.53	0.52	0.65

Training a substitute model While knowing the number of target classes, the attackers could train a substitute DNN to mimic the behavior of the model. Ideally, the substitute DNNs shares the same decision boundaries as the target model and the adversarial samples generated by the substitute DNNs would be able to fail the target network as well. We use the recent Practical Black-Box Attack method proposed in [28] to test our defending approach against this type of methods.

4. Experiments

Our experiments are conducted on CIFAR-10 [17] and MNIST [20] dataset, following the previous experiment protocols [7, 8]. We pre-process all the images by re-scaling the pixel values to the range [-0.5, 0.5]. For all of the experiments on CIFAR-10, we use two popular neural networks VGG-16 [30] and AlexNet [18] as our backbone networks. For MNIST, we use the architecture proposed in the C&W attack [7], *i.e.*, a 7-layer convolutional neural network and we denote it as *MNISTnet* in our following experiments. In the testing stage, we also follow the C&W attack and use the first 1000 originally correctly classified images from the test set to evaluate the success rate of the attack.

We consider the untargeted attack for the substitute model method [28] and targeted attack for the other methods. In the targeted attack, we use all the 9 classes in turn except for the ground truth class as the target class for each image and calculate the average attack success rate over all the target classes. For the ZOO and ultra white-box attacks, margin κ [7] is set to zero. In our experiments, we use the attack success rate (*i.e.* ASR) to measure the effectiveness of different attacks (or, our defense). ASR is calculated as the percentage of the adversarial examples that are successfully classified to the target classes by the attacks out of all examples tested. Additionally, we also include ℓ_2 distance in most of our experiments [7, 8], which is calculated using the mean squared distance between the original images and the adversarial images. We find that a random forest of 300 trees and 10,000 nodes gives rise to the best trade-off between the classification accuracy and the defense performance on CIFAR-10. However, to save computational cost, we use 300 trees and 10,000 nodes for AlexNet and 300 trees and 1,000 nodes for VGG-16 in our experiments. For AlexNet, we keep the first five layers as the feature extractor and, for VGG-16, we grow the forest from the 8^{th} layer. For the neural network applied to MNIST, we use 10 trees and 1,000 nodes taking as input the activations of the 6^{th} layer.

4.1. Finding the proper layers to grow the random forests

To find the most suitable layer to grow the random forests, we use the relative ℓ_2 distance (Equation 1) defined in Section 3.1 as our criterion. In this experiment, we randomly select 1,000 originally correctly classified images out of the validation sets to evaluate the relative distortion between the original samples $\{x\}$ and the adversarial samples $\{\tilde{x}\}$ generated by the C&W attack. Since the amplifying ability of each activation is different, the relative growth of ℓ_2 distance between each two activations changes. To our surprise, the relative ℓ_2 distance grows slowly in both the first layers and the last few layers for AlexNet and VGG-16, while on the contrary, faster in some intermediate layers as shown in Table 1.

Recall that our original intention of building the hybrid model is to take advantage of the strength of the representation power of DNNs as feature extractor, the experimental results declare the fact that each layer in the neural networks plays a different role (*e.g.* the bottom layers extract low level features and the top layers in charge of the classification). Then the different role of each layer result in the different ability of the layer to manipulate the adversarial distance. It turns out the intermediate layers in AlexNet and VGG-16 have best of the ability to generate the adversarials. Hence we are safe to replace the top layers which are responsible to classification with the random forest which has no gradient. The distance also grows differently in different networks (MNISTnet and others), which also request special dispatch point in distinctive networks.

We therefore choose to grow the random forests after the one with the largest relative growth which is the 6th for MNISTnet, 5th for AlexNet and 8th for VGG-16 after trade-off between the defense performance and the classification accuracy. The selection turns out to be effective in the following experiments. The ablation study of the ultra white-box performance while growing random forests from different layers is shown in Section 5.3.

4.2. White-box attacks and transfer attacks

We use the C&W method as the white-box attack methods following the setting of the original paper [7]. As mentioned before, directly attacking our hybrid model gives 0% attack success rate. And our model also has a very low transferability in the ultra white-box setting. Since there's no counterpart in the original DNNs for comparison with our ultra white-box setting, we also test the regular transfer attacks in which we generate adversarial samples on one network(*i.e.* AlexNet) and test the accuracy of the generated adversarial samples on another network(*i.e.* VGG-16 and our AlexNet hybrid model) The results are shown in Table 2. For this and the following tables, we denote accuracy as 'acc' and attack success rate as 'ASR'.

Table 2. Results of the white-box and transfer attack

		Acc%	ASR%	Ultra%
AlexNet	dnn	83.4	100	100
	ours	82.7	**0.0**	**4.8**
VGG-16	dnn	93.5	100	100
	ours	90.6	**0.0**	**9.7**
MNISTnet	dnn	99.2	100	100
	ours	98.4	**0.0**	**4.4**

Compared to the 100% attack success rate achieved in the original DNNs, the C&W attack barely succeeds in one single attack to our hybrid models as shown in the 4th column of Table 2. Besides, testing the transferability on the same DNNs is meaningless, we just keep it as 100% for comparison. After making it a hybrid model, we achieves a transfer attack success rate of 4.8%,9.7% and 4.4% on AlexNet, VGG-16 and MNIST, respectively. Thus we can safely claim that even if we release the architecture and weights of both our original neural network and the hybrid network, our approach can still successfully defend against the current strongest white-box attack.

4.3. Black-box attack: ZOO

Since the comparison is already significant enough, in the experiment of the Zeroth Order Optimization (ZOO) attack [8] we only run 1,500 iterations to attack the original neural networks but 15,000 iterations to attack our hybrid model to save computational time. For MNIST, we did not re-produce the experiment but directly include the result reported in their paper as the same network is used. The results are shown in Table 3.

Table 3. Results of defending against the ZOO attack [8]

Network		Acc%	ASR%	ℓ_2 distance
AlexNet	dnn	83.2	89.7	8.79
	ours	82.7	**5.4**	**37.2**
VGG-16	dnn	93.5	86.8	9.89
	ours	90.6	**3.6**	**38.1**
MNISTnet	dnn	99.2	98.9	2.0
	ours	98.2	**0.9**	**185.2**

In terms of the classification accuracy, our hybrid models are on a par with the original neural networks; the absolute decreases are only 0.5% 2.9% and 1% for AlexNet VGG-16 and MNISTnet, respectively. In terms of the defense performance, the hybrid models are much better than the original neural networks. For AlexNet, the attack success rate (*i.e.* ASR, the lower the better) decreases from 89.7% to 5.4% with an average ℓ_2 distance between the original input examples and the adversarial examples increasing from 8.79 to 37.2 (the higher the better from the defense perspective). The same observation also happens to VGG-16 and MNISTnet, where the ASR decreases from 86.8% to 3.6% and from 98.9% to 0.9% with an average distance increasing from 9.89 to 38.1 and from 2.0 to 185.2, respectively. Thanks to our hybrid defense method that has about the same classification accuracy as DNNs, we can safely claim that the strong black-box attack method, ZOO, fails.

4.4. Black-box attack: Decision-based attack

For decision-based attack [5], we test 100 samples from the testset that have been correctly classified by our model. For each example with label l, we set the target label as $(l + 1) \mod 10$. For each example, we set the maximum number of steps to be 100, where very few steps are already sufficient to demonstrate the difference between our model and the original. In each step, the length of the total perturbation δ and the length of the step ϵ towards the original input are initially set to 0.1 and 1.0, respectively, as suggested by [5]. In each δ step, 5 orthogonal perturbations are generated compared with 10 in the original paper. Our results are shown in Table 4.

Table 4. Results attacked by Decision Based Attack Model [5]

Network		Acc%	ℓ_2 dist	Time (s)
AlexNet	dnn	83.2	4.58	1.84
	ours	82.7	**5.87**	**61.2**
VGG-16	dnn	93.5	5.66	2.57
	ours	90.6	**5.74**	**46.0**
MNISTnet	dnn	99.2	16.6	0.86
	ours	99.2	11.1	**5499.6**

Since the decision-based method follows the rule that the target classification must been maintained, it makes no sense to compare the attack success rate. Instead, we compare the ℓ_2 distance of the generated adversarial samples through attacking the DNNs and our hybrid models and the attack time. The longer the attack time and the ℓ_2 distance, the poor the attack. Our results are shown on Table 4.

4.5. Black-box attack: Practical Substitute Model

Recall that the substitute model in [28] only conducts experiments on MNIST. We expand their experiment setting to CIFAR-10 by re-adjusting the parameters. To compare adversarial attacks on CIFAR-10, we set learn rate = 0.001. In addition, we increase the number of seed samples from 150 to 1500. The substitute model is also enhanced, specifically, we use VGG-16 as the substitute model for AlexNet and AlexNet as the substitute model for VGG-16. We use learning rates 0.001 and 0.0001 for AlexNet and VGG-16, respectively. We train the model with 5 data augmentations, each for 50 epochs, which is 1 data augmentation less than the original experiments in [28] since the number of training samples grow exponentially with each data augmentation process and the seed training set is already 10 times larger compared to the original experiments. And for the experiment of MNIST, we use the same setting as the original paper, which means ϵ is set to 0.4. The results are shown in Table 5. "Acc" and "Acc:sub" denote the accuracy of the original and the substitute model respectively.

Table 5. Results attacked by Practical Substitute Model [28]

Network	ϵ		Acc%	Acc:sub%	ASR%
AlexNet	0.03	dnn	83.2	45.6	**19.2**
		ours	82.5	44.1	19.8
	0.1	dnn	83.2	45.5	47.2
		ours	82.5	43.0	**44.9**
VGG-16	0.03	dnn	93.5	42.8	13.1
		ours	90.5	44.4	**12.5**
	0.1	dnn	93.5	42.1	70.3
		ours	90.5	40.1	**69.7**
M-net	0.4	dnn	99.2	70.1	45.7
		ours	98.7	78.1	**43.1**

In Table 5, ASR denotes the performance of our defense of untargetd attacks, as only untargeted attacks are involved in [28]. It represents that the generated adversarial samples cannot be correctly classified by our hybrid model after the attack. It can be found from the experiment that it is very hard to train a substitute model on natural images especially when the model is complex. Therefore we also test when $\epsilon = 0.1$ where ϵ is the perturbation magnitude preset for the FGSM attack [14] which in most cases, ϵ is set to 0.031.

Among the three categories of black-box attacks, our hybrid model also performs well qualified. It either decreases attack success rate or increase ℓ_2 distance, or attack time. To the best of our knowledge, there are very few research that have done a wholesome attack towards a single defense method from white-box to black-box attacks, from directly attacks to transfer attacks, and gain a satisfying performance on almost all of the experiments.

5. Technical study

In this section, we perform technical studies to quantify the details of our designs including the effect of parameters involved in our model, the influence of adversarial training, and the difference of the depths of DNN layers that the random forests build on. Specifically, we re-train our approach using different parameters in the random forests and DNNs, and demonstrate the results after adversarial training.

5.1. Effectiveness of adversarial training

To investigate whether adversarial training helps, we implement two kinds of experiments to verify this issue, as shown in Table 6. In the first experiment, we train the original DNNs with the adversarial samples generated from C&W attack and PGD attack [22], respectively, and then generate two new hybrid models. Then we test the two new hybrid models using the ultra white-box attack. It can be found from Table 6 that adversarial training on original DNNs does not affect the attack success rate too much but the average distortion increases.

Table 6. Results of adversarial training

Adv train	Acc%	ASR%	Distortion
None	83.4	32.8	0.44
Hybrid: C&W	81.1	33.5	**0.69**
Hybrid: PGD	79.4	33.9	**0.52**
Random forest	83.2	**16.2**	0.44

In the second experiment, for a hybrid model, we tune its random forest by the adversarial samples generated from C&W attack. Then we test the tuned hybrid model again using the ultra white-box attack. The attack success rate drops since the random forests already learns the adversarial samples. To better observe the experiment difference, only the last few fully connected layers are deprived and replaced by random forests. The experiment is conducted on AlexNet and all the attack success rates are calculated based on the ultra white-box settings.

5.2. Parameters in the random forests

We only tune two parameters of the random forests, *i.e.,* the number of trees and the number of max nodes in each tree. We test the number of trees between 10 to 300 and the number of max nodes between 100 and 10000. The parameters are tested using grid search and the results are reported with respect to the performance of the ultra white-box task. Table 7 shows the results on AlexNet where the substitution is conducted from the 5^{th} layer and VGG-16 where the sub-

Table 7. Results of the ultra white-box attack with different number of trees and nodes

Trees	Nodes	AlexNet				VGG-16				MNISTnet			
		Acc	ASR	Hyb acc	Trans	Acc	ASR	Hyb acc	Trans	Acc	ASR	Hyb acc	Trans
10	100			74.2	10.7			83.1	14.2			94.0	9.1
10	1000			77.7	8.2			86.4	13.0			97.1	8.3
10	10000			77.8	8.9			86.8	12.3			97.8	7.6
100	100			80.5	6.2			87.9	11.8			96.4	7.1
100	1000	83.4	100	81.9	6.2	93.5	100	89.8	11.6	99.2	100	97.8	5.6
100	10000			82.6	5.8			90.4	10.4			98.2	4.5
300	100			80.8	6.6			88.3	10.7			96.3	6.5
300	1000			82.3	5.3			90.1	11.1			97.7	5.1
300	10000			82.7	**4.8**			90.6	**9.7**			98.4	**4.4**

Table 8. Results when starting the substitution from the second-last and the best activation

Trees	Nodes	AlexNet				VGG-16			
		From $7th$ activation		From $6th$ activation		From $14th$ activation		From $7th$ activation	
		Hybrid acc	Trans	Hybrid acc	Trans	Hybrid acc	Trans	Hybrid acc	Trans
10	100	81.4	**36.1**	74.2	10.7	93.4	**48.6**	69.2	9.0
100	1000	83.3	37.1	81.9	6.2	93.5	55.2	81.8	6.6
300	10000	83.3	36.9	82.7	**4.8**	93.6	57.3	84.4	**5.6**

Table 9. Results when starting the substitution from different layers (AlexNet)

Trees	Nodes	After the conv		After activation		After Pooling		After Batchnorm	
		Hybrid acc	Trans	Hybrid acc	Trans	Hybrid acc	Trans	Hybrid acc	Trans
10	100	72.8	13.0	74.2	10.7	75.9	17.7	76.0	16.6
100	1000	81.8	**9.8**	81.9	6.2	82.1	**10.7**	82.2	11.2
300	10000	82.6	10.6	82.7	**4.8**	83.0	10.9	82.9	**10.9**

stitutition is conducted from the 8^{th} layer. The activation layers are numbered from the bottom layers to top layers.

It can be found that the more trees and nodes we have, the higher the performance of our hybrid network achieves on the original testset, and the lower transferrability the hybrid network has. Basically, the defense performance grows with the performance of the hybrid model. This turns out to be a very pleasant trend since some existing defense models increase the robustness of their systems but at the cost of the drop of the accuracy [14]. However, when the number of parameters grows, the system turns out to be slower and requires more computing resources. We utterly choose 300 trees and 10000 nodes to attain the proper performance and speed within our hardware capabilities.

5.3. From which layer to grow random forests

Theoretically speaking, we can substitute the neural network from any layer. But actually, the starting layer matters regarding the final performance of our model. Using the proposed criterion defined in Section 4.1, we find starting from the layer that has the largest gap of relative ℓ_2 distance from its prior layer gains a better transferability than the penultimate layer, which is shown in Table 8.

We also investigate the impact of max pooling and batch normalization layers. We conduct the experiment after the 6^{th} activation of AlexNet, where there are 4 layers including convolutional, activation, max pooling and batch normalization. Table 9 shows the best performance is achieved after the activation layer.

6. Conclusion

To the best of our knowledge, there is no previous work considering using models with no gradients to defend adversarial attacks, probably because of the poor performance of non deep learning methods on high-level tasks (*e.g.* image classification). While in our paper, we propose a simple hybrid model to overcome the drawbacks of individual DNNs and random forests. Through comprehensive experiments, we demonstrate that our proposed method is very effective for defending both white-box (including ultra white-box attack) and different kinds of black-box attacks. Simple but advantageous, our approach provides a preliminary evidence in support of the effectiveness of hybrid models, shedding lights on a promising direction to pursue.

References

[1] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, July 2018.

[2] S. Baluja and I. Fischer. Adversarial transformation networks: Learning to generate adversarial examples. *arXiv preprint arXiv:1703.09387*, 2017.

[3] A. N. Bhagoji, W. He, B. Li, and D. Song. Exploring the space of black-box attacks on deep neural networks. *arXiv preprint arXiv:1712.09491*, 2017.

[4] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

[5] W. Brendel, J. Rauber, and M. Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

[6] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. In *Submissions to International Conference on Learning Representations*, 2018.

[7] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.

[8] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26. ACM, 2017.

[9] T. Colthurst, D. Sculley, G. Hendry, and Z. Nado. Tensorforest: scalable random forests on tensorflow. In *Machine learning systems workshop at NIPS*, 2016.

[10] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and D. H. Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv preprint arXiv:1705.02900*, 2017.

[11] P. Domingos and G. Hulten. Mining high-speed data streams. In *Kdd*, volume 2, page 4, 2000.

[12] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[14] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[15] C. Guo, M. Rana, M. Cissé, and L. van der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[16] A. Kantchelian, J. Tygar, and A. Joseph. Evasion and hardening of tree ensemble classifiers. In *International Conference on Machine Learning*, pages 2387–2396, 2016.

[17] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[19] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[21] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

[22] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[23] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

[24] S. M. Moosavi Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, number EPFL-CONF-218057, 2016.

[25] N. Narodytska and S. Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1310–1318. IEEE, 2017.

[26] B. Nelson, B. I. Rubinstein, L. Huang, A. D. Joseph, S. J. Lee, S. Rao, and J. Tygar. Query strategies for evading convex-inducing classifiers. *Journal of Machine Learning Research*, 13(May):1293–1332, 2012.

[27] L. Nguyen and A. Sinha. A learning and masking approach to secure learning.

[28] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017.

[29] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 372–387. IEEE, 2016.

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[31] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

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

[33] F. Tramr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. Ensemble adversarial training: Attacks

and defenses. In *International Conference on Learning Representations*, 2018.

[34] W. Xu, D. Evans, and Y. Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.