



Research Article

Boosting the Transferability of Ensemble Adversarial Attack via Stochastic Average Variance Descent

Lei Zhao,¹ Zhizhi Liu,¹ Sixing Wu,¹ Wei Chen,¹ Liwen Wu ,¹ Bin Pu ,² and Shaowen Yao¹

¹School of Software, Yunnan University, Kunming, Yunnan, China

²College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan, China

Correspondence should be addressed to Liwen Wu; lwwu@ynu.edu.cn and Bin Pu; pubin@hnu.edu.cn

Received 27 November 2023; Revised 25 April 2024; Accepted 30 April 2024; Published 11 May 2024

Academic Editor: Guowen Xu

Copyright © 2024 Lei Zhao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Adversarial examples have the property of transferring across models, which has created a great threat for deep learning models. To reveal the shortcomings in the existing deep learning models, the method of the ensemble has been introduced to the generating of transferable adversarial examples. However, most of the model ensemble attacks directly combine the different models' output but ignore the large differences in optimization direction of them, which severely limits the transfer attack ability. In this work, we propose a new kind of ensemble attack method called stochastic average ensemble attack. Unlike the existing approach of averaging the outputs of each model as an integrated output, we continuously optimize the ensemble gradient in an internal loop using the model history gradient and the average gradient of different models. In this way, the adversarial examples can be updated in a more appropriate direction and make the crafted adversarial examples more transferable. Experimental results on ImageNet show that our method generates highly transferable adversarial examples and outperforms existing methods.

1. Introduction

Deep neural networks (DNNs) have made promising breakthroughs in the field of computer vision (CV), such as automatic driving [1], face recognition [2], image classification [3], and many others [4, 5]. However, DNNs are proven to be vulnerable to adversarial attacks; the well-designed perturbed examples (adversarial examples) can often mislead CV models while keeping the imperception at the same time [6, 7]. This has drawn much attention to the research on adversarial attacks, because it can help us to identify model flaws [8] and improve the robustness of them [7]. In practice, it is hard to get specific information about the victim's model. Therefore, more practical black-box attacks are beginning to be studied extensively.

In general, there are two paradigms for Black-box attacks: query-based and transfer-based. Among them, query-based attacks have poor practical usability, because they typically require a lot of querying of the victim's model outputs, which tends to attract the victim's attention. Therefore, we concentrate on the transfer-based attacks. The principle behind transfer attacks is that adversarial examples have cross-model

transferability [9]; the adversarial examples crafted on surrogate models often can mislead other models, even these models use different structures [9]. This kind of transferability provides feasibility for strictly black box attacks. In recent years, the idea of an ensemble started to be used to improve the transferability of the adversarial examples [10, 11]. The model ensemble uses the outputs of multiple models instead of a single model to minimize the bias of a single model. When combined with adversarial attacks, it can help adversarial examples find an ensemble optimization direction, reducing overfitting to individual models and enhancing their transferability [10].

However, the optimization directions among different models may have significant variance due to the different model architectures [12]. Existing approaches ignore this kind of variance. They simply combine the outputs of different models and use it as the ensemble optimization direction, which leads to a variance between the average ensemble output and the individual model outputs and limits the transferability of the adversarial examples. In this work, we notice the optimization algorithm stochastic average gradient (SAG) [13] for stochastic gradient descent (SGD), which

reduces the stochastic gradient variance caused by randomly selecting samples in SGD. In the SGD algorithm, the variance between the gradient of randomly selected samples and the average gradient of all samples is similar with the variance encountered in model ensemble attacks. Both of these variations can be categorized as variances between the mean value and the individual values. Therefore, we plan to address it using the principles of SAG.

Based on the analyses above, we propose a novel method called stochastic average ensemble attack (SAEA). In our method, we use an internal loop to reduce the variance between the ensemble output and the multiple models' output. Then, we use this ensemble output in the external loop to generate adversarial examples. Specifically, at the external loop, we compute the outputs of the input image across multiple models and maintain them in memory. In the internal loop, similar with SAG, we randomly select one model and update the corresponding model output through the input images. Then, we fuse this updated output with the maintained output to obtain an ensemble output. Alternatively, we will update the internal loop image at the end of the internal loop using the ensemble output and feed it into the next internal loop. After multiple rounds of internal loop updates, the differences between the ensemble output and the multiple model outputs will be reduced. Finally, we use this ensemble output to perform iterative model ensemble attacks. In this way, SAEA can yield a more accurate update direction of the adversarial examples across multiple models to generate adversarial with higher transferability.

We have conducted extensive experiments to evaluate our method on the ImageNet dataset [14], and the results have indicated our SAEA can achieve better results in transfer attack scenarios compared to existing model ensemble attack methods.

The remainder of this paper is structured as follows: in Section 2, we summarize the related work on adversarial attacks and defenses. In Section 3, we first introduce our motivation and then introduce our attack algorithm. In Section 4, we demonstrate the effectiveness of our attack through extensive experiments and highlight its superiority over two other model ensemble attacks. Finally, Section 5 concludes this work.

2. Related Works

Since the concept of adversarial examples was proposed, a lot of attack algorithms have been subsequently designed, such as gradient-based attacks, input transformation attacks, and model ensemble attacks.

2.1. Gradient-Based Attacks. The fast gradient sign method (FGSM) [7] is the most representative attack method, which adding perturbations in the direction of the gradient to benign samples as follows:

$$x^{\text{adv}} = x^{\text{clean}} + \varepsilon \cdot \text{sign}(\nabla_x J(x, y^{\text{true}})), \quad (1)$$

where ε denote the magnitude of adversarial perturbations, and $J(\cdot)$ denote the loss function.

The iterative fast gradient sign method (I-FGSM) [15] proposed an iterative version of FGSM. It increases the transferability of adversarial samples by repeatedly adding small perturbations to the images as follows:

$$x_{t+1}^{\text{adv}} = \text{Clip}_x^\varepsilon \left\{ x_t^{\text{adv}} + \alpha \cdot \text{sign} \left(\nabla_{x_t^{\text{adv}}} J(x_t^{\text{adv}}, y^{\text{true}}) \right) \right\}, \quad (2)$$

where $\text{Clip}_x^\varepsilon(\cdot)$ limits the perturbation within the ε -ball of the benign input x , t denote the iteration number and α denote the step size.

The momentum iterative method (MIM) [11] introduced momentum into the I-FGSM to make the updating direction of the adversarial examples more stable. Nesterov iterative method [16] accelerated the craft speed of adversarial examples by applying accelerated gradients into the attack algorithm. The lookahead iterative method [17] tuned the update direction by recording the gradient in multiple previous steps to get rid of suboptimal regions during the update of the adversarial examples. These gradient-based attacks make the generated adversarial perturbations more accurate by optimizing the gradient, which is highly effective in both white-box and black-box attacks.

2.2. Input Transformation Attacks. Iterative gradient-based attacks require to update the adversarial examples multiple times on the local surrogate model, which can lead crafted examples overfit to the surrogate model and affect the ability to transfer to other models. To address this issue, input transformation attacks use the idea of data augmentation to reduce the risk of overfitting. Diverse input method (DIM) [18] proposed using the ideas of data augmentation, which adding random transformations to benign inputs to reduce the effect of overfitting. The translation invariant method (TIM) [19] proposed using translation invariant to generate a series of transformed copies to make the perturbations more accurate and used an optimized method to reduce computation. The scale invariant method (SIM) [16] proposed a method that performs a gradient attack by scaling the input image and averaging the gradients computed on the resulting copies. Admix [20] implements data augmentation by combining images from different categories in a small ratio. SSA [21] transforms the input samples in the spatial frequency domain for input augmentation to reduce the overfitting of the adversarial examples. PAM [22] introduces a semantic discriminator to prevent the difference between the semantics of the augmented samples and the original samples from being too large and generating inaccurate adversarial perturbations.

2.3. Model Ensemble Attacks. To further improve the robustness and eliminating bias in a single model, model ensemble methods have been widely studied and applied in the model training process. Such methods of improving the accuracy of model outputs can also be used to adversarial attacks. Liu et al. [10] first proposed the model ensemble attack, which increases the transferability by combing the prediction of different models. Dong et al. [11] proposed different ensemble methods to implement ensemble attacks by combing

logits and losses of different models. To further improve the transferability of the adversarial examples, Xiong et al. [12] proposed a stochastic variance-reduced method to improve the accuracy of ensemble output.

2.4. Adversarial Defenses. If it is possible to successfully attack a model with added defense mechanisms, it can significantly demonstrate the effectiveness of the attack method. In recent years, many methods have been proposed to improve the robustness of models. In general, there are three kinds of defense methods, including adversarial training [15, 23–25], adversarial detection [26–28], and input transformation defenses [29–34].

Adversarial training improving robustness by retraining on adversarial examples. Adversarial detection and input transformation defense detects and cleans samples before they are input into the model to reduce the threat of potential adversarial examples. In this paper, we will verify our method using some advanced defense models, including reducing the resolution of the adversarial examples (JPEG) [29], randomly resizing and padding the images (randomly resize and pad (R&P)) [30], using denoising to eliminate the perturbation (high-level representation-guided denoiser (HGD)) [31], using feature distillation (FD) [32] to purify the perturbation by redesigning the image compression framework JPEG, end-to-end image compression model to defend against adversarial examples (ComDefend) [33], randomized smoothing (RS) [34] technique to make the target model more robust.

3. Materials and Methods

3.1. Preliminary. Let x denote the benign input and y denote the corresponding ground-truth label of x . Given a classifier f with parameters θ that outputs a label as the prediction of the input image. The task of the adversarial attack is to craft an example x^{adv} , which is indistinguishable to human eyes with benign input x but can mislead the classifier f . Formally, the optimization function of this task can be defined as follows:

$$f(x^{\text{adv}}; \theta) \neq y^{\text{true}} \& d(x^{\text{adv}}, x) \leq \epsilon, \quad (3)$$

where $d(x^{\text{adv}}, x)$ represents the discrepancy between the perturbed images and the benign images, and we consider it as the l_∞ constraint in this paper.

The idea of ensembling has been widely used to improve model robustness [25]. It can also be applied to adversarial attacks because ensemble methods can diminish biases of individual models; they can yield an adversarial example update direction that is suitable for the majority of models. The existing model ensemble methods can be classified into three categories: (1) ensemble on predictions, (2) ensemble on logits, and (3) ensemble on losses.

Liu et al. [10] proposed a model ensemble attack through combining predictions of different models. The loss function for K models can be ensembled as follows:

$$J(x, y) = -1_y \cdot \log \left(\sum_{k=1}^K \omega_k p_k(x) \right), \quad (4)$$

where $p(\cdot)$ denote prediction output, ω_k denote the combine weight with $\omega_k \geq 0$ and $\sum_{k=1}^K \omega_k = 1$. Dong et al. [11] proposed using logits and losses to implement model ensemble attacks. The logits of K models can be ensembled as follows:

$$l(x, y) = \sum_{k=1}^K \omega_k l_k(x, y^{\text{true}}), \quad (5)$$

where l_k denote the logits output of the k th model. The losses of K models can be ensembled as follows:

$$J(x, y) = \sum_{k=1}^K \omega_k J_k(x, y^{\text{true}}), \quad (6)$$

where J_k denote the loss of the k th model.

In this work, we select the I-FGSM to craft adversarial examples, which craft adversarial examples by adding small perturbations multiple times. Adding perturbations multiple times can help prevent adversarial examples from getting stuck in the local optimum, thus increasing the transferability. In addition, we use the ensemble gradient to implement the model ensemble attack as follows:

$$G^{\text{ens}} = \sum_{k=1}^K \omega_k g_k(x, y^{\text{true}}), \quad (7)$$

where G^{ens} denote the ensemble gradient.

3.2. Motivation. Lin et al. [16] link the training of the models to the generation of adversarial examples. When generating adversarial examples, the parameters of the model are fixed, and the adversarial examples are updated by adjusting the added perturbations. This seems similar with the training process of the model, which fixes the training samples and seeks appropriate parameters to improve model performance. As a result, many methods for optimizing model training are beginning to be used to optimize adversarial examples [16].

To further improve the quality of the adversarial example, the idea of model ensemble optimization algorithms has been widely applied in the generation of adversarial examples [10, 11]. The principle of model ensemble attacks is that if an adversarial example can mislead multiple models simultaneously, it may have the ability to mislead more black-box models [11]. Additionally, by combining the outputs of multiple models, it is possible to reduce individual model biases and improve the accuracy of attacks. However, most ensemble attack methods directly add the outputs (predictions, logits, and losses) of different models with equal weights, and there is no additional processing applied to the obtained average result. Since adversarial examples have different optimization directions on different models, directly averaging the outputs of multiple models may limit the quality of

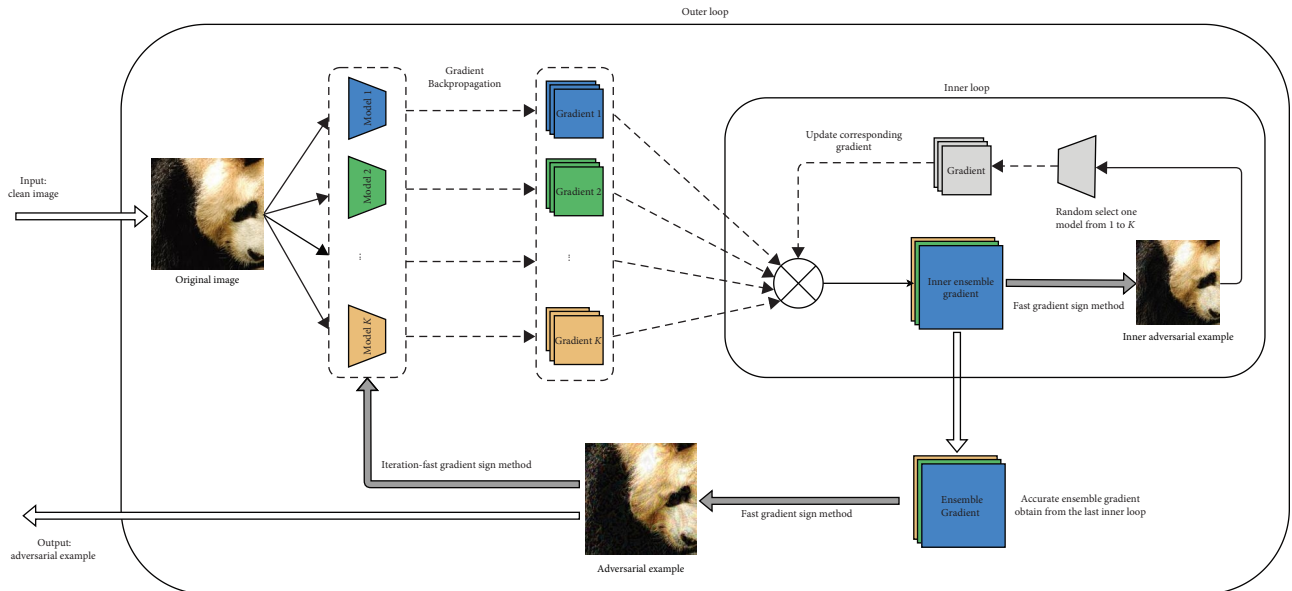


FIGURE 1: Overview of the stochastic average ensemble attack (SAEA). We use an internal loop to obtain an ensemble gradient with lower gradient variance, then use this ensemble gradient in an external loop to implement the iteration model ensemble attacks.

adversarial examples. This is similar to the issue of the large variance between the stochastic gradient and the ground-truth gradient in SGD, which can lead to getting trapped in local optima [35]. To solve the problem of gradient variance in SGD, algorithms such as SAG [13] and stochastic variance-reduced gradient [36] have been proposed. These kinds of algorithms aim to prevent getting stuck in local optima by reducing the stochastic variance introduced by randomly selecting samples. Based on the analysis above, we propose to combine model ensemble attacks with the optimization algorithms of SGD. This combination aims to reduce the stochastic variance in model ensemble attacks to improve the transferability of crafted adversarial examples; the overall structure of the SAEA can be seen in Figure 1.

3.3. SAEA Method. In this work, we view the process of model ensemble attack as the model training process, like Liu et al. [10]. We attempt to optimize the update direction of adversarial examples on ensembled models by reducing the difference between the ensemble gradient and individual gradients in the model ensemble attacks. This issue is similar to reducing the variance between the randomly selected sample gradients and actual gradients in the SGD algorithm. Therefore, we notice the SAG [13]. SAG retains the average of historical gradients and uses these averages to estimate the variance of the gradient, which can reduce the stochastic variance in the SGD algorithm. Additionally, by using the average gradient with lower random variance, SAG exhibits much faster convergence compared to the traditional SGD. If combined with adversarial attacks, it can also enhance the speed and quality of generating adversarial examples.

Inspired by SAG, we propose SAEA. Specifically, we used two loops, internal and external, to implement the attack algorithm. The internal loop obtains an ensemble gradient G^{ens} with a smaller variance, and the external loop is used to

combine with the iterative gradient attack algorithm to craft higher transferability adversarial examples.

The integration of SAEA with I-FGSM [15] can be summarized as Algorithm 1. At the beginning of the algorithm, we treat clean image x as the initial adversarial examples x_0^{adv} . In each external loop, we input the adversarial examples crafted in the previous loop into K models to obtain their respective gradients and maintain them in memory, denoted as G_k . After that, we utilize the idea of SAG in the internal loop to obtain an ensemble gradient G^{ens} with the minimum variance among these k gradients. The internal loop that obtains the ensemble gradient is the crucial part of our algorithm. Specifically, at the beginning of each internal loop, we treat the image input from the external loop as the initial internal adversarial examples \tilde{x}_0^{adv} . We randomly select one model from k models and input the internal adversarial examples into this selected model to update $G_k(\tilde{x}_m)$. After that, we use this updated $G_k(\tilde{x}_m)$ and others model gradient to obtain the new G^{ens} by Equation (7). Then, we use the ensemble gradient from each internal loop to update the internal adversarial examples. In order to maintain the semantic information of the internal adversarial examples, we apply a Clip function to them to ensure the accuracy of the ensemble gradient obtained in the internal loop. The crafted internal adversarial examples will be used in the next internal loop to continue update the G_k and G^{ens} by Equation (7). After M rounds of internal loop updates, G^{ens} will have a smaller random variance compared to the ground-truth gradients of each model, and we treat it as the ensemble gradient. This ensemble gradient can better represent the direction of example updates on the model set and craft adversarial examples with higher transferability.

In short, SAEA uses an ensemble gradient with smaller gradient differences among models to craft adversarial examples. This enables adversarial examples to achieve better

Input: A clean image x and its ground-truth label y , K surrogate models and their gradients $\{G_1, G_2, \dots, G_k\}$,
Input: The size of perturbation ε , iterations T , number of internal loops M , external loop step size α , internal loop step size β .
Output: Adversarial example x^{adv} .

- 1: $\alpha = \varepsilon/T$; $G_0 = 0$
- 2: Initialize $x_0^{\text{adv}} = x$;
- 3: **for** $t = 0$ to $T - 1$ **do**
- 4: Input x_t^{adv} and output $G_k(x_t^{\text{adv}})$ for
- 5: Initialize $\tilde{x}_0 = x$;
- 6: **for** $m = 0$ to $M - 1$ **do**:
- 7: Random choose a model k from $\{1, 2, \dots, K\}$;
- 8: Get the gradient of the chosen model $G_k(\tilde{x}_m)$;
- 9: Update G_m^{ens} by Equation (7);
- 10: Update $\tilde{x}_{m+1}^{\text{adv}} = \text{Clip}_x^\varepsilon\{x_m^{\text{adv}} + \beta \cdot \text{sign}(G_m^{\text{ens}})\}$;
- 11: **end for**
- 12: Update $x_{t+1}^{\text{adv}} = \text{Clip}_x^\varepsilon\{x_t^{\text{adv}} + \alpha \cdot \text{sign}(G_{M-1}^{\text{ens}})\}$;
- 13: **end for**
- 14: **return** $x^{\text{adv}} = x_T^{\text{adv}}$.

ALGORITHM 1: The SAEA-I-FGSM Algorithm.

results on multiple surrogate models, and the capability to transfer to other models is also improved. Additionally, SAEA can combine with other advanced input transformation methods (e.g., SI [16], DI [18], TI [19], and Admix [20]) to show better results.

3.4. Relationships between Different Attacks. SAEA, SVRE, and ENS are all model ensemble attacks and belong to Iteration FGSM; their relationships are shown in Figure 2, where p is the probability of the random transformation, k is the size representing the Gaussian kernel, and m is the scale copy numbers. The differences and transformations between them can be summarized as follows:

- (i) If the internal iteration M is set to 0, the SAEA and SVRE will become Ens.
- (ii) SVRE and Ens use different methods to obtain ensemble outputs with SAEA.
- (iii) If the internal iteration M , the external iteration T and the surrogate model numbers K is set to 1, ENS, SVRE, and SAEA will degrade to the FGSM.

4. Experiments

In this section, we will demonstrate our method with extensive experiments. First, we introduce the parameter settings about our experiment. Then, we compare the attack success rate (ASR) of our method on single-representative models and ensemble-trained models with other SOTA ensemble methods. In addition, we also validate our methodology on some advanced defense methods. Finally, we will conduct ablation experiments to explain the parameters that affect the experimental results.

4.1. Experiments Setup

4.1.1. Models. We choose four representative models, including Inception-v3 (Inc-v3) [36], Inception-v4 (Inc-v4) [37], Inception-Resnet-v2 (IncRes-v2) [37], Resnet-v2-101 (Res-v2) [38] to craft adversarial examples. To better evaluate the transferability of the crafted adversarial examples, we add three adversarially trained models, including Inc-v3_{ens3}, Inc-v3_{ens4}, and IncRes-v2_{ens} [25].

In addition, we also choose seven input transformation defense methods to validate the adversarial examples, which are: JPEG compression [29], R&P [30], NIPS-r3, HGD [31], FD [32], ComDefend [33], and RS [34].

4.1.2. Dataset. The dataset we used is the ImageNet-compatible dataset [14] that is commonly used in adversarial attack algorithms [12, 19], which contains 100 images selected from the ImageNet dataset.

4.1.3. Baseline. In the experiment, we compare our method with SVRE and ENS algorithms on multiple baseline attack methods, which are I-FGSM [15], MIM [11], SI [16], DI [18], and TI [19]. The attack type is a nontargeted attack. For all ensemble attack methods, we set the weight ω of multiple outputs of the ensemble model to $1/K$, where K denotes the total number of ensembled models.

For better comparison, the hyperparameters in the experiment remain the same as in previous works [11, 12], with a maximum perturbation ε value of 16 and pixel values in the range of 0–255. The attack iterations T is set to 10, the decay factor μ for all baselines to craft adversarial example are set to 1. For MIM, the step size α is set to 1.6. For TIM, we set the kernel size to 7. For DIM, the probability of random transformation p is set to 0.5. For SIM, we set the scale copy numbers m to 5. The parameters in the SAEA are the same as

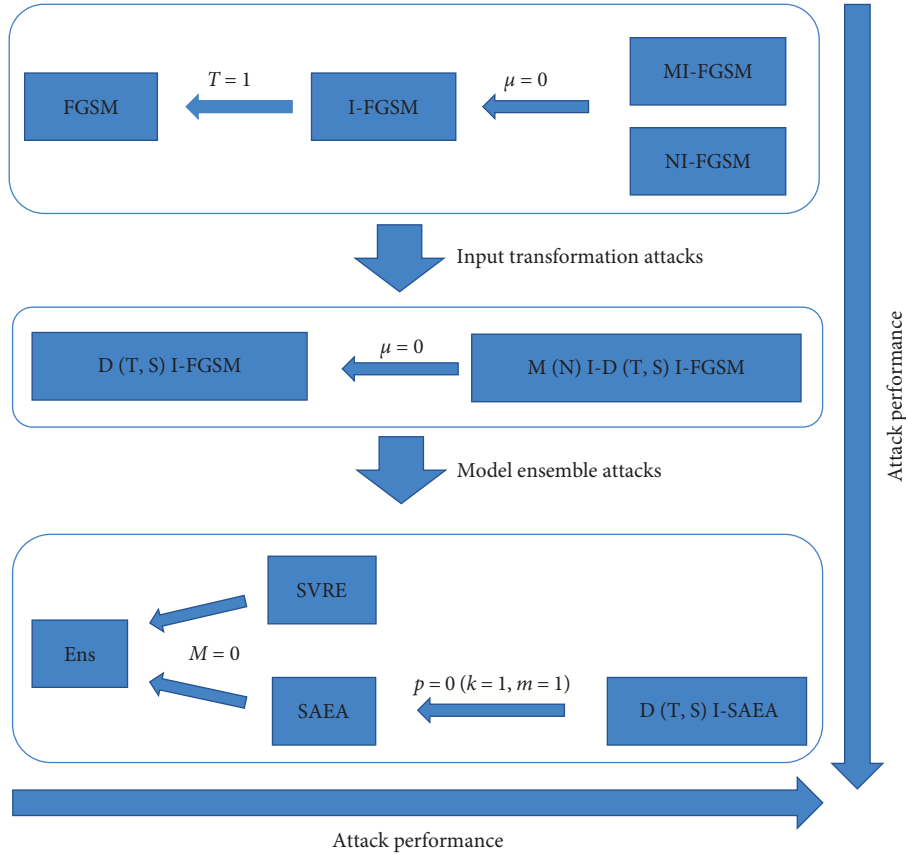


FIGURE 2: The relationships among different attack methods, we can achieve different types of attacks by adjusting hyperparameters, particularly when employing the results of multiple models for attack in input transformation attacks. This attack method will become a model ensemble attack.

in SVRE, where the internal step size β are set to 1.6, and the number of internal iterations is set to 16.

4.2. Result on Single Representative Model. We first compare SAEA with Ens and SVRE on four representative models, including Inc-v3, Inc-v4, IncRes-v2, and Res-101. We test the effectiveness of the attack with four models ensembled and with one model excluded, respectively.

Table 1 shows the ASR of the three methods on the ensemble four models and hold-out corresponding models (the best results are marked in bold). While our method is slightly inferior to Ens and SVRE in the white box environment, it outperforms both in the black box environment, indicating higher transferability. Moreover, transferability will be further improved when combined with other methods.

4.3. Result on Ensemble-Trained Models. We then compare the ASR of adversarial examples on three models that have undergone ensemble training, including Inc-v3_{ens3}, Inc-v3_{ens4}, and IncRes-v2_{ens}. The adversarial examples are crafted on the representative models, which are Inc-v3, Inc-v4, Res-101, and IncRes-v2. By combining Ens, SVRE, and SAEA with multiple attack baseline methods, we test the transferability of the crafted adversarial examples on these defense models.

Table 2 shows the ASR of adversarial examples on the ensemble-trained model when hold out the corresponding model, Table 3 shows the ASR of adversarial examples

against ensemble-trained models on the four representative models (the best results are marked in bold). We can see that our method outperforms Ens and is better than SVRE in most cases on various baselines. The greatest improvement is observed when combined with the TIM baseline, where our method SAEA-TIM achieves a success rate improvement of 17.6% over Ens-TIM and 3.4% over SVRE-TIM, demonstrating stronger transferability.

4.4. Result on Advanced Defense Models. In this section, we select seven advanced input transformation defense methods, including JPEG [29], RP [30], HGD [31], FD [32], ComDefend [33], RS [34], and NIPS-r3 to evaluate our method. The adversarial examples are crafted on the representative models, including Inc-v3, Inc-v4, Res-101, and IncRes-v2. For R&P, ComDefend, and JPEG, we use Inc-v3_{ens3} as the verification model. For RS, we set the failure probability to 0.001 and the hyperparameter σ to 0.5. For other methods, we use the official models proposed in their respective papers. Table 4 shows the results (the best results are marked in bold). We can see that, under each combination with the baseline method, our proposed method achieves the best performance on most defense methods, especially combined with the TI-DIM method. For the average success rate on these seven models, our method is 10.5% higher than Ens and 3.2% higher than SVRE, demonstrating strong transferability and the ability to deal with various defense mechanisms.

TABLE 1: The ASR (%) on the four representative models, which are Inc-v3, Inc-v4, IncRes-v2, and ResNet-101.

Method	Ensemble (white-box)				Hold-out (black-box)				
	Inc-v3	Inc-v4	IncRes-v2	Res-101	Inc-v3	Inc-v4	IncRes-v2	Res-101	
I-FGSM	Ens	99.9	99.7	99.8	99.8	78.1	68.0	58.0	49.3
	SVRE	99.9	99.8	99.8	99.6	87.8	82.7	76.3	62.5
	SAEA	99.8	99.6	98.6	98.3	85.7	78.7	73.5	65.1
MI-FGSM	Ens	99.8	99.8	99.5	99.5	90.8	85.5	81.6	77.7
	SVRE	99.9	99.6	99.3	99.8	96.6	93.2	91.2	86.6
	SAEA	100	99.2	99.4	97.7	96.6	93.4	91.3	88.7
TIM	Ens	99.8	99.7	99.5	99.2	91.8	88.4	83.9	78.8
	SVRE	99.5	99.5	99.0	99.7	93.9	91.9	86.8	77.4
	SAEA	99.8	99.3	98.7	97.5	94.8	91.5	87.9	85.0
TI-DIM	Ens	99.5	99.4	98.9	97.6	95.3	94.2	93.7	90.3
	SVRE	99.9	99.1	99.4	99.2	98.3	97.2	95.9	90.9
	SAEA	100	99.7	99.5	98.8	98.1	97.7	96.7	95.4
SI-TI-DIM	Ens	99.8	99.5	99.5	99.2	97.6	97.1	97.1	96.3
	SVRE	99.8	99.7	99.6	99.8	99.2	98.8	98.2	96.7
	SAEA	99.9	99.8	99.7	99.5	99.4	98.9	98.4	98.4

The adversarial examples are also crafted on these models. The adversarial examples on the left are crafted on all of the four models, while those on the right are crafted on hold-out corresponding models.

TABLE 2: The ASR (%) on ensemble-trained models, with adversarial examples crafted on the other three representative models, excluding the respective model.

Method	Hold-out model					
	Inc-v3	Inc-v4	IncRes-v2	Res-101	Average	
I-FGSM	Ens	20.93	19.03	17.57	17.30	18.71
	SVRE	29.70	28.33	27.03	22.33	26.85
	SAEA	30.90	28.43	27.13	24.47	27.73
MI-FGSM	Ens	42.10	38.87	42.10	36.70	38.28
	SVRE	49.27	44.40	42.30	38.47	43.61
	SAEA	49.30	44.27	41.13	41.33	44.03
TIM	Ens	63.73	60.30	63.73	59.23	59.38
	SVRE	74.70	75.10	69.90	66.13	71.46
	SAEA	77.17	75.53	71.50	76.50	75.18
TI-DIM	Ens	79.10	77.40	79.10	76.27	77.97
	SVRE	89.50	88.70	85.60	81.83	86.41
	SAEA	92.87	91.17	89.87	92.03	91.48
SI-TI-DIM	Ens	92.63	98.60	92.63	92.57	92.26
	SVRE	96.13	96.13	94.20	93.57	95.06
	SAEA	97.10	96.40	95.53	96.83	96.47

4.5. *Results on Baseline Method.* Finally, we validate our approach on separate baseline methods, including DIM, TIM, SIM, and Admix. We use four representative models as surrogate models and both the representative models and the ensemble-trained models as target models. The results of the experiments are shown in Table 5 (the best results are marked in bold). We can find that there is an advantage of our method over a variety of baseline methods, representing the effectiveness of our method.

4.6. Ablation Study

4.6.1. *Internal Iteration M .* The number of internal iterations M determines the variance between the ensemble gradient

and the individual model gradients. We compare SAEA combined with five baseline methods with internal iteration rounds that are multiples of 4; when the internal iteration round is 0, the SAEA becomes the Ens. The ASR on ensemble-trained models is shown in Figure 3. As internal iteration M increases, the ASR of the five methods gradually increases. Each method achieves the best performance at different internal iterations. We can also see that when the internal iterations exceed a certain value, the adversarial examples will cause overfitting on the four surrogate models.

4.6.2. *Internal Step Size β .* The internal step size β determines the degree of update for each internal image. We integrate SAG with three different baselines, fixing external step size α at 1.6, and varied β between 0.1 and 25.6, as shown in

TABLE 3: The ASR (%) on ensemble-trained models and the adversarial examples are crafted on the representative models.

	Method	Black-box setting			Average
		Inc-v3 _{ens3}	Inc-v4 _{ens4}	IncRes-v2 _{ens}	
I-FGSM	Ens	25.8	24.2	16.0	22.00
	SVRE	40.1	35.4	24.5	33.33
	SAEA	42.1	40.1	26.0	36.07
MI-FGSM	Ens	50.4	49.3	32.2	43.97
	SVRE	64.7	59.0	38.1	53.93
	SAEA	66.1	58.8	40.1	55.00
TIM	Ens	73.2	68.6	59.3	67.03
	SVRE	84.9	82.5	76.3	81.23
	SAEA	88.0	86.2	79.7	84.63
TI-DIM	Ens	87.4	84.3	77.6	73.10
	SVRE	94.0	93.0	88.3	91.77
	SAEA	96.5	94.6	92.4	94.50
SI-TI-DIM	Ens	95.1	94.7	92.3	94.03
	SVRE	98.1	97.9	95.2	97.07
	SAEA	99.0	97.9	96.3	97.73

TABLE 4: The ASR (%) on seven defense models, with adversarial examples crafted on the four representative models.

	Attack	JPEG	RS	HGD	FD	NIPS-r3	R&P	ComDefend	Average
I-FGSM	Ens	40.4	23.8	26.4	26.7	15.1	29.0	34.0	27.91
	SVRE	58.6	25.4	45.4	39.7	25.3	43.9	49.5	41.11
	SAEA	59.5	26.2	43.1	42.5	26.1	44.5	50.8	41.81
MI-FGSM	Ens	74.4	29.6	40.6	64.2	33.5	55.9	66.8	52.14
	SVRE	88.4	32.1	46.7	77.3	40.3	69.4	81.7	62.27
	SAEA	87.3	32.2	43.7	78.8	41.0	69.6	81.3	61.99
TIM	Ens	84.4	36.4	73.1	78.3	59.3	73.6	73.7	68.4
	SVRE	90.0	45.2	84.0	90.7	75.8	84.9	82.3	78.99
	SAEA	91.6	46.0	85.6	91.7	79.0	86.6	85.3	79.02
TI-DIM	Ens	91.7	40.8	86.6	89.0	79.8	87.4	85.9	79.22
	SVRE	86.5	51.4	93.4	96.3	90.8	94.9	92.5	86.54
	SAEA	97.0	56.2	95.1	97.1	92.9	95.9	94.1	89.76
SI-TI-DIM	Ens	96.8	57.9	95.9	96.8	93.0	95.9	94.9	90.17
	SVRE	98.7	62.5	97.6	98.9	95.9	98.5	96.7	92.69
	SAEA	99.4	66.3	98.2	99.5	96.2	98.8	97.7	93.73

TABLE 5: The ASR (%) when combine with the baselined method.

Method	Representative models (white-box)				Ensemble trained models (black-box)			
	Inc-v3	Inc-v4	IncRes-v2	Res-101	Inc-v3 _{ens3}	Inc-v3 _{ens4}	IncRes-v2 _{ens}	
DIM	Ens	99.5	99.5	99.1	98.1	69.2	64.6	48.3
	SVRE	99.8	99.8	99.7	99.7	79.5	74.1	53.8
	SAEA	99.9	99.7	99.8	98.7	85.7	80.8	62.1
TIM	Ens	99.8	99.7	99.5	99.2	73.2	68.6	59.3
	SVRE	99.5	99.5	99.0	99.7	84.9	82.5	76.3
	SAEA	99.8	99.3	98.7	97.5	88.0	86.2	79.7
SIM	Ens	100	100	99.7	99.7	87.3	83.2	70.6
	SVRE	99.9	99.9	100	99.8	90.2	87.9	70.8
	SAEA	99.9	99.9	99.9	99.9	90.6	88.5	72.5
Admix	Ens	100	100	99.8	99.9	89.2	84.2	71.2
	SVRE	99.9	99.8	99.9	100	91.4	88.7	72.1
	SAEA	100	99.9	99.9	99.7	91.2	88.8	73.5

The adversarial examples are crafted on four representative models.

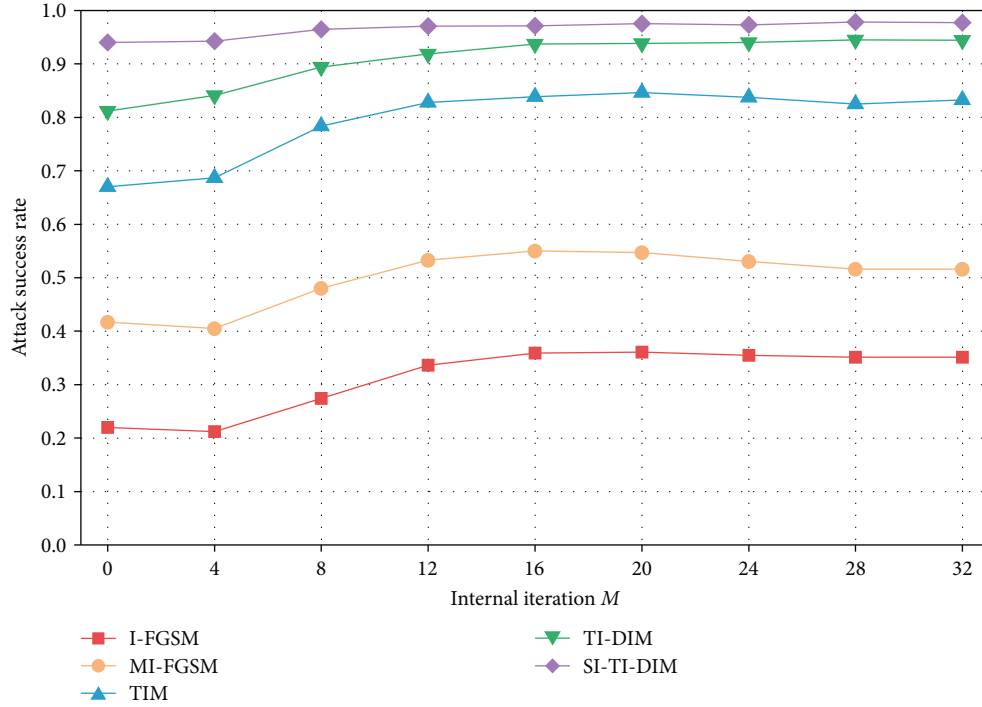


FIGURE 3: The ASR of SAEA combined with five baseline methods at different internal iteration numbers M .

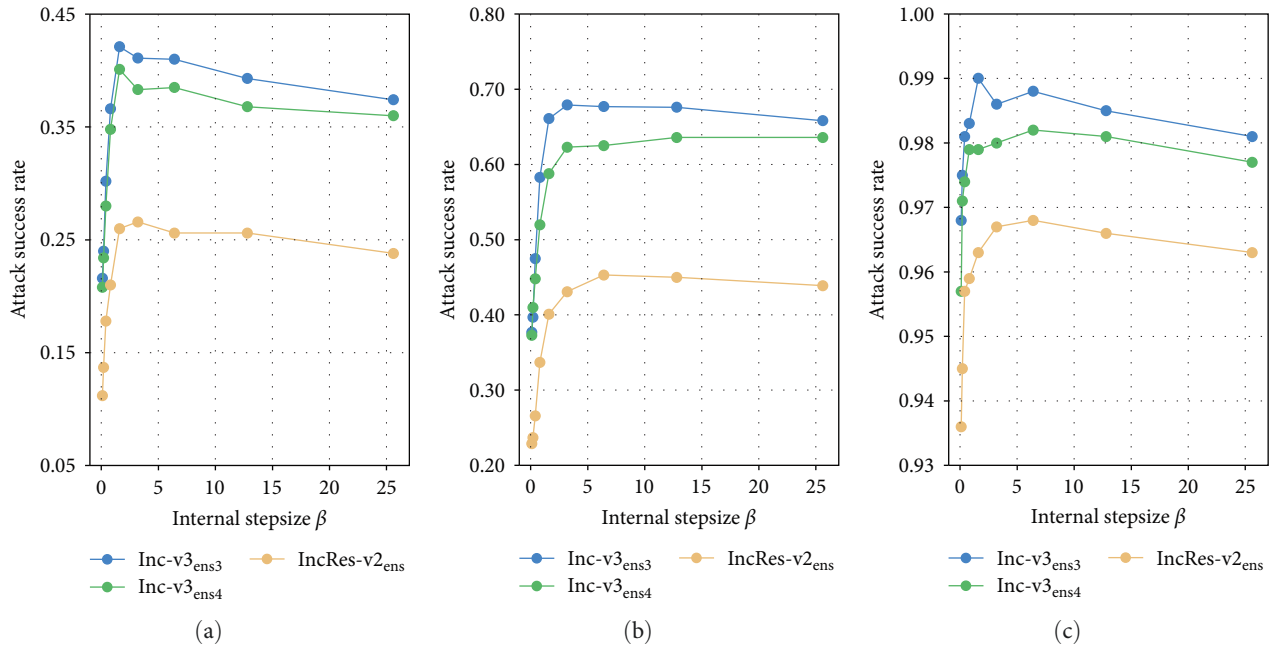


FIGURE 4: The ASR of SAEA combines with different baselines at different internal step sizes: (a) I-FGSM; (b) MI-FGSM; and (c) SI-TI-DIM.

Figure 4, the ASR (%) of various methods reaches the maximum at different step sizes. To make better comparisons with other methods, we also set β to 1.6.

4.7. *The Calculate Times.* As SAEA and SVRE both have an internal loop, they require more computational resources than ENS when compared at the same number of iterations.

To better assess the effect of the gradient computation times on the experimental results, we compare SAEA with SVRE under the same number of computations. In the case of the MI-FGSM baseline method with 16 internal loops, SVRE needs to compute the gradient on the ensemble model once in each external loop, which can be viewed as computing the gradient of each single model four times. In addition,

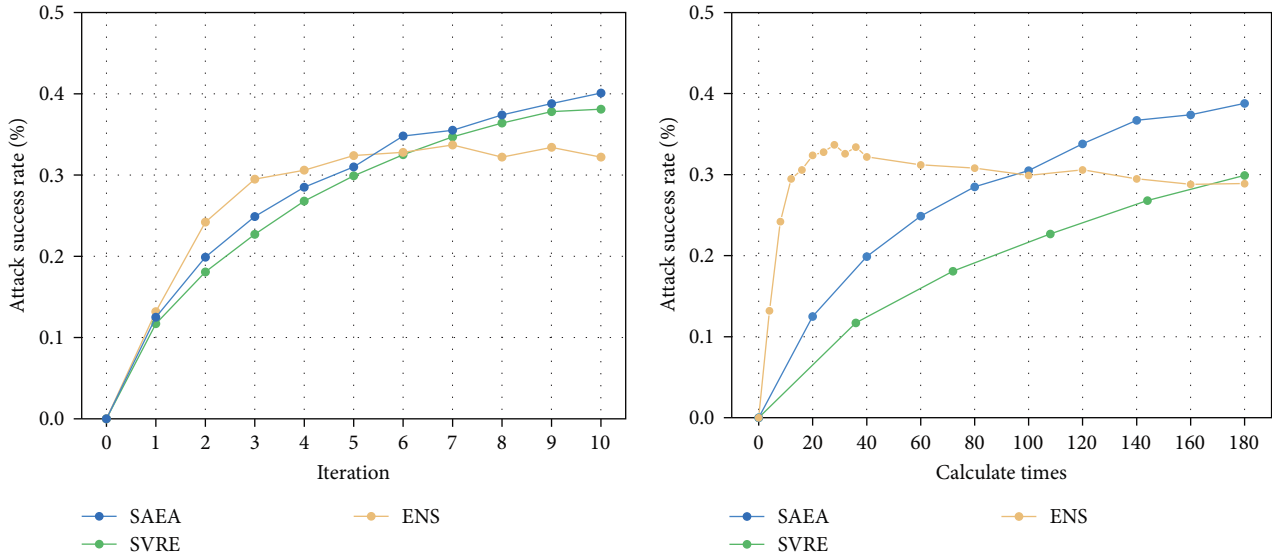


FIGURE 5: The ASR with SAEA, ENS, and SVRE under different calculated costs.

32 gradient computations are required in 16 internal loops, making a total of 36 computations for SVRE. In contrast, SAEA directly aggregates the gradients without the need for additional gradient computations and only requires one-time ensemble gradient computation in the external loop and one gradient computation per internal loop for a total of 20 computations. Therefore, the computational cost of SVRE is 1.8 times that of SAEA. Figure 5 shows the experimental results of the three methods under different computational expenses; we can see that ENS is able to generate adversarial examples faster, but it can only exhibit a limited success rate of transfer attacks. In addition, SAEA requires less computational resources compared to SVRE, and the final generated adversarial examples have the highest transferability among the three methods, which shows the effectiveness of our method.

5. Conclusion

In this work, we propose a new attack method called SAEA, aiming to craft more transferable adversarial examples. SAEA generates adversarial examples by reducing the difference between the average gradient and the gradients of individual models. In addition, SAEA can be combined with other methods to further improve attack capability, and extensive experiments show that our method crafts more transferable adversarial examples than existing methods. However, there are still some limitations in our approach; for example, the inner loops increase the computational cost and the time to generate adversarial examples, which performs poorly in some attack scenarios with strict time requirements. In future work, we will try to optimize this problem, consider introducing our approach to multimodal attacks, and try to experiment on more datasets.

Data Availability

The data and code can be found at <https://github.com/LeizhaoYNU/SAEA>.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the Youth Project for Basic Research of Yunnan Province Science and Technology Department (no. 202301AU070194), the Fundamental Research Funds for the Central Universities (no. 2042022kf0021), and the Science and Technology Plan in Key Fields of Yunnan Province (no. 202202AD080002).

References

- [1] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," arXiv preprint arXiv: 1704.02532, 2017.
- [2] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "Deepface: closing the gap to human-level performance in face verification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, IEEE, Columbus, OH, USA, 2014.
- [3] S. Guo, T. Zhang, G. Xu, H. Yu, T. Xiang, and Y. Liu, "Topology-aware differential privacy for decentralized image classification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 6, pp. 4016–4027, 2022.
- [4] L. Kuang, F. Zeng, D. Liu, H. Cao, H. Jiang, and J. Liu, "LipAuth: securing smartphone user authentication with lip motion patterns," *IEEE Internet of Things Journal*, vol. 11, no. 1, pp. 1096–1109, 2024.

- [5] H. Jiang, H. Cao, D. Liu, J. Xiong, and Z. Cao, "SmileAuth: using dental edge biometrics for user authentication on smartphones," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 3, pp. 1–24, 2020.
- [6] C. Szegedy, W. Zaremba, I. Sutskever et al., "Intriguing properties of neural networks," arXiv preprint arXiv:1312.6199, 2013.
- [7] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," arXiv preprint arXiv:1412.6572, 2014.
- [8] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57, IEEE, San Jose, CA, USA, 2017.
- [9] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," arXiv preprint arXiv: 1605.07277, 2016.
- [10] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," In International Conference on Learning Representations, 2016.
- [11] Y. Dong, F. Liao, T. Pang et al., "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9185–9193, IEEE, 2018.
- [12] Y. Xiong, J. Lin, M. Zhang, J. E. Hopcroft, and K. He, "Stochastic variance reduced ensemble adversarial attack for boosting the adversarial transferability," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14983–14992, IEEE, 2022.
- [13] N. Roux, M. Schmidt, and F. Bach, "A stochastic gradient method with an exponential convergence rate for finite training sets," *Advances in Neural Information Processing Systems*, vol. 25, 2012.
- [14] O. Russakovsky, J. Deng, H. Su et al., "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, pp. 211–252, 2015.
- [15] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Artificial intelligence Safety and Security*, pp. 99–112, Chapman and Hall/CRC, 2018.
- [16] J. Lin, C. Song, K. He, L. Wang, and J. E. Hopcroft, "Nesterov accelerated gradient and scale invariance for adversarial attacks," In International Conference on Learning Representations
- [17] D. Jang, S. Son, and D. S. Kim, "Strengthening the transferability of adversarial examples using advanced looking ahead and self-cutmix," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 148–155, IEEE, 2022.
- [18] C. Xie, Z. Zhang, Y. Zhou et al., "Improving transferability of adversarial examples with input diversity," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2730–2739, IEEE, 2019.
- [19] Y. Dong, T. Pang, H. Su, and J. Zhu, "Evading defenses to transferable adversarial examples by translation-invariant attacks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4312–4321, IEEE, 2019.
- [20] X. Wang, X. He, J. Wang, and K. He, "Admix: enhancing the transferability of adversarial attacks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16138–16147, IEEE, 2021.
- [21] Y. Long, Q. Zhang, B. Zeng et al., "Frequency domain model augmentation for adversarial attack," in *European Conference on Computer Vision*, pp. 549–566, Springer Nature Switzerland, Cham, 2022.
- [22] J. Zhang, J. T. Huang, W. Wang et al., "Improving the transferability of adversarial samples by path-augmented method," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8173–8182, IEEE, 2023.
- [23] Y. Zhang, H. Li, G. Xu, S. Yuan, and X. Huang, "One radish, one hole: specific adversarial training for enhancing neural network's robustness," *Peer-to-Peer Networking and Applications*, vol. 14, no. 4, pp. 2262–2274, 2021.
- [24] 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.
- [25] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: attacks and defenses," In International Conference on Learning Representations, 2018.
- [26] X. Li, K. Kong, S. Xu, P. Qin, and D. He, "Feature selection-based android malware adversarial sample generation and detection method," *IET Information Security*, vol. 15, no. 6, pp. 401–416, 2021.
- [27] W. Xu, D. Evans, and Y. Qi, "Detecting adversarial examples in deep neural networks," arXiv preprint arXiv: 1704.01155, 2017.
- [28] G. Cohen, G. Sapiro, and R. Giryes, "Detecting adversarial samples using influence functions and nearest neighbors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14453–14462, IEEE, 2020.
- [29] C. Guo, M. Rana, M. Cisse, and L. Van Der Maaten, "Countering adversarial images using input transformations," arXiv preprint arXiv: 1711.00117, 2017.
- [30] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," International Conference on Learning Representations, 2018.
- [31] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, "Defense against adversarial attacks using high-level representation guided denoiser," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1778–1787, IEEE, 2018.
- [32] Z. Liu, Q. Liu, T. Liu et al., "Feature distillation: DNN-oriented jpeg compression against adversarial examples," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 860–868, IEEE, 2019.
- [33] X. Jia, X. Wei, X. Cao, and H. Foroosh, "Comdefend: an efficient image compression model to defend adversarial examples," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6084–6092, IEEE, 2019.
- [34] J. Cohen, E. Rosenfeld, and Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *International Conference on Machine Learning*, pp. 1310–1320, PMLR, 2019.
- [35] X. Wang and K. He, "Enhancing the transferability of adversarial attacks through variance tuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1924–1933, IEEE, 2021.
- [36] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," *Advances in Neural Information Processing Systems*, vol. 26, 2013.
- [37] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, IEEE, 2016.
- [38] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.