Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

EnsembleFool: A method to generate adversarial examples based on model fusion strategy

Wenyu Peng[a,c], *Renyang Liu*[b,c], *Ruxin Wang*[a,c], *Taining Cheng*[a,c], *Zifeng Wu*[a,c], *Li Cai*[a,c], *Wei Zhou*[a,c,*]

[a] *Piolet School of Software, Yunnan University, Kunming 650500, China*
[b] *School of Information Science and Engineering, Yunnan University, Kunming 650500, China*
[c] *Engineering Research Center of Cyberspace, China*

ARTICLE INFO

Article history:
Received 13 December 2020
Revised 23 March 2021
Accepted 29 April 2021
Available online 7 May 2021

Keywords:
Deep learning
Adversarial examples
Ensemble models
Self-adaptive
White-box attack
Transferability

ABSTRACT

Deep neural networks have been shown vulnerable to adversarial attacks launched by adversarial examples. These examples' transferability makes an attack in the real-world feasible, which poses a security threat to deep learning. Considering the limited representation capacity of a single deep model, the transferability of an adversarial example generated by a single attack model would cause the failure of attacking other different models. In this paper, we propose a new adversarial attack method, named EnsembleFool, which flexibly integrates multiple models to enhance adversarial examples' transferability. Specifically, the model confidence concerning an input example reveals the risk of a successful attack. In an iterative attacking case, the result of a previous attack could guide us to enforce a new attack that possesses a higher probability of success. Regarding this, we design a series of integration strategies to improve the adversarial examples in each iteration. Extensive experiments on ImageNet indicate that the proposed method has superior attack performance and transferability than state-of-the-art methods.

1. Introduction

Deep neural networks enable intelligent systems to exhibit high fidelity on interested tasks such as computer vision, natural language processing, speech recognition, and recommendation, considerably lowering the human labours in real applications (Kurakin et al., 2017). However, the intention to defraud these models makes the systems to be possibly unsafe. Especially, the adversarial examples (Akhtar and Mian, 2018; Szegedy et al., 2014), generated by adding undetectable perturbations into the original input, have the ability of misleading the deep models and resulting in incorrect predictions(Déniz et al., 2019; Pedraza et al., 2020; Zhao et al., 2020). As Fig. 1 illustrates, a well-trained model predicts the unperturbed input as the true label with 95.62% confidence, whereas giving a wrong label with 87.56% confidence when the image is corrupted by well-designed noise. The deep models' vulnerability remains a critical safety issue and it is necessary to evaluate the model robustness with respect to adversarial attacks. Hence, advanced attack techniques are expected to cooperate

* Corresponding author.
E-mail address: zwei@ynu.edu.cn (W. Zhou).
https://doi.org/10.1016/j.cose.2021.102317

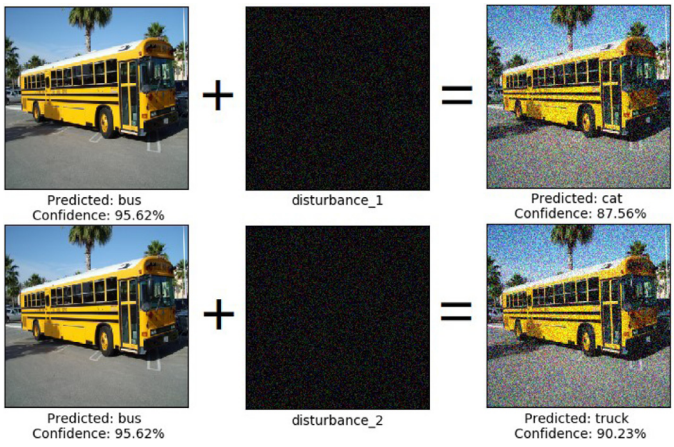


Fig. 1 – An example of adversarial perturbations. First row: the clean image *x* classified as "Bus" with 95.62% confidence by DNN models. Second row and Third row: the image *x* plus perturbations *r* classified as "cat" and "truck" with confidence 87.56% and 90.23%, respectively.

with the adversarial defense community to improve the safety of intelligent systems.

The existing adversarial attack methods can be roughly divided into two categories: **(1) the white-box attack** (Carlini and Wagner, 2017; Chen et al., 2020; Goodfellow et al., 2015; Kurakin et al., 2017) and **(2) the black-box attack** (Brendel et al., 2018; Chen et al., 2019; 2017; Ren et al., 2020; Zhao et al., 2018). The white-box attack depicts a problem that the attacker has the knowledge of the structure and parameters of the to-be-attacked model, which could generate extremely deceptive adversarial examples during attacking. For instance, Goodfellow et al. (2015) proposed the fast gradient sign method (FGSM) that can effectively calculate the perturbation by adjusting the model's gradients. Unlike the white-box attack, the black-box attack only allows the model outputs in terms of the adversarial inputs to be accessible, while the model details are absent. This poses a more difficult problem than the white-box attack because of the model's limited on-hand information. To handle this problem, recent researches indicate that adversarial examples have transferability (Liu et al., 2017), which is saying that the examples crafted for a given model can fool other unknown models. This property inspires several state-of-the-art black-box attacks (Brendel et al., 2018; Chen et al., 2017; Zhao et al., 2018). For example, MI-FGSM (Dong et al., 2018) applies an iterative and momentum technique into FGSM to generate improved adversarial examples, achieving enhanced attack ability in both white-box and black-box cases.

The transferability of adversarial examples endows the methods mentioned above with the ability to attack pure deep models. Unfortunately, when the models are equipped with certain defense mechanisms, the above methods exhibit low efficacy of fooling the black-box models. For example, adversarial training (Song et al., 2020; Tramèr et al., 2018) and input modification (Cohen et al., 2019; Liu et al., 2019) are two typical ways to enhance the robustness of deep models in the black-box case. This implies an imparity issue between different models, suggesting that it is not implementable to create a universal attack with a single model. Instead, the characteristics of multiple models could be simultaneously considered when synthesizing the input perturbation such that the generated example could be a threat to all the models.

To further investigate the model diversity, we visualize the attention maps of different deep models with varying architectures through CAM (Zhou et al., 2016). Fig. 2 shows that the main focuses of different models during prediction are located in different spatial regions, where we illustrate using a heatmap mask and the focus difference are marked using red boxes. This verifies that different models would have resistance to attacks in different regions of the input image and more importantly, the attack result of one model could be beneficial to attack another model. Hence, the input perturbation could be improved by involving the expressed robustness of multiple models.

Inspired by the above analyses, in this paper, we propose a novel attack method, named as EnsembleFool. Specifically, considering the attack map of a single model is limited and different models possess different maps, we follow the MI-FGSM framework, which integrates multiple models to enlarge the attack map such that the attacking capability is enhanced. To implement a flexible integration process, we develop a series of fusion strategies based on the attacking results in the previous iteration. This is motivated by the attack effect of an adversarial example that can be clearly expressed by the output of the model, and hence, the output could guide the attacking in the next iteration. In such a way, the adversarial examples crafted by the ensemble of multiple models exhibit enhanced attacking ability in both white-box and black-box cases. The main contributions of this paper are summarized as follows:

- We investigate the transferability of the adversarial examples crafted by a single model, showing that the primary focuses of different models are located in different receptive fields and the attack map could not be shared between different models. This validates the necessity of integrating multiple models in a flexible way.

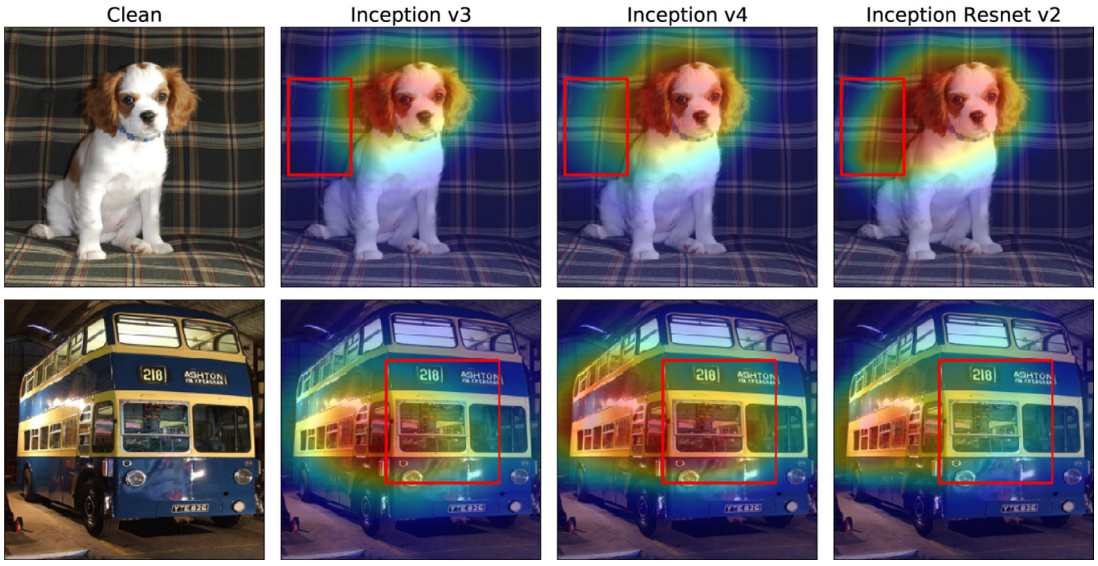


Fig. 2 – Demonstration of the diverse discriminative regions in different models. The first column is clean image and the rest is adopting class activation mapping Zhou et al. (2016) to visualize the attention maps of three trained models–Inception v3 Szegedy et al. (2016), Inception v4 Szegedy et al. (2017) and Inception Resnet v2 Szegedy et al. (2017). (Better viewed in color.).

- Based on MI-FGSM, we are well motivated to develop a novel ensemble attack method called EnsembleFool, which produces the adversarial examples by integrating the attack results of multiple models in an adaptive way such that the most informative attack could be dominant in each attack iteration.
- Extensive experiments validate the state-of-the-art performance of the proposed method even in the cases of attack models with defense mechanisms.

The remainder of this paper is organized as follows. The related works and the preliminary are briefly reviewed in Section 2 and Section 3, respectively. The proposed method is introduced in Section 4.1. The experiments are presented in Section 5, with the conclusion drawn in Section 6.

2. Related work

In this section, we briefly review the relevant methods to the current work, including adversarial attacking, attacking using ensemble of models, and adversarial defence. **Adversarial attacking.** Deep neural networks have been shown vulnerable to adversarial examples (Szegedy et al., 2014), posing the requirement of developing advanced attack methods to evaluate the robustness of the models. For example, Goodfellow et al. (2015) argued that the primary reason of the vulnerability was the linear nature of data in high-dimensional spaces and proposed the fast gradient sign method (FGSM) to generate adversarial examples by one single gradient step. Kurakin et al. (2017) extended FGSM with an iterative manner (I-FGSM) and demonstrated real-world adversarial examples. Dong et al. (2018) proposed a broad class of momentum-based iterative algorithms (MI-FGSM) to boost adversarial attacking. Xie et al. (2019) and Dong et al. (2019) used diverse inputs and translational invariance (TIM) to craft more transferable examples, respectively. While these methods have achieved pleasing performance, the transferability of adversarial examples is still a concerning factor in developing novel techniques.

Attacks using an ensemble of models. The representation capacity of a single deep model is limited, even in the case of very deep architectures. This brings an issue that this model's attack effect cannot be completely transferred to another model with a different architecture. If there is a significant difference between the model architectures, the transferability of the adversarial example could be considerably reduced. To maximally enhance the transferability of adversarial examples or boost the attack effect, multi-model fusion is a promising way to compensate for the shortage of a single model, which is seldomly considered. To the best of our knowledge, the method MI-FGSM (Dong et al., 2018) is an early trial on assembling multiple models, which achieves improved performance compared with single-model methods. In detail, MI-FGSM forces the weight of each model to be fixed during fusion. Differently, the current work advocates that different models have different focused regions in the input image, resulting in different attack maps. These maps could be evaluated by using the model outputs, which can further guide the attacking process in future iterations. Hence, the models to be assembled should be fused in a flexible way instead of a fixed manner.

Adversarial defense. Facing the threat of adversarial attacking, adversarial defense has recently been deeply investigated to alleviate the vulnerability issue of deep models. One typical way is to improve the prediction accuracy in the case of corrupted inputs (Déniz-Suárez et al., 2020; Liao et al., 2018; Xie et al., 2018). Alternatively, a trial to detect the adversarial examples before feeding them into, say classification models, is also researched (Metzen et al., 2017; Pang et al., 2018). Besides these methods, the fuzzy gradient technique has ob-

tained wide attention in existing literature (Athalye et al., 2018; Papernot et al., 2016), which yields pleasing performance in the black-box attack case whereas showing limited robustness in the white-box attack case (Guo et al., 2018; Liao et al., 2018; Xie et al., 2018; 2018). In this article, we focus on generating more transferable adversarial samples against both black-box and white-box defenses.

3. Preliminary

Before presenting the proposed method, we first introduce the notations and the preliminary knowledge about the current work. Let x denote an image, y^{true} denote the corresponding ground-truth label, θ denote the network parameters, and $L(x, y^{true}; \theta)$ denote the loss function. To generate the adversarial example, our goal is to maximize the loss function $L(x, y^{true}; \theta)$ given x under the constraint that the generated example $x^{adv} = x + r$ should look visually similar to the original image x and the corresponding predicted label $y^{adv} \neq y^{true}$. In this work, we use $l_\infty - norm$ to measure the perceptibility of adversarial perturbations, i.e., $||r||_\infty \leq \epsilon$, where ϵ is the upper bound of perturbations allowed. The loss function is defined as

$$L(x, y^{true}; \theta) = -1_{y^{true}} \cdot \log(softmax(l(x; \theta))), \tag{1}$$

where $1_{y^{true}}$ is the one-hot encoding of the ground-truth y^{true} and $(l(x; \theta))$ is the logits output.

Based on the above formulation, we introduce a series of attack methods that inspire the proposed method.

Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015): FGSM is a typical white-box attack algorithm that generates the adversarial perturbation by maximizing the loss function $L(x^{adv}, y^{true}; \theta)$ with a one-step update. The generated perturbation is added to the original input to produce the adversarial example. The advantage is low computational cost and fast generation speed, while the disadvantage is weak attack ability. The update can be expressed as:

$$x^{adv} = x + \epsilon \cdot sign(\nabla_x L(x, y^{true}; \theta)), \tag{2}$$

where $sign(\cdot)$ is a sign function to restrict the perturbation in the $l_\infty - norm$ bound, and $\nabla_x L$ is the gradient of the loss function with respect to x.

Iterative Fast Gradient Sign Method (I-FGSM) (Kurakin et al., 2017): I-FGSM extends FGSM to an iterative version by applying FGSM in iterations with a small step size α. This method alleviates the issue of FGSM which states that one gradient update could not sufficiently reveal the potential risk of the model, hence possibly failing to attack. The update schema of I-FGSM is:

$$x_{t+1}^{adv} = Clip_x^{\epsilon}\left\{x_t^{adv} + \alpha \cdot sign(\nabla_x L(x_t^{adv}, y^{true}; \theta))\right\}, \tag{3}$$

where $x_0^{adv} = x$ and $Clip_x^{\epsilon}$ denotes the adversarial example clipped by the ϵ-ball of the original image x.

Momentum Iterative Fast Gradient Sign Method (MI-FGSM) (Dong et al., 2018): MI-FGSM introduces a variety of momentum-based iterative algorithms to enhance the adversarial attack capability by incorporating momentum into the iterative attacking process. This method allows to compute more and more accurate perturbation in iterations, resulting in enhanced attacking performance. The update procedure is formalized as follows:

$$g_{t+1} = \mu \cdot g_t + \frac{\nabla_x L(x_t^{adv}, y^{true}; \theta)}{\left\|\nabla_x L(x_t^{adv}, y^{true}; \theta)\right\|_1}, \tag{4}$$

$$x_{t+1}^{adv} = Clip_x^{\epsilon}\left\{x_t^{adv} + \alpha \cdot sign(g_{t+1})\right\}, \tag{5}$$

where g_t is the accumulated gradient at iteration t and μ is the decay factor.

4. Methods

In this section, we introduce the proposed adaptive ensemble model in detail, with the variations of manual and dynamical adaptivity.

4.1. *Framework of ensembleFool*

The resistant ability to adversarial attacks varies across different deep models. In the case of fusing multiple models, different models should provide different contributions to the final adversarial perturbation, instead of the same contribution like MI-FGSM. This is inspired by the investigation of Fig. 2 which depicts the diversity of different models. Hence, we develop the EnsembleFool framework, as shown in Fig. 3. Different from MI-FGSM, we consider that each of the models should be adapted according to certain strategies such that the generated perturbation could maximize the vulnerability of all models. This is implemented via the weight $w_k \in [1, k]$ as illustrated in the step 2 of Fig. 3, where K is the number of the models involved. Then, the fusion process is conducted on the logit layers, yielding a final logit expression which is written as:

$$l(x_t^{adv}) = \frac{\sum_{k=1}^{K} w_k \cdot l_k(x_t^{adv})}{\sum_{k=1}^{K} w_k}, \tag{6}$$

where $l_k(x)$ are the logit of the k-th model, and w_k is the weight of the k-th model. The loss function $L(x, y)$ is defined as the softmax cross-entropy loss:

$$L(x_t^{adv}, y^{true}) = -1_{y^{true}} \cdot \log(softmax(l(x_t^{adv}))), \tag{7}$$

where 1_y is the one-hot encoding of label y. Based on this framework, the setting of the weights is the key to the success of the improvement, which is elaborated in the following sections.

4.2. *EnsembleFool with forced adaptivity*

In the iterative attacking scenario, the adversarial perturbation is obtained by cumulatively exploiting the properties of the attacked model. In this regard, the attacking result in each

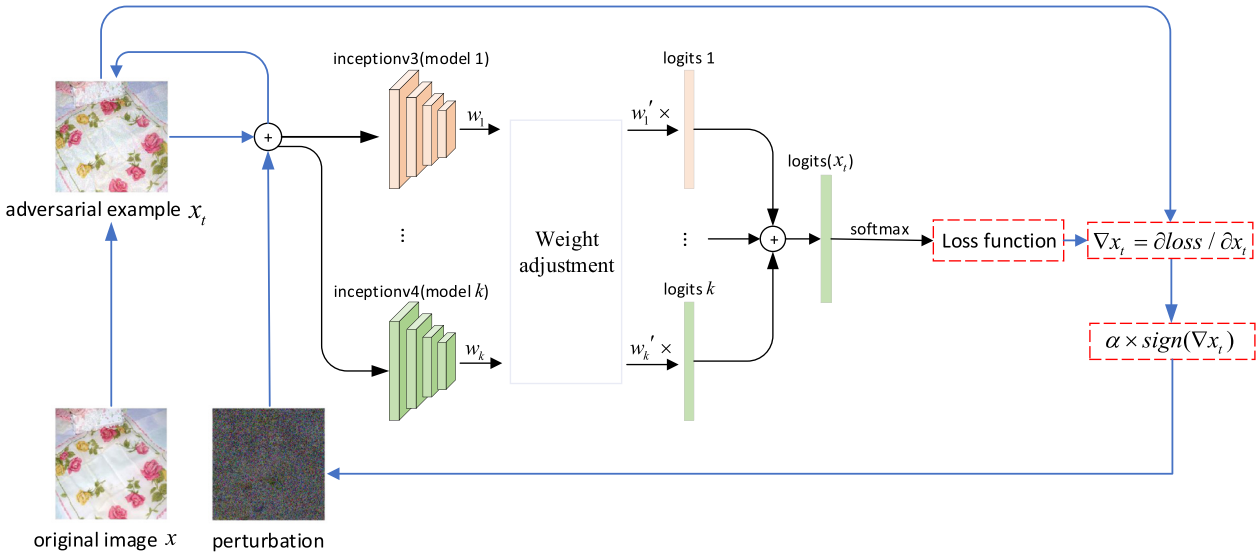


Fig. 3 – The framework of our proposed fusing strategy for adversarial attack.

iteration could provide valuable cues for updating the perturbation in the next iteration. Intuitively, in a certain iteration, if the perturbed input successfully attacks the model, the perturbation is viewed as a useful one, and otherwise, as a useless one. This inspires us to develop a forced adaptivity scheme by utilizing the attacking results in each iteration. Specifically, the results have two cases: successful and unsuccessful attacks. The successful attack informs us that the logit should have a lower contribution to the perturbation generation since this model has already been attacked. By contrast, the unsuccessful attack implies a higher contribution, conveying that this model should focus more on the next iteration. As such, we assign the model successfully attacked with a lower $w_k = \delta_l$ and the model unsuccessfully attacked with a higher $w_k = \delta_h$. In this way, the whole process would pay more and more attention to the models that have not been attacked successfully, thus maximizing all models' vulnerability. In detail, δ_l is fixed to be 1 and δ_h is fixed to be greater than 1, where the values are manually specified, hence implying the name *forced adaptivity*(EnsembleFool-C). In the initial iteration, all weights w_i are set to 1. Here, one may argue that when a certain model has already been attacked successfully, this model could provide nothing more information in the following iterations. In fact, we keep the minimal value of w_i greater than 0, such that the contribution of each model could be remembered consistently and the contributions of different models could have a stable balance, ensuring the improvement of performance.

4.3. *EnsembleFool with dynamical adaptivity*

The above-mentioned forced adaptivity imposes adaptive weights to different models, whereas the weights are predefined, which could be viewed as discretized adaptivity. Stepping from the discretised version to a continuous version, we obtain a more flexible fusion procedure, which is named as dynamical adaptivity. Consider that the output probability of a model tells how confident the model is about the input. Hence, the probability is naturally a choice of the weight setting. While the model has higher confidence on the correct label, the model is not yet attacked successfully, implying a higher weight value in next iteration; instead, while the model has lower confidence on the correct label, the model is probably attacked successfully, implying a lower weight value. This is consistent with the intuition mentioned in the forced adaptivity. As shown in Fig. 4, the dynamic adaptivity estimates the probabilities of all models with respect to the input, where the probabilities are multiplied with the corresponding logits followed by a summation to obtain a cumulative logit, which can be observed from the blue box. In this way, the weights of different models can be adjusted adaptively in each iteration of the attack.

Mathematically, the above fusion process can be expressed as:

$$l(x_t^{adv}) = \frac{\sum_{k=1}^{K} p_k(x_t^{adv}, y^{true}) \cdot l_k(x_t^{adv})}{\sum_{k=1}^{K} p_k(x_t^{adv}, y^{true})}, \tag{8}$$

where y^{true} means the ground-truth label of x, and $p_k(x, y)$ are the prediction of the label y for x by the k-th model.

We present the proposed EnsembleFool algorithm with dynamical adaptivity(EnsembleFool-A) in Algorithm 1. As in the forced adaptivity, We initialize the weights of all models to 1. In the following iterations, the weights are set according to the ground-truth label probabilities, predicted by the models.

5. Experiments

In this section, we conduct a series of experiments on public datasets to validate the effectiveness of the proposed method by comparing it with the state-of-the-arts.

5.1. *Experimental settings*

Dataset: To construct the attack dataset, we randomly choose 1000 images belonging to the 1000 categories from the ILSVRC

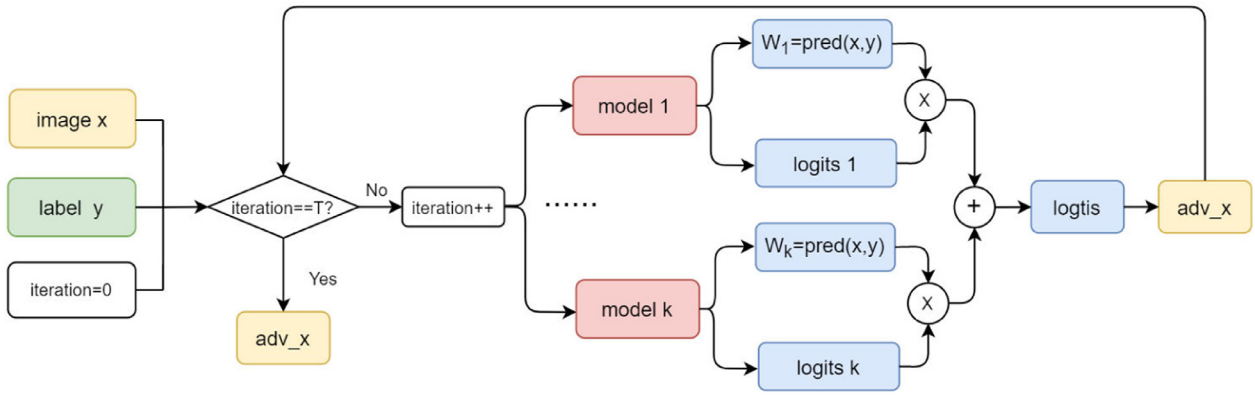


Fig. 4 – The process of adjusting the weight of adaptive selection method(EnsembleFool-A).

Algorithm 1 EnsembleFool-A.

Input: K classifiers $f_1, f_2, \ldots, f_k$, its logits $l_1, l_2, \ldots l_k$ and predictions $p_1, p_2, \ldots p_k$; a real example x with ground-truth label y^{true}.
Input: Perturbation size ϵ; iterations T and decay factor μ.
Output: An adversarial example x^{adv}.

$\alpha = \epsilon/T$
$g_0 = 0; x_0^{adv} = x$;
for all $t = 0 \rightarrow T-1$ **do**
  $w_i = 1, \forall_i \in [1, K]$
  for all $k = 1 \rightarrow K$ **do**
    Get the logit by $l_k(x_t^{adv})$;
    Get the prediction by $p_k(x_t^{adv}, y^{true})$;
  end for
  Fuse the logits as Eq. 7;
  Get softmax cross-entropy loss as Eq. 8;
  Get the gradient $\nabla_x J(x_t^{adv}, y^{true})$
  Update g_{t+1} by Eq. 4;
  Update x_{t+1}^{adv} by Eq. 5;
end for
return $x^{adv} = x_T^{adv}$

2012 validation set (Russakovsky et al., 2015), which are correctly classified by all the testing models as specified in Section 5.2. All the images are resized to 299x299x3 beforehand.

Competitors: The proposed method follows the framework of I-FGSM and MI-FGSM, which are selected as the baselines for fair comparison. Besides, to show the effectiveness, we employ three advanced defence models, including high-level representation guided denoiser (HGD) (Liao et al., 2018), random resizing and padding (R&P) (Xie et al., 2018), and NIPS-r3[1].

Platform: All the experiments are conducted on a GPU server with one Intel Xeon E5 2620 v4 CPU, 128GB RAM, and two NVIDIA RTX 2080 TI GPUs. The software environment is under CentOS 7, Python 3.6, and Tensorflow-GPU 1.13.1.

Metrics: We use the attack success rate and score to compare the performance of different methods. **Success rate** is an essential metric in the adversarial attack, which is calculated by dividing the number of misclassified adversarial examples by the total number of examples. **Attack Score** is an excellent way to evaluate the success rate of attack and the perturbation size. While there is a trade-off between the success rate of attack and the perturbation size, our method could yield a noticeable improvement on both metrics. For each generated adversarial example, multiple defense models are used to predict the example, and the corresponding perturbation amount settles according to the recognition result. The formulation is:

$$D(x, x^{adv}) = \begin{cases} up_{limit} & f(x^a) = y \\ mean(||x - x^{adv}||) & f(x^{adv}) \neq y, \end{cases} \tag{9}$$

where up_{limt} denotes the upper limit. If the defense model correctly identifies the example, the attack is unsuccessful, and the perturbation amount is set to the upper limit up_{limit} directly. If the attack is successful, we use the average l_2 distance to calculate the perturbation of the adversarial example relative to the original example. For n adversarial examples, we finally calculate the average of all perturbation amounts under the defense model, as the dist score, which is calculated as:

$$Dist\_score(A) = \frac{1}{n}\sum_{i=1}^{n} \cdot D(x_i, x_i^{adv}). \tag{10}$$

where A denotes the defense model A. To make a fair comparison, we convert the dist score to the attack score which is ranging from 0 to 100. The conversion is:

$$Attack\_score = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{up_{limt} - Dist\_score(i)}{up_{limit}} * 100\right). \tag{11}$$

where m denotes the number of the defense models.

5.2. *Model settings*

Models: To implement the mutil-model architecture, we consider six different models. Three of them are normally trained models: Inception-v3 (Inc-v3) (Szegedy et al., 2016), Inception-v4 (Inc-v4) (Szegedy et al., 2017), and Inception-resnet-v2 (IncRes-v2) (Szegedy et al., 2017). The other three

[1] https://github.com/anlthms/nips-2017/tree/master/mmd

Table 1 – Black-box attack success rate (%) of EnsembelFool-C using different weights .

δh	Inc-v3	Inc-v4	IncRes-v2	Inc-v3$_{ens3}$	Inc-v3$_{ens4}$	IncRes-v2$_{ens}$
1	77.5	72.9	69.3	25.2	32.3	20.7
2	78.1	72.9	69.3	25.9	33.3	**20.8**
3	78.5	73.4	**69.7**	25.6	**33.7**	**20.8**
4	**78.6**	**73.9**	69.4	**26.6**	33.3	**20.8**
5	78.3	73.5	69.4	26.1	33.5	20.4
6	78.3	73.4	69.4	25.8	33.5	20.1
7	77.8	73.3	69.3	25.7	32.0	20.1

Table 2 – White-box attack success rate (%) of EnsembleFool-C using different weights.

δh	Inc-v3	Inc-v4	IncRes-v2	Inc-v3$_{ens3}$	Inc-v3$_{ens4}$	IncRes-v2$_{ens}$
1	99.5	98.1	96.1	**100**	**100**	97.0
2	**99.7**	98.7	97.3	99.9	**100**	97.6
3	**99.7**	98.7	97.3	**100**	**100**	98.7
4	**99.7**	**98.8**	**97.5**	99.9	**100**	**98.0**
5	**99.7**	98.7	**97.5**	99.9	**100**	97.7
6	**97.7**	**99.8**	**97.5**	99.8	**100**	97.6
7	99.6	**98.8**	**97.5**	99.8	**100**	97.6

are adversarially trained models: Ens3-adv-inception-v3(Inc-v3$_{ens3}$) (Tramèr et al., 2018), Ens4-adv-inception-v3(Inc-v3$_{ens4}$) (Tramèr et al., 2018), and Ens-adv-inception-resnet-v2(IncRes-v2$_{ens}$) (Tramèr et al., 2018). All these models are publicly available from the Tensorflow's Github repository. Github[2].

Hyper-parameters Settings: Following the settings of MI-FGSM (Dong et al., 2018), we set the maximum perturbation ϵ to 16, where the pixel range of all images to [0,255], the step size α to 1.6, and the number of iterations T to 10. In EnsembleFool-C, and EnsembleFool-A, we adopt the decay factor μ as 1.0.

5.3. *Ablation study*

5.3.1. *Investigation on model weights*

The forced adaptivity reminds us the task of manually assigning the weights to successfully and unsuccessfully attacked models. As described in Section 4.3, the weights play an important role in improving the performance of the model fusion, which are investigated here. Specifically, the weight values for successfully attacked models are fixed to 1, as mentioned in previous sections. We change the weight values for unsuccessfully attacked models from 1 to 7, and examine the performance of each case. The experiments are conducted for both black-box and white-box attacks. In the black-box attack, the adversarial examples are generated by the ensemble five networks and tested on the hold-out model. In the white-box attack, the adversarial examples are generated by integrating six networks and testing on the model within the ensemble model.

The result for the black-box and white-box attacks are persented in Table 1 and Table 2, respectively. We can see that when $\delta h = 4$, the attack performance is the best in both cases.

5.4. *Comparison with the baseline methods:*

5.4.1. *Comparison of attack success rate*

In this section, we compare the attack success rate of the proposed EnsembleFool-C and EnsembleFool-A with the baseline attack methods in both white-box and black-box attacks. As shown in Table 3, we observe that in the white-box attack, the proposed methods perform better than the competitors. For example, in Incres-v2, the success rate of I-FGSM and MI-FGSM both reach 96.1%, while that of EnsembleFool-A achieves 99.5%. A noticeable improvement by the proposed methods can be observed from the black-box attack. The performance of EnsembleFool-C is pleasing, and more importantly, in almost all cases, the attack success rate of EnsembleFool-A is about 10% higher than that of I-FGSM, and is 2%-4% higher than that of MI-FGSM. This validates the effectiveness of the proposed adaptive schemes, including forced and dynamical adaptivity.

Table 3 – Ensemble attack success rate (%).The *ensemble* column represents the white-box attack case that adversarial examples are generated by six model ensemble attacks tested on the corresponding models. The *hold-out* column means the black-box attack case that adversarial examples are produced by five model ensemble attacks and tested on the hold-out model.

Model	Attack method	Ensemble	Hold-out
Inc-v3	I-FGSM	99.6	52.3
	MI-FGSM	99.5	77.5
	EnsembleFool-C(**Ours**)	**99.7**	78.6
	EnsembleFool-A(**Ours**)	**99.7**	**81.9**
Inc-v4	I-FGSM	98.5	40.7
	MI-FGSM	98.1	72.9
	EnsembleFool-C(**Ours**)	98.8	73.9
	EnsembleFool-A(**Ours**)	**99.3**	**76.8**
IncRes-v2	I-FGSM	96.1	37.1
	MI-FGSM	96.1	69.3
	EnsembleFool-C(**Ours**)	97.5	69.4
	EnsembleFool-A(**Ours**)	**99.5**	**73.4**
Inc-v3$_{ens3}$	I-FGSM	99.8	14.7
	MI-FGSM	**100**	25.2
	EnsembleFool-C(**Ours**)	99.9	26.6
	EnsembleFool-A(**Ours**)	99.9	**27.3**
Inc-v3$_{ens4}$	I-FGSM	99.9	19.1
	MI-FGSM	**100**	32.3
	EnsembleFool-C(**Ours**)	**100**	32.3
	EnsembleFool-A(**Ours**)	99.8	**34.3**
IncRes-v2$_{ens}$	I-FGSM	97.4	12.4
	MI-FGSM	97.0	20.7
	EnsembleFool-C(**Ours**)	98.0	20.8
	EnsembleFool-A(**Ours**)	**99.2**	**21.2**

5.4.2. *Comparison of attack score*

We explore the effect of the proposed attack strategies under different upper limits, where the results are shown in Table 4

[2] https://github.com/tensorflow/models/tree/master/research/slim/nets

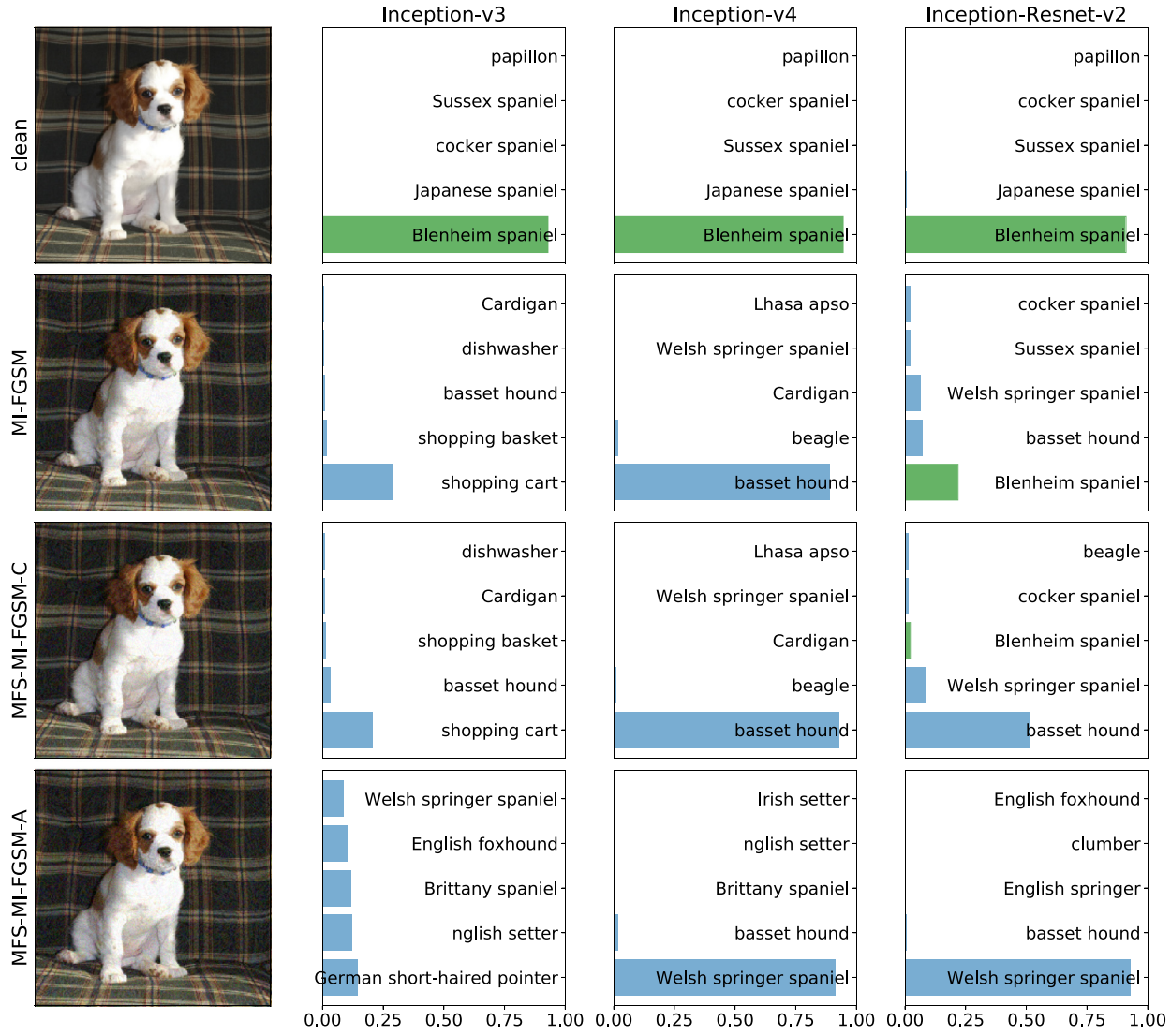


Fig. 5 – The top 5 confident predictions of the examples by different methods.

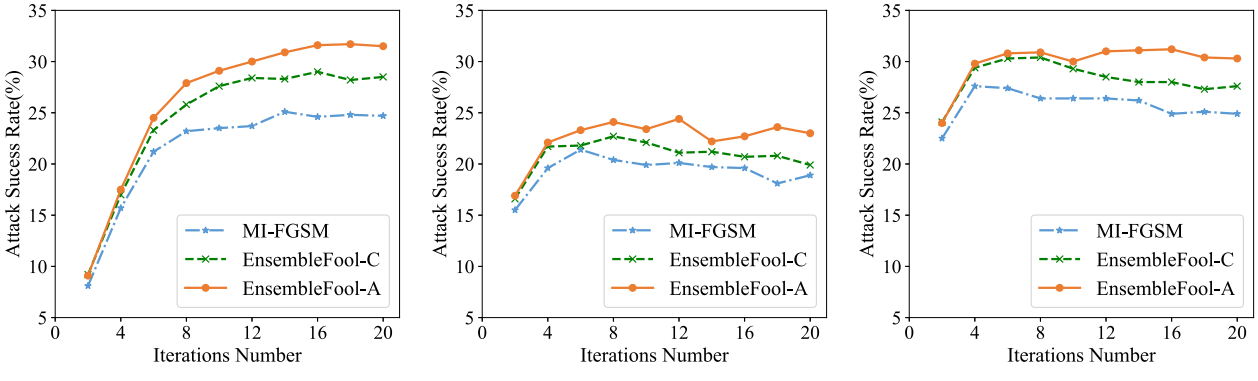


Fig. 6 – Attack success rates (%) on various numbers of iterations against advanced defence models.

Table 4 – Ensemble attack score when $up_{limit} = 32$. The *ensemble* column represents the white-box attack case and the *hold-out* column means the black-box attack case.

Model	Attack method	Ensemble	Hold-out
Inc-v3	MI-FGSM	40.05	31.09
	EnsembleFool-C(**Ours**)	**40.12**	31.51
	EnsembleFool-A(**Ours**)	40.06	**32.81**
Inc-v4	MI-FGSM	39.49	29.24
	EnsembleFool-C(**Ours**)	39.76	29.77
	EnsembleFool-A(**Ours**)	**39.90**	**30.77**
IncRes-v2	MI-FGSM	38.65	27.80
	EnsembleFool-C(**Ours**)	39.18	27.96
	EnsembleFool-A(**Ours**)	**39.97**	**29.23**
Inc-v3$_{ens3}$	MI-FGSM	**40.24**	10.12
	EnsembleFool-C(**Ours**)	40.20	10.68
	EnsembleFool-A(**Ours**)	40.15	**10.88**
Inc-v3$_{ens4}$	MI-FGSM	40.24	12.84
	EnsembleFool-C(**Ours**)	**40.24**	13.24
	EnsembleFool-A(**Ours**)	40.11	**13.67**
IncRes-v2$_{ens}$	MI-FGSM	39.02	8.27
	EnsembleFool-C(**Ours**)	39.41	8.33
	EnsembleFool-A(**Ours**)	**39.86**	**8.59**

Table 5 – Ensemble attack score when $up_{limit} = 64$. The *ensemble* column represents the white-box attack case and the *hold-out* column means the black-box attack case.

models	Attack method	Ensemble	Hold-out
Inc-v3	MI-FGSM	69.77	54.55
	EnsembleFool-C	**69.91**	55.05
	EnsembleFool-A	69.88	**57.36**
Inc-v4	MI-FGSM	68.80	51.07
	EnsembleFool-C	69.28	51.84
	EnsembleFool-A	**69.60**	**53.78**
IncRes-v2	MI-FGSM	67.38	48.55
	EnsembleFool-C	68.34	48.68
	EnsembleFool-A	**69.73**	**51.32**
Inc-v3$_{ens3}$	MI-FGSM	**70.12**	17.66
	EnsembleFool-C	70.05	18.64
	EnsembleFool-A	70.03	**19.09**
Inc-v3$_{ens4}$	MI-FGSM	70.12	22.57
	EnsembleFool-C	**70.12**	23.27
	EnsembleFool-A	69.95	**23.99**
IncRes-v2$_{ens}$	MI-FGSM	68.01	14.49
	EnsembleFool-C	**68.70**	14.57
	EnsembleFool-A	69.53	**14.89**

and Table 5. I-FGSM is ignored in this experiment because of its limited performance compared with MI-FGSM. The results demonstrate the superiority of the EnsembleFool methods compared with MI-FGSM in either cases.

5.4.3. *Exemplar top-5 predictions*

We employ an integrated model to predict the top-5 classes for a clean image and the images generated by MI-FGSM, EnsembleFool-C and EnsembleFool-A. Fig. 5 visualizes the predictions which show the difference of preference between the models. This validates the necessity of using flexible adjustment of model weights. Notably, the image corrupted by EnsembleFool has a low probability of the ground-truth label (Blenheim spaniel), and especially, we cannot find the ground-truth label in the top-5 possibilities by EnsembleFool-A.

Table 6 – Ensemble attack success rate(%) on advanced defence models.

Attack method	HGD	R&P	NIPS-r3
MI-FGSM	23.5	19.9	26.4
EnsembleFool-C(**Ours**)	27.6	22.1	29.3
EnsembleFool-A(**Ours**)	**29.1**	**23.4**	**30.0**

5.5. *Performance on attack defence model*

To further evaluate the effectiveness of our attack method, we conduct experiments on attacking the advanced defence models. For fair comparison, we employ three adversarially training models including HGD, R&P, and NIPS-r3. As shown in Table 6, it is seen that the attack success rates of EnsembleFool-A and EnsembleFool-C are superior to that of MI-FGSM in each of the three advanced defence models. To be precise, EnsembleFool-A outperforms the baseline attacks by 13%-23%.

We further compare EnsembleFool and MI-FGSM by using different numbers of attacking iterations ranging from 2 to 20. As shown in Fig. 6, it can be seen that the attack success rates of EnsembleFools (EnsembleFool-A and EnsembleFool-C) are higher than that of MI-FGSM with the same iteration number. In other words, EnsembleFool can achieve the same attack success rate as MI-FGSM with a fewer number of iterations.

6. Conclusions

Different deep models exhibit different characteristics of vulnerability when being attacked. Exploring this property would help to generate more powerful adversarial examples. Starting from this, in this paper, we propose a novel adaptive ensemble-based adversarial attack method that employs an adaptive strategy to fuse the information of multiple models. Specifically, the model prediction reveals whether correct or confident the model is with respect to the input. This fact informs us to use the model output as a guideline to adjust the weights of different models in fusion, resulting in two strategies: forced adaptivity and dynamical adaptivity. Compared with the existing gradient-based attack methods, the proposed method obtains noticeably improved attack success rates in black-box and white-box attacks. Moreover, the experiments of attacking defense methods validate the effectiveness of the proposed strategies.

Declaration of Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Declaration of Competing Interest

No.

CRediT authorship contribution statement

Wenyu Peng: Conceptualization, Methodology. **Renyang Liu:** Writing - original draft. **Ruxin Wang:** Writing - review & editing. **Taining Cheng:** Data curation. **Zifeng Wu:** Validation. **Li Cai:** Visualization, Investigation. **Wei Zhou:** Supervision.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 61762089, Grant 61663047, Grant 61863036, Grant 61762092, Grant 71972165, and Grant 61763048, the Yunnan Applied Basic Research Projects under Grant No. 202001BB050034, and Yunnan Province Science Foundation for Youths under Grant No.202005AC160007.

REFERENCES

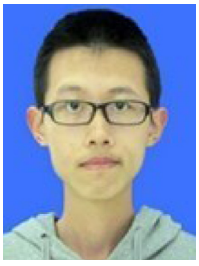
Akhtar N, Mian AS. Threat of adversarial attacks on deep learning in computer vision: a survey. IEEE Access 2018;6.
Athalye A, Carlini N, Wagner DA. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, 80; 2018.
Brendel W, Rauber J, Bethge M. In: ICLR. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models; 2018.
Carlini N, Wagner DA. In: SP. Towards evaluating the robustness of neural networks. IEEE Computer Society; 2017.
Chen J, Su M, Shen S, Xiong H, Zheng H. POBA-GA: perturbation optimized black-box adversarial attacks via genetic algorithm. Comput. Secur. 2019;85:89–106.
Chen J, Zheng H, Chen R, Xiong H. RCA-SOC: a novel adversarial defense by refocusing on critical areas and strengthening object contours. Comput. Secur. 2020;96:101916.
Chen P, Zhang H, Sharma Y, Yi J, Hsieh C. ZOO: zeroth order optimization based black-box attacks to deep neural networks without training substitute models. Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, 2017.
Cohen JM, Rosenfeld E, Kolter JZ. Certified adversarial robustness via randomized smoothing, 97; 2019.
Déniz O, Vállez N, García GB. In: IWANN. Adversarial examples are a manifestation of the fitting-generalization trade-off; 2019.
Déniz-Suárez O, Pedraza A, Vállez N, Salido J, Bueno G. Robustness to adversarial examples can be improved with overfitting. Int. J. Mach. Learn. Cybern. 2020;11(4).
Dong Y, Liao F, Pang T, Su H, Zhu J, Hu X, Li J. In: CVPR. Boosting adversarial attacks with momentum; 2018.
Dong Y, Pang T, Su H, Zhu J. In: CVPR. Evading defenses to transferable adversarial examples by translation-invariant attacks; 2019.
Goodfellow IJ, Shlens J, Szegedy C. In: ICRL. Explaining and harnessing adversarial examples; 2015.
Guo C, Rana M, Cissé M, van der Maaten L. In: ICLR. Countering adversarial images using input transformations; 2018.
Kurakin A, Goodfellow IJ, Bengio S. In: ICLR. Adversarial examples in the physical world; 2017.
Liao F, Liang M, Dong Y, Pang T, Hu X, Zhu J. In: CVPR. Defense against adversarial attacks using high-level representation guided denoiser; 2018.
Liu Y, Chen X, Liu C, Song D. In: ICLR. Delving into transferable adversarial examples and black-box attacks; 2017.
Liu Z, Liu Q, Liu T, Xu N, Lin X, Wang Y, Wen W. In: CVPR. Feature distillation: Dnn-oriented JPEG compression against adversarial examples; 2019.
Metzen JH, Genewein T, Fischer V, Bischoff B. In: ICLR. On detecting adversarial perturbations; 2017.
Pang T, Du C, Dong Y, Zhu J. In: NeurIPS. Towards robust detection of adversarial examples; 2018.
Papernot N, McDaniel PD, Wu X, Jha S, Swami A. In: IEEE Symposium on Security and Privacy. Distillation as a defense to adversarial perturbations against deep neural networks; 2016.
Pedraza A, Déniz O, Bueno G. Approaching adversarial example classification with chaos theory. Entropy 2020;22(11):1201.
Ren Y, Zhou Q, Wang Z, Wu T, Wu G, Choo KR. Query-efficient label-only attacks against black-box machine learning models. Comput. Secur. 2020;90:101698.
Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein MS, Berg AC, Li F. Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. 2015;115(3).
Song C, He K, Lin J, Wang L, Hopcroft JE. In: ICLR. Robust local features for improving the generalization of adversarial training; 2020.
Szegedy C, Ioffe S, Vanhoucke V, Alemi AA. In: AAAI. Inception-v4, inception-resnet and the impact of residual connections on learning. AAAI Press; 2017.
Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. In: CVPR. Rethinking the inception architecture for computer vision; 2016.
Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow IJ, Fergus R. In: ICLR. Intriguing properties of neural networks; 2014.
Tramèr F, Kurakin A, Papernot N, Goodfellow IJ, Boneh D, McDaniel PD. In: ICLR. Ensemble adversarial training: Attacks and defenses; 2018.
Xie C, Wang J, Zhang Z, Ren Z, Yuille AL. In: ICLR. Mitigating adversarial effects through randomization; 2018.
Xie C, Zhang Z, Zhou Y, Bai S, Wang J, Ren Z, Yuille AL. In: CVPR. Improving transferability of adversarial examples with input diversity; 2019.
Zhao Z, Dua D, Singh S. In: ICLR. Generating natural adversarial examples; 2018.
Zhao Z, Liu Z, Larson MA. In: CVPR. Towards large yet imperceptible adversarial image perturbations with perceptual color distance; 2020.
Zhou B, Khosla A, Lapedriza À, Oliva A, Torralba A. In: CVPR. Learning deep features for discriminative localization; 2016.

Wenyu Peng received the BE degree in network engineering from Yunnan University in 2019. She is a master degree candidate at the School of Software, Yunnan University. Her research interests include deep learning, adversarial attack and bio-informatics.

Renyang Liu received the BE degree in Computer Science from the Northwest normal University in 2017, He is a current Ph.D. candidate in School of Information Science and Engineering at Yunnan University, Kunming, China. His current research interest includes deep learning, adversarial attack and graph learning.

Ruxin Wang is currently an associate professor with the National Pilot School of Software, Yunnan University, Kunming, China. He received his BEng from Xidian University, his MSc from Huazhong University of Science and Technology, and his PhD degree from the University of Technology Sydney. His research interests include image restoration, deep learning, and computer vision. He focuses on the topic of image synthesis by using both discriminative models and generative models. He has authored and co-authored 20+ research papers including IEEE T-NNLS, T-IP, T-Cyb, ICCV, and AAAI. He has received "the 1000 Talents Plan for Young Talents of Yunnan Province" award.

Taining Cheng received the bachelor's degree from Yunnan University, Yunnan, China, in 2019. He is currently working toward the Masters degree in School of Software at Yunnan University. His current research interest includes learned index and distributed computing



Zifeng Wu received the BE degree in network engineering from Yunnan University in 2019. He is a master degree candidate at the School of Software, Yunnan University. His current research interests include machine learning, model compression and acceleration.

Li Cai was born in Kunming, China, in 1975. She received the M.S. degree in computer application from Yunnan University, China, in 2007. Then, she received the Ph.D. degree with the School of Computer Science, Fudan University, China, in 2020. From 1997 to 2002, she was a Research Assistant with Network Center. Since 2010, she has been an Associate Professor with the School of Software, Yunnan University. Her research interests include intelligent transportation, machine learning, visualization, and data quality.

Wei Zhou received the Ph.D. degree from the University of Chinese Academy of Sciences. He is currently a Full Professor with the Software School, Yunnan University. His current research interests include the distributed data intensive computing and bioinformatics. He is currently a Fellow of the China Communications Society, a member of the Yunnan Communications Institute, and a member of the Bioinformatics Group of the Chinese Computer Society. He won the Wu Daguan Outstanding Teacher Award of Yunnan University in 2016, and was selected into the Youth Talent Program of Yunnan University in 2017. Hosted a number of National Natural Science Foundation projects.