Generalizable Black-Box Adversarial Attack with Meta Learning

Fei Yin*, Yong Zhang*, Baoyuan Wu*†, *Member, IEEE,* Yan Feng, Jingyi Zhang, Yanbo Fan, Yujiu Yang†, *Member, IEEE*

Abstract—In the scenario of black-box adversarial attack, the target model's parameters are unknown, and the attacker aims to find a successful adversarial perturbation based on query feedback under a query budget. Due to the limited feedback information, existing query-based black-box attack methods often require many queries for attacking each benign example. To reduce query cost, we propose to utilize the feedback information across historical attacks, dubbed example-level adversarial transferability. Specifically, by treating the attack on each benign example as one task, we develop a meta-learning framework by training a meta generator to produce perturbations conditioned on benign examples. When attacking a new benign example, the meta generator can be quickly fine-tuned based on the feedback information of the new task as well as a few historical attacks to produce effective perturbations. Moreover, since the meta-train procedure consumes many queries to learn a generalizable generator, we utilize model-level adversarial transferability to train the meta generator on a white-box surrogate model, then transfer it to help the attack against the target model. The proposed framework with the two types of adversarial transferability can be naturally combined with any off-the-shelf query-based attack methods to boost their performance, which is verified by extensive experiments. The source code is available at https://github.com/SCLBD/MCG-Blackbox.

Index Terms—Black-box Adversarial Attack, Meta Learning, Example-level and Model-level Adversarial Transferability, Conditional Distribution of Perturbation

1 Introduction

Deep neural networks (DNNs) have been shown to be vulnerable to adversarial examples [1], where stealthy and malicious perturbations are added onto benign examples to fool the DNN model. According to the accessible information about the attacked DNN model ($i.e.$, the target model), existing adversarial attacks can be generally categorized into two scenarios. The first is *white-box attack*, which assumes that the attacker knows the parameters of the target model, such that the adversarial perturbation can be easily generated based on the gradient of the target model. The second is *black-box attack*, where the attacker does not know the parameters of the target model, while only the query feedback is accessible. Compared to the white-box scenario, the black-box scenario is more practical and more challenging. Thus, this work focuses on the black-box scenario, especially on the *score-based* black-box attack, where the feedback is a continuous score ($i.e.$, the posterior probability in classification problem).

The general procedure of attacking one benign example in the score-based black-box attack scenario can be described as follows: given a query budget ($i.e.$, the allowed query number) and an allowed perturbation region in the vicinity of the attacked benign example, with the starting solution at the benign example, the attacker keeps searching for a successful perturbation that satisfies the attack goal for a black-box target model; if such a successful perturbation is found within the allowed perturbation region under the query budget, then the attack is successful and stopped. Otherwise, the attack failed.

As image statistics differ in benign images, attacking a benign image can be viewed as an individual task where the feedback information from the target model serves as the supervision to guide the perturbation generation. This perspective inspires us to utilize the information across different tasks to learn generic prior knowledge that can be transferred to boost the performance of each individual task, as did in meta-learning. The intuition is that, when attacking more benign examples, the attacker is supposed to be more experienced in how to generate perturbation conditioned on a given benign image and also know the target model better. Consequently, compared to the fresh attacker, one experienced attacker is expected to find a successful perturbation with fewer queries when attacking a new benign example. The underlying rationale is that adversarial perturbations around different benign examples may have some similar properties, and we call it **example-level adversarial transferability**. One typical example is universal adversarial perturbations [2], where one perturbation may fool multiple benign examples simultaneously. However, to the best of our knowledge, example-level adversarial transferability has not been proposed explicitly to boost the attack performance, especially in the scenario of black-box attack. To capture the above intuition, here we propose to utilize meta-learning to learn a meta generator across different attacking tasks ($i.e.$, attacking different benign examples), dubbed *Meta Conditional Generator (MCG)*, which encodes the prior into the parameters of the network and can produce adversarial perturbations based on the benign example accordingly. When attacking a new benign example, the meta generator can be quickly fine-tuned based on the information of the benign example as well as the feedback information of a few historical attacks to produce effective perturbations that are specific

- *Fei Yin, Yong Zhang and Baoyuan Wu are co-first authors.*
- *Baoyuan Wu (*wubaoyuan@cuhk.edu.cn*) and Yujiu Yang (*yang.yujiu@sz.tsinghua.edu.cn*) are corresponding authors.*
- *Fei Yin, Yan Feng and Yujiu Yang are with Tsinghua Shenzhen International Graduate School, Tsinghua University. Baoyuan Wu is with School of Data Science, Shenzhen Research Institute of Big Data, the Chinese University of Hong Kong, Shenzhen. Yong Zhang and Yanbo Fan are with Tencent AI Lab. Jingyi Zhang is with the Center for Future Media and the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China.*

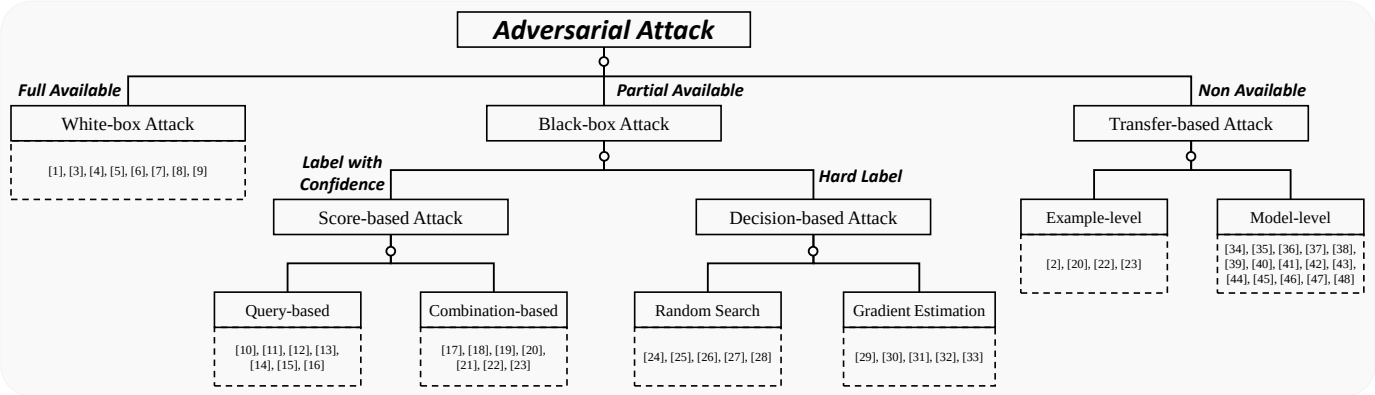


Fig. 1. Taxonomy structure of existing black-box adversarial attack methods.

to the new benign example.

However, directly training the meta generator with the perturbations of successful attacks in the meta-train process may still require many queries to the target model. To further reduce the number of queries, we resort to the widely used **model-level adversarial transferability**, which assumes that some shared terms can be transferred from white-box surrogate models to the target model to help the black-box attack. Specifically, we propose to conduct meta training based on white-box surrogate models, then the learned generator is transferred to help the attack against the target model, which serves as the *meta-test* process. Besides, to mitigate the difference between the surrogate and target models, we also fine-tune the surrogate model during the meta-test process by minimizing the feedback difference between the surrogate and target models for the same query, which encourages the surrogate model to mimic the behaviour of the target model and makes the generated perturbation more specific to the target model. The updated surrogate model is then exploited to fine-tune the meta generator for adaptation. In short, inspired by the perspective of meta-learning, we propose a general meta-learning framework for black-box adversarial attacks, by utilizing two levels of adversarial transferability, including example-level and model-level.

One prominent advantage of the proposed framework is that it can be naturally combined with any off-the-shelf black-box attack methods to boost their original performance. For example, for sampling-based methods, the meta generator can serve as the sampling distribution; for random-search-based methods that gradually adjust the perturbation, the meta generator can provide a suitably initialized perturbation. Extensive experiments on benchmark datasets and against several state-of-the-art attack methods verify the superiority of the proposed attack framework.

The main contributions of this work are three-fold. **1)** We propose to treat the black-box attack to each benign example as an individual task, which inspires us to utilize the information across different tasks to boost the attack performance of each task. **2)** We develop a general meta-learning framework for the black-box attack scenario by utilizing both the example- and model-levels of adversarial transferability. **3)** Extensive experiments demonstrate that the proposed framework can be naturally combined with any existing query-based black-box attack methods and significantly boost their original performance.

2 Related Work

As shown in Fig. 1, we partition existing adversarial attack methods into three categories at the first levels, including **white-box methods** which utilize the full information ($i.e.$, parameters) of the attacked model to generate perturbations, **black-box methods** which utilize the query feedback returned by the attacked model, and **transfer-based methods** which don't utilize any information of the attacked model ($i.e.$, target model). Their detailed reviews are presented in the following sub-sections.

2.1 White-box Adversarial Attack

The white-box attack problem is generally formulated as follows:

$$\max_{\boldsymbol{\delta}\in\mathbb{B}_\epsilon(\boldsymbol{x})} \mathcal{L}(f(\boldsymbol{x}+\boldsymbol{\delta}), t), \tag{1}$$

where $\boldsymbol{\delta}$ represents the adversarial perturbation, $\mathbb{B}_\epsilon(\boldsymbol{x}) = \{\boldsymbol{x}' - \boldsymbol{x} | \ \|\boldsymbol{x}' - \boldsymbol{x}\|_p \leq \epsilon\}$ denotes a neighboring region around $\boldsymbol{x}$, and the attacker-specified scalar $\epsilon > 0$ represents the upper bound of allowed perturbations. The loss function $\mathcal{L}(f(\boldsymbol{x}+\boldsymbol{\delta}), t)$ could be specified as the cross entropy loss in untargeted attack where t indicates the ground-truth label of $\boldsymbol{x}$, while as the negative cross entropy in targeted attack where t indicates a target label that is different from the ground-truth label of $\boldsymbol{x}$. Many gradient-based optimization methods have been utilized to solve the above problem, including the fast gradient sign method (FSGM) [1], iterative fast gradient sign method (I-FGSM) [3], C&W method [4], functional adversarial attack [5], adversarial camouflage [6], $etc.$ Moreover, several works [7], [8], [9] focus on the adversarial attack in the physical world, where many types of environmental distortions may weaken the effectiveness of the generated adversarial perturbations.

2.2 Black-box Adversarial Attack

The formulation of Problem (1) is also applicable to the black-box attack. However, the parameters of the target model $f(\cdot)$ is unknown, while only the objective value $f(\boldsymbol{x}+\boldsymbol{\delta})$ for each query $\boldsymbol{x}+\boldsymbol{\delta}$ is provided. Consequently, the gradient-based optimization methods cannot be directly utilized. According to the type of query feedback $f(\boldsymbol{x}+\boldsymbol{\delta})$, black-box attacks can be further partitioned to two sub-categories, including score-based and decision-based attacks.

2.2.1 Score-based Black-box Adversarial Attack

In the scenario of score-based attack, the feedback $f(\boldsymbol{x}+\boldsymbol{\delta})$ is continuous, such as the posterior probability in the image classification task. Existing methods for solving this problem can be generally partitioned into three categories, including query-based and combination-based methods. **1) Query-based methods** iteratively adjust the perturbation only based on queries to the target model. Many black-box optimization approaches have been utilized, mainly including random search ($e.g.$, [10], Square, SignHunter [11]), evolution strategies ($e.g.$, NES [12], Bandit [13]), and gradient-estimation approaches ($e.g.$, ZOO [14], AutoZoom [15], ZO-signSGD [16]). Compared to transfer-based methods, query-based methods often achieve a higher attack success rate, but with the cost of many queries to the target model. **2) Combination-based methods** aim to achieve a high attack access rate and high query efficiency simultaneously, by taking advantage of both queries to the target model and the transferred item from the surrogate model. Existing methods proposed different kinds of transferring strategies, such as transferring the perturbation magnitude ($e.g.$, Square [17]), perturbation gradient ($e.g.$, [18], [19] and Meta attack [20]), perturbation distribution ($e.g.$, $\mathcal{N}$ATTACK [21], AdvFlow [22]), the projection to a low-dimensional space ($e.g.$, TREMBA [23]). Combination-based methods have shown superior attack performance to the other two categories, and our method also belongs to this category. However, most existing methods (only except Meta attack [20]) only utilized the model-level adversarial transferability, while our method captures both example-level and model-level transferability to improve the query efficiency further.

2.2.2 Decision-based Black-box Adversarial Attack

In the scenario of decision-based attack, the feedback $f(\boldsymbol{x}+\boldsymbol{\delta})$ is discrete, such as the class label in the image classification task. Existing methods for solving this problem can be generally partitioned into random search based and gradient-estimation-based methods. **Random search based methods** aim to find the best perturbation around the invisible decision boundary, such as sampling from a normal distribution in Boundary method [24] or from a learnable Gaussian distribution in Evolutionary method [25], searching along the estimated normal direction of the decision boundary in GeoDA [26], or searching on the surface of the allowed perturbation region in the vicinity of the benign example in SFA [27] and Rays [28]. **Gradient-estimation-based methods** propose different estimation approaches of gradient, such as utilizing the neighboring points around the current solution in NES [12] and qFool [29], Monte Carlo estimation in HopSkipJumpAttack [30], or estimating the gradient in a low-dimensional subspace for acceleration in QEBA [31]. OPT [32] and Sign-OPT [33] were developed based on a continuous formulation that alternatively optimizes the magnitude and direction of perturbation, such that any gradient-estimation approaches can be utilized.

2.3 Transfer-based Adversarial Attack

We list the transfer-based methods as another category, as each transfer-based method could be applied for the white-box attack or the black-box attack. For example, (MI-FGSM) [34] and Nesterov iterative fast gradient sign method (NI-FGSM) [35] can be used as white-box attacks, since they are extensions of the classic white-box attacks FSGM [1] and I-FGSM [3]. However, as claimed in the manuscripts of [34] and [35], their goals are to generate more transferable perturbations to improve the attack success rate in the black-box settings, and their experiments contain both white-box and black-box settings. Thus, if one method mainly aims to improve the adversarial transferability, we partition it to the transfer-based category, rather than white-box or black-box. According to the transfer's objective, we further present example-level and model-level transferability, respectively.

2.3.1 Example-level adversarial transferability

Although without explicit illustration, some works have actually studied the example-level transferability. For example, universal adversarial perturbations [2] revealed that it is possible to find a perturbation to fool multiple benign examples simultaneously, with respect to the same attacked model. It reveals that different benign examples may have some common fragile directions, following which adversarial perturbations can be found easily. Another example is generation-based adversarial attacks [23], [22], where a generative model is trained to directly generate an adversarial perturbation or adversarial example for each benign example. Its default assumption is that adversarial perturbations around different benign examples may follow the same distribution or mapping. However, the example-level adversarial transferability has been rarely utilized to boost the attack performance, especially in the scenario of black-box attack. The only attempt we have found is called Meta attack [20], where a meta attacker is trained to generate gradient, which is then used as the update direction in gradient-estimation-based black-box attack methods. In contrast, our proposed meta-learning framework aims at capturing the distribution of adversarial perturbations, thus can be naturally combined with any kinds of query-based attack methods.

2.3.2 Model-level adversarial transferability

The model-level adversarial transferability has been observed and studied in many existing works [36], [37], [23], [38], [39], [40], [41]. Two important issues are mainly explored about the model-level transferability, including the intrinsic reason, and how to enhance the transferability across models, especially in the black-box scenario. **1) The intrinsic reason of the model-level transferability**. Tramèr *et al.* [40] found that different models share a large fraction of adversarial subspace, which consists of orthogonal basis vectors that are highly aligned with the gradient of the loss function. Consequently, perturbations within this shared subspace are likely to be transferable across models. A geometric perspective provided by [37] showed that the transferability is partially due to the fact that decision boundaries of different models align well with each other. Demontis *et al.* [42] demonstrated that the model-level transferability is closely related to the intrinsic vulnerability of the target model, and the complexity of the surrogate model. The theoretical analysis provided in [41] derives two lower bounds for transferability based on data distribution similarity and model gradient similarity, as well as the upper bound for transferability based on gradient orthogonality and smoothness. Wang *et al.* [43] illustrated that the model-level transferability is negatively correlated with the interaction inside adversarial perturbations. **2) Enhancing the model-level transferability**. Compared to FGSM [1], its extensions, including momentum iterative fast gradient sign method (MI-FGSM) [34] and Nesterov iterative fast gradient sign method (NI-FGSM) [35], showed better model-level transferability. In [44], MI-FGSM and NI-FGSM were further extended by reducing the variance of the iterative update directions, such that the update direction is stabilized to escape from poor local optima, leading to the transferability improvement

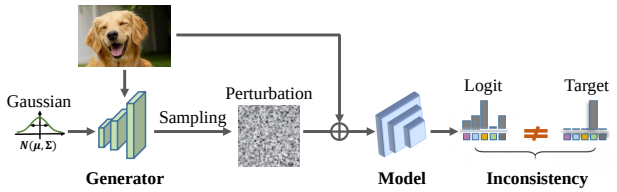


Fig. 2. Task definition in meta-learning. Given an image and a target model, the goal is to learn a generator that can generate an effective adversarial example to attack the target model.

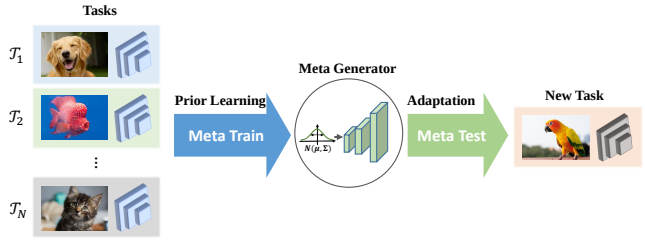


Fig. 3. Overview of meta-learning. During the meta-train phase, a meta generator can be obtained by training on a large set of tasks, which contains the generic prior of how to generate effective perturbations for different images to attack the target model. During the meta-test phase, the meta generator can be quickly adapted to new tasks with only a few steps of fine-tuning.

even when there are defenses for the target model. The ensemble attack method [37] generated adversarial perturbations based on multiple models to enhance the transferability for both untargeted and targeted attacks. Intermediate level attack (ILA) [45] proposed to generate adversarial perturbations based on intermediate layers of surrogate models to avoid overfitting, such that the transferability to the target model can be enhanced. Feature distribution attack (FDA) [46] focused on improving the transferability of a targeted attack by maximizing the activation of one intermediate layer between the benign example and the perturbed example. It is extended in [38] from one intermediate layer to multiple intermediate layers, such that the transferability of targeted attack is further enhanced. Some works employed the meta-learning ideology to enhance transferability either. MSM [47] obtained a Meta-Surrogate Model via optimizing a differentiable attacker. The Meta-surrogate model gained prior from one or a set of surrogate models and was able to generate adversarial examples with eximious transferability. Meta Gradient [48] randomly sampled multiple models from a model zoo to compose different tasks and iteratively simulated a white-box attack and a black-box attack in each task, which narrowed the gap between the gradient directions in white-box and black-box attacks.

3 The Proposed Approach

3.1 Problem Formulation

We denote the classification model as $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$, where the model parameters $\boldsymbol{\theta}$ are unknown in the black-box attack setting, and $\mathcal{X}$ and $\mathcal{Y}$ are the input and output spaces, respectively. Given an input example $\boldsymbol{x}$, the index of its ground-truth label is denoted as y; $f_{\boldsymbol{\theta}}(\boldsymbol{x}, i) \in [0, 1]$ indicates the posterior probability *w.r.t.* the i-th class, and $f_{\boldsymbol{\theta}}(\boldsymbol{x}, i)$ denotes the corresponding logit. Our attack goal is to generate an adversarial perturbation $\boldsymbol{\delta}$ for the benign example $\boldsymbol{x}$ to fool the model $f_{\boldsymbol{\theta}}(\cdot)$, given that the model parameters $\boldsymbol{\theta}$ are unknown, dubbed *score-based black-box adversarial attack*. It can be generally formulated as the following optimization problem:

$$\min_{\boldsymbol{\delta} \in \mathbb{B}_\epsilon(\boldsymbol{x})} \mathcal{L}^{\text{adv}}(\boldsymbol{\delta}, \boldsymbol{x}, y) = \begin{cases} \max(0, \triangle_{ut}), & \text{if } \textit{untargeted attack} \\ \max(0, \triangle_t), & \text{if } \textit{targeted attack} \end{cases} \tag{2}$$

where $\triangle_{ut} = f_{\boldsymbol{\theta}}(\boldsymbol{x} + \boldsymbol{\delta}, y) - \max_{j \neq y} f_{\boldsymbol{\theta}}(\boldsymbol{x} + \boldsymbol{\delta}, j)$, and $\triangle_t = \max_{j \neq y} f_{\boldsymbol{\theta}}(\boldsymbol{x} + \boldsymbol{\delta}, j) - f_{\boldsymbol{\theta}}(\boldsymbol{x} + \boldsymbol{\delta}, t)$, with $t \in \mathcal{Y}$ being the target label. Note that $\mathcal{L}^{\text{adv}}$ is *non-negative*, and if 0 is achieved, then the corresponding $\boldsymbol{\delta}$ is a successful adversarial perturbation.

3.2 Meta Conditional Generator for Black-Box Attack

Conditional Perturbation Generator. Unlike most previous combination-based methods that use a deterministic network to predict the initial perturbation for a benign image, our generator captures a conditional distribution of perturbation conditioned on the benign image, *i.e.,* $p(\boldsymbol{\delta}|\boldsymbol{x}; \boldsymbol{\varphi})$, where $\boldsymbol{\delta} = G_{\boldsymbol{\varphi}}(\boldsymbol{z}; \boldsymbol{x})$. G is the generator with $\boldsymbol{\varphi}$ as its parameters. $\boldsymbol{\delta}$ is the perturbation and $\boldsymbol{z}$ is a random vector that follows a simple distribution, *e.g.,* Gaussian distribution. In the conditional distribution, effective perturbations are supposed to have a high probability of being sampled. When attacking the target model, we can sample a perturbation $\boldsymbol{\delta} \sim p(\boldsymbol{\delta}|\boldsymbol{x}; \boldsymbol{\varphi})$ and add it to the benign image to construct an adversarial example to fool the target model.

Modeling Conditional Distribution. The generator captures a conditional distribution of perturbation, which can be realized with using a simple distribution and a complex non-linear function that maps the simple distribution to a complex one with an image as the condition. In this work, we use a conditional generative flow (c-Glow) [49] as the generator due to its superior property that the mapping between a random vector and the output perturbation is invertible. By using c-Glow, we have $\boldsymbol{\delta} = g_{\boldsymbol{\varphi}}(\boldsymbol{z}; \boldsymbol{x})$ and the inverse version $\boldsymbol{z} = g^{-1}_{\boldsymbol{\varphi}}(\boldsymbol{\delta}; \boldsymbol{x})$. The random vector $\boldsymbol{z}$ follows a Gaussian distribution, *i.e.,* $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Since c-Glow consists of a set of invertible functions/layers, the parameters $\boldsymbol{\varphi}$ can be decomposed into several individual parts, *i.e.,* $g = g_{\boldsymbol{x},\varphi_1} \circ \cdots \circ g_{\boldsymbol{x},\varphi_M}$. M is the number of layers and φ_i represents the parameters of the i-th layer. With the change of variables [50], we can write the conditional likelihood as

$$\log p(\boldsymbol{\delta}|\boldsymbol{x}; \boldsymbol{\varphi}) = \log p(\boldsymbol{z}) + \sum_{i=1}^{M} \log \left| \det\left(\frac{\partial g^{-1}_{\varphi_i}(\boldsymbol{r}_{i-1}; \boldsymbol{x})}{\partial \boldsymbol{r}_{i-1}}\right) \right|, \tag{3}$$

where $\boldsymbol{r}_i = g_{\varphi_i}(\boldsymbol{r}_{i-1}; \boldsymbol{x})$, $\boldsymbol{r}_0 = \boldsymbol{x}$, and $\boldsymbol{r}_M = \boldsymbol{z}$.

Perspective of Meta Learning. Attacking on one benign image can be viewed as an individual process where the generator can be optimized by sampling several perturbations from the conditional distribution and obtaining their corresponding feedback scores from the target model as supervision. However, when attacking a black-box model, the budget of query is always limited, *i.e.,* we can only sample a few perturbations. Hence, the learning of the generator in each attacking process can be formulated as a few-shot learning problem, *i.e.,* given $\{\boldsymbol{\delta}_i, f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), y_i\}_{i=1}^{K}$, the goal is to learn the generator $G_{\boldsymbol{\varphi}}(\boldsymbol{z}; \boldsymbol{x})$. K is the number of shots.

As mentioned in Sec. 1, adversarial perturbations around different benign images may share certain common properties, *i.e.,* example-level adversarial transferability. Inspired by the concept of "learning to learn" in meta-learning, we can solve the few-shot learning problem from the perspective of meta-learning, and learn a meta generator to capture the common properties of perturbations by performing a large set of attacking tasks. The prior of how

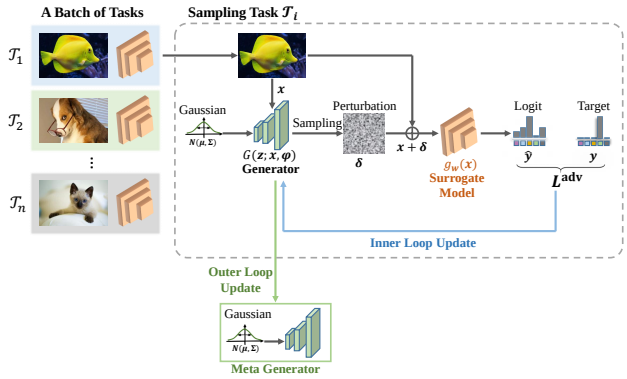


Fig. 4. The pipeline of meta training. The batch version of REPTILE is exploited for meta training. We sample a batch of tasks and perform the inner loop update of task-specific parameters for each task. Then the task-specific parameters of all tasks in the batch are aggregated to do the outer loop update, *i.e.,* updating the meta parameters.

to generate a conditional distribution of perturbation around a benign image is learned across various and diverse tasks. Since the perturbation distributions of different benign images are generated through the same generator, the prior is implicitly encoded in the parameters of the generator.

Task Definition. Task is the atom in meta-learning. In the scenario of adversarial attack, "task" is defined as: "*given a benign image and a target model, the goal is to learn a conditional generative model from which a sampled perturbation could successfully fool the target model.* " As shown in Fig. 2, given a benign image, we can sample a perturbation from the generative model. The addition of the benign image and the perturbation yields an adversarial example which is then fed into the target network for attack. The adversarial example is effective if its the predicted label is not consistent with the ground truth label (*i.e.,* untargeted attack) or the predicted label of the adversarial example is the same as a specified label (*i.e.,* targeted attack). Hence, the parameters of the generator are updated by maximizing the difference between the prediction and the ground truth or minimizing the difference between the prediction and the specified label.

Since the parameters and architecture of the target model are unknown, inspired by the transferability of adversarial example [36], we use a surrogate model to replace the target model in the meta-train phase. As shown in Fig. 3, we can achieve a meta generator by performing a large set of tasks, which captures the prior of generating adversarial perturbations and is adaptive to different tasks during the meta-test phase.

3.3 Meta Training with A Surrogate Model

As described in Sec. 3.2, in a task $\mathcal{T}$, given a benign image $\boldsymbol{x}$ and a target model $f_{\boldsymbol{\theta}}(\boldsymbol{x}, \cdot)$, the goal is to learn a conditional perturbation generator $G_{\boldsymbol{\varphi}}(\boldsymbol{z}; \boldsymbol{x})$ that can generate an effective perturbation $\boldsymbol{\delta}$ to fool the target model, *e.g.,* $f_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta}, y) < \max_{j \neq y} f_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta}, j)$ for untargeted attack and $f_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta}, t) > \max_{j \neq t} f_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta}, j)$ for targeted attack, where $\boldsymbol{\delta} = G_{\boldsymbol{\varphi}}(\boldsymbol{z}; \boldsymbol{x})$. The meta training process is illustrated in Fig. 4. In this work, the target black-box model is the same for different tasks.

In the scenario of black-box attack, we have no access to the parameters and the architecture of the target model. Hence, we have to make many queries for each benign image to generate successful perturbations and then use them to learn the meta generator, which is costly for query and hard to realize in real-world scenarios. Based on the model-level adversarial transferability, in the meta-train phase, the target model is replaced by a surrogate model $g_{\boldsymbol{w}}(\boldsymbol{x})$ of which the architecture and parameters are available. Therefore, the gradients can backpropagate through the surrogate model to update the generator. Since the meta-train phase does not involve any target model, once the meta training is over, the meta generator can be applied to any target model in the meta-test phase.

To evaluate the effectiveness of the generated perturbation, the adversarial loss function is similar to that in Eq. (2). The difference is that $\tilde{\triangle}_{ut}$ and $\tilde{\triangle}_{t}$ are computed by using the surrogate model rather than the target model, *i.e.,*

$$\begin{aligned} \tilde{\triangle}_{ut} &= g_{\boldsymbol{w}}(\boldsymbol{x}+\boldsymbol{\delta}, y) - \max_{j \neq y} g_{\boldsymbol{w}}(\boldsymbol{x}+\boldsymbol{\delta}, j), \\ \tilde{\triangle}_{t} &= \max_{j \neq y} g_{\boldsymbol{w}}(\boldsymbol{x}+\boldsymbol{\delta}, j) - g_{\boldsymbol{w}}(\boldsymbol{x}+\boldsymbol{\delta}, t), \end{aligned} \tag{4}$$

where t is the specified target class in targeted attack.

Given a set of tasks $\{\mathcal{T}_i\}_{i=1}^{N}$, we follow the batch version of REPTILE [51] to perform meta-learning. We sample n tasks to form a batch and update the task-specific parameters k times using Adam [52] for each task. The objective of the inner loop optimization is $\boldsymbol{\phi}_{\mathcal{T}_i} = \arg\min_{\boldsymbol{\phi}_{\mathcal{T}_i}} \mathcal{L}_{\mathcal{T}_i}^{\text{adv}}$. The optimization procedure can be represented as

$$\boldsymbol{\phi}_{\mathcal{T}_i} = \text{Adam}(\mathcal{L}_{\mathcal{T}_i}^{\text{adv}}, \boldsymbol{\varphi}, k, \alpha), \tag{5}$$

where $\boldsymbol{\phi}_{\mathcal{T}_i}$ is the final task-specific parameters of the generator after k steps of performing Adam, starting from $\boldsymbol{\varphi}$. At each of the k steps, a perturbation is sampled from the current conditional distribution. $\mathcal{L}_{\mathcal{T}_i}^{\text{adv}}$ is the adversarial loss of the i-th task. α is the learning rate of the inner loop.

Then, for the outer loop optimization, we can update the meta parameters of the generator with the resulted task-specific parameters in a mini-batch, *i.e.,*

$$\boldsymbol{\varphi} \leftarrow \boldsymbol{\varphi} + \beta \frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{\phi}_{\mathcal{T}_i} - \boldsymbol{\varphi}), \tag{6}$$

where β is the learning rate of the outer loop.

3.4 Meta Test Using Historical Attack Experience

The standard meta-test process is not applicable in black-box attack because the target model is unknown and the gradient cannot be backpropagated to fine-tune the meta generator through it. To solve this issue in the meta-test, we propose to transfer the information of the black-box target model to a surrogate model by using the feedback of previous attacks to fine-tune the surrogate model. The usage of the historical attack experience could make the surrogate imitate the behaviour of the target network, which provides more accurate and effective information to fine-tune the meta generator than directly using the surrogate model. The pipeline of meta-test is illustrated in Fig. 5. Given a new benign image, it consists of three steps to conduct the attack to the target model, *i.e.,* fine-tuning the surrogate model for introducing the information of the target model, fine-tuning the meta generator for adaptation to the new benign image, and boosting off-the-shelf black-box attack methods. The attacking ability of the whole framework is gradually improved in the process of circulation.

Fine-tuning the Surrogate Model. In Fig. 5(b), to use historical attack experience, the predicted logits of previous benign images, their adversarial examples, and the current benign image by the

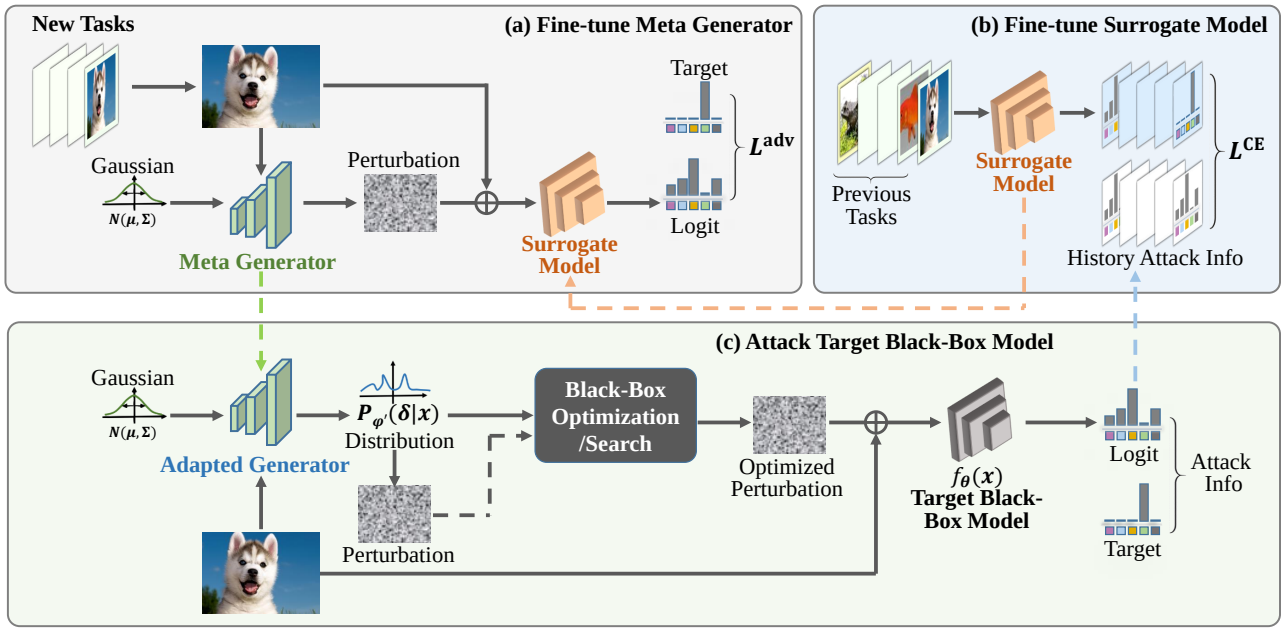


Fig. 5. The pipeline of meta test. (a) Fine-tune the meta generator. Given the image of the new task and the updated surrogate model, the meta generator is fine-tuned by performing the inner optimization with $\mathcal{L}^{\text{adv}}$. (b) Fine-tune the surrogate model. In order to transfer the information of the target black-box model to the surrogate model, the historical information (*i.e.,* logits of adversarial examples and the target labels) of previous tasks as well as the current task are used to make the surrogate model mimic the behaviour of the target model. (c) Attack the target black-box model. The adapted generator is combined with the off-the-shelf black-box attack methods. The generator can provide an initial distribution of perturbation or a sampled perturbation according to the given image. The initialization is leveraged by the attack methods as the starting state to get a refined perturbation. Then, the generated adversarial example is used to attack the target model. The logits of the attack are recorded to fine-tune the surrogate model in stage (b).

target model are collected to provide supervision for the fine-tuning of the surrogate model. The loss function is defined as

$$\begin{aligned}\mathcal{L}^{\text{CE}} = \; & CE(g_{\boldsymbol{w}}(\boldsymbol{x}+\boldsymbol{\delta}), f_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta})) \\ & + CE(g_{\boldsymbol{w}}(\boldsymbol{x}), f_{\boldsymbol{\theta}}(\boldsymbol{x}))\end{aligned} \tag{7}$$

where $CE(\cdot)$ is the cross entropy loss function. The two terms represent the losses of the adversarial example and the benign image, respectively. For the benign image in the current task, only the second term is used. The optimization of the parameters of the surrogate model can be represented as

$$\boldsymbol{w}' = \text{Adam}(\sum_{i}^{m} \mathcal{L}_{\mathcal{T}_i}^{\text{CE}}, \boldsymbol{w}, s, \lambda), \tag{8}$$

where $\boldsymbol{w}'$ represents the parameters of the updated surrogate model. λ is the learning rate and m is the number of tasks in a mini-batch. $\mathcal{L}_{\mathcal{T}_i}^{\text{CE}}$ is the loss for the i-th task. s is the number of update steps.

Fine-tuning the Meta Generator. Fig. 5(a) presents the fine-tuning of the meta generator with using the updated surrogate model $g_{\boldsymbol{w}'}(\boldsymbol{x})$ for the benign image of the new task. The fine-tuning procedure in the meta-test phase is similar to the inner optimization in the meta-train phase. We use the loss in Eq. (4) and Eq. (5) to update the parameters of the meta generator, *i.e.,*

$$\boldsymbol{\phi}_{\mathcal{T}} = \text{Adam}(\mathcal{L}_{\mathcal{T}}^{\text{adv}}, \boldsymbol{\varphi}, k, \alpha), \tag{9}$$

where $\boldsymbol{\phi}_{\mathcal{T}}$ represents the adapted parameters for the new task. Different from Eq. (5), the updated surrogate model is used in Eq. (9) to yield out the loigts for the adversarial examples, *i.e.,* $g_{\boldsymbol{w}'}(\boldsymbol{x}+\boldsymbol{\delta})$. As the updated surrogate model contains the transferred information from the target model, it can provide more accurate and effective supervision to fine-tune the meta generator than the original surrogate model. Hence, the generated perturbation is more specific to the target model.

Boosting Off-the-shelf Black-Box Attack Methods. Our adapted generator can provide an initial distribution of perturbation or a perturbation conditioned on the given benign image, enabling it be combined with other off-the-shelf black-box attack methods to boost their original performance. Fig. 5 (c) shows the process of attacking the target model. The initial distribution or perturbation is leveraged by a black-box attack method as the starting state. The further optimized perturbation is then added to the benign image to generate an adversarial example. The output logits from the target model are recorded as historical attack information, which is then used to fine-tune the surrogate model. When combined with sampling-based methods [12], [53], the adapted generator served as a distribution. When combined with random-search-based methods [10], [11], [17], a perturbation can be a sample from the generator and serves as an initial state.

4 Experiments

4.1 Experimental Settings

Datasets. To demonstrate the effectiveness of the proposed method, we conduct comprehensive experiments on two commonly used benchmark databases, *i.e.,* CIFAR-10 [54] and ImageNet [55]. Following the setting in [53], for the CIFAR-10 dataset, we randomly select $1,000$ images from the testing set for evaluation which cover all classes evenly. The images are resized to 32×32. For the ImageNet dataset, we first randomly select 10 classes from the $1,000$ classes and then use the 500 images of each class from the validation set for evaluation. The images are resized to 224×224. The target and surrogate models are trained on the

training set of the corresponding dataset. On CIFAR-10, we use the full training set for meta-learning to learn the meta generator. On ImageNet, the training set of the 10 chosen classes are used for meta-learning.

Evaluation. We select l_∞-based attacks and set the maximal distortion as $\epsilon = 0.031$ for CIFAR-10 and $\epsilon = 0.05$ for ImageNet with image pixel values re-scaled to $[0, 1]$. We set the maximal query budget to 10,000 times in all experiments. If the attacker cannot successfully fool the target m odel within the query limit, we consider it a failure case. Following the prior work [53], we adopt the attack success rate (ASR), the mean query number (Mean), and the median query number (Median) of successful attacks to evaluate the attack performance.

Target and Surrogate Models. On CIFAR-10, we consider four target models: ResNet-Preact-110 [56], DenseNet-121 [57], VGG-19 [58], and PyramidNet-110 [59]. We follow the standard training process of image classification to obtain the checkpoints of these target models. The top-1 error rates of these four target models are 6.29%, 6.17%, 7.28%, and 7.51% on the standard testing set, respectively. On ImageNet, we also evaluate our method on four target models: ResNet-18 [56], VGG-16-BN [58], WRN-50 [60], and InceptionV3 [61]. We use the official implementation of these methods and download their pre-trained checkpoints from torchvision. The top-1 error rates of these target models are 28.41%, 30.24%, 21.53%, and 22.71% on the validation set of ImageNet, respectively. In all experiments, ResNet-18 [56] and ResNet-50 [56] are used as the surrogate models on CIFAR-10 and ImageNet, respectively. The corresponding top-1 error rates of the surrogate models are 6.37% for ResNet-18 and 23.97% for ResNet-50.

To further verify the performance of our framework, we also conduct experiments of attacking black-box adversarial defense models on ImageNet, including JPEG-Compression-WideResNet-50 [62], Small-Noise-Defense-WideResNet-50 [63], FreeAdv-ResNet-50 [64], and FastAdv-ResNet-50 [65].

Competing Methods. As our framework can provide a good initialization of perturbation, it can be treated as a plug-and-play component that can be combined with other black-box attack methods to boost their performance. In order to verify the versatility of our model, we combine our framework with 6 query-based black-box attack methods, including NES [12], CG-Attack [53], SimBA [10], SignHunter [11], Square [17], and MetaAttack [20]. For search-based methods such as SignHunter, Square, SimBA, and MetaAttack, a sampled perturbation from the generator is the initialization. For sampling-based methods such as NES and CG-Attack, a distribution of perturbation represented by the generator is the initialization.

We also compare with several transfer-based attack methods to verify the transferability of the proposed method, including PGD [66], MI [34], TIMI [67], and DI [68], under the transfer attack setting where only the information of the surrogate model is accessible. Moreover, we finally compare with several combination-based methods under the query attack setting, including AdvFlow [22] and TREMBA [23]. They exploit both the queries to the target model and the transferred item from the surrogate model. To ensure the fairness of comparison, we retrain the combination-based methods with the same surrogate model as ours. All the experiments are implemented with the source code provided by their authors under the same setting.

Implementation Details. Following [49], in all the experiments, we adopt the same architecture for the generator c-Glow with 3 blocks composed of 8 flow steps. Each block starts with a squeeze operation followed by 8 flow steps and ends with a split operation. To improve the efficiency, we adopt the discrete cosine transform (DCT) and inverse DCT for dimension reduction by downsampling the size of images in ImageNet to $\frac{1}{8} \times \frac{1}{8}$ lower frequency subspace before feeding them into the generator. On CIFAR-10, we use the original shape of images as they are in a small size.

Before meta training, we pre-train the c-Glow to provide an initial state of modelling the distribution of perturbation. We use the surrogate model to generate a large set of perturbations through PGD attack with the perturbation strength of $\ell_{\inf} = 0.05$, the step size of 0.01, and the number of iterations of 50. The parameters are optimized by maximizing the log-likelihood, *i.e.,* $\max_{\boldsymbol{\varphi}} \log p(\boldsymbol{\delta}|\boldsymbol{x}; \boldsymbol{\varphi})$.

For the pre-training of the generator, the learning rate is set to 0.001. The batch size is $m = 16$. The generator is trained for 10 epochs. For meta training, we sample 16 tasks every batch. The update stepsize of the inner optimization is set to $k = 4$. The learning rates of the inner and outer loops are set to $\alpha = 0.0003$ and $\beta = 0.0006$, respectively. For meta-test, when fine-tuning the surrogate model, we freeze the parameters of all layers except the last three layers. The surrogate model is fine-tuned with both benign images and their adversarial examples. The learning rate of fine-tuning is set to $\lambda = 0.0003$. The number of benign images is set to 4 in a batch. When fine-tuning meta generator, the process is similar to the inner loop optimization of the task-specific parameters.

4.2 Experiments in Closed-set Attack Scenario

In the closed-set attack scenario, the surrogate and target models are trained on the same training set, *i.e.,* both the training images and the categories are the same. Experiments includes boosting the off-the-shelf black-box attack methods, attacking defended models, comparisons with transfer-based methods, and comparisons with combination-based methods.

Performance on CIFAR-10. The specification of the surrogate model and the target models is presented in Sec. 4.1. Table 1 illustrates the results of several off-the-shelf black-box attack methods and those of the combinations with our proposed MCG.

The MCG can boost the attack efficiency of all the black-box attack methods without ASR drop under both untargeted and targeted attacks. Specifically, under the untargeted attack setting, the median query numbers of these methods are decreased to 1s for all the target models by using the initialization from our MCG, which means that we can fool the target models with the initially generated perturbations for over 50% images. Meanwhile, the ASRs are improved to nearly 100% for almost all the cases. Besides, the mean query numbers are also improved significantly. For example, for the *state-of-the-art* Square attack, our MCG can further improve its query efficiency in terms of the mean query number by a factor of 5 for all the four target models. We further plot the tendency curves of ASR *w.r.t.* the query number in Fig. 6. We can observe that the MCG can boost the attack performance under all values of query number for all the competing attack methods, especially for small query numbers.

Under the targeted attack setting, our proposed MCG can also boost the attack performance. The targeted attack is generally harder to achieve than the untargeted attack, yet our MCG still obtains a satisfactory ASR attacking all the four target models. The best attack performance is achieved by MCG+Square. Compared

TABLE 1
Closed-set evaluation on the CIFAR-10 dataset.

Target model →	ResNet-PreAct-110			DenseNet-121			VGG-19			PyramidNet-110		
Attack Method ↓	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
Untargeted Attack												
NES [12]	**100.0%**	285.2	211.0	99.4%	430.8	274.0	99.1%	822.8	421.0	**100.0%**	287.9	190.0
MCG + NES	**100.0%**	**124.1**	**1.0**	**100.0%**	**133.2**	**1.0**	**99.9%**	**371.7**	**1.0**	**100.0%**	**90.4**	**1.0**
CG-Attack [53]	**100.0%**	230.0	81.0	**100.0%**	222.7	21.0	**100.0%**	386.4	101.0	**100.0%**	93.7	**1.0**
MCG + CG-Attack	**100.0%**	**112.3**	**1.0**	**100.0%**	**113.4**	**1.0**	**100.0%**	**262.7**	**1.0**	**100.0%**	**74.1**	**1.0**
SimBA-DCT [10]	**100.0%**	428.0	359.5	**100.0%**	449.9	352.0	**99.7%**	557.6	426.0	**100.0%**	341.1	273.5
MCG + SimBA-DCT	**100.0%**	**199.3**	**1.0**	**100.0%**	**127.0**	**1.0**	**99.7%**	**234.8**	**1.0**	**100.0%**	**114.6**	**1.0**
Signhunter [11]	**100.0%**	167.0	83.0	**100.0%**	196.6	87.0	**100.0%**	238.3	99.0	**100.0%**	128.4	63.5
MCG + Signhunter	**100.0%**	**83.3**	**1.0**	**100.0%**	**61.3**	**1.0**	**100.0%**	**157.9**	**1.0**	**100.0%**	**26.9**	**1.0**
Square [17]	**100.0%**	227.3	144.5	**100.0%**	260.3	159.0	**100.0%**	342.0	175.5	**100.0%**	165.5	100.5
MCG + Square	**100.0%**	**39.7**	**1.0**	**100.0%**	**47.6**	**1.0**	**100.0%**	**57.9**	**1.0**	**100.0%**	**29.6**	**1.0**
MetaAttack [20]	**100.0%**	642.6	632.0	99.8%	759.4	635.0	**100.0%**	686.4	633.0	**100.0%**	601.8	430.0
MCG + MetaAttack	**100.0%**	**204.6**	**1.0**	**100.0%**	**81.4**	**1.0**	**100.0%**	**188.6**	**1.0**	**100.0%**	**99.3**	**1.0**
Targeted Attack												
NES [12]	**100.0%**	774.3	610.0	99.8%	1205.2	946.0	92.4%	2738.8	1954.0	**100.0%**	889.7	694.0
MCG + NES	**100.0%**	**591.8**	**443.0**	**100.0%**	**1009.2**	**800.0**	**98.5%**	**2688.6**	**1472.0**	**100.0%**	**657.8**	**506.0**
CG-Attack [53]	**100.0%**	884.8	741.0	**99.8%**	859.7	681.0	96.8%	1476.9	901.0	**100.0%**	567.3	501.0
MCG + CG-Attack	**100.0%**	**430.8**	**181.0**	**99.8%**	**608.5**	**241.0**	**97.4%**	**944.1**	**381.0**	**100.0%**	**290.6**	**141.0**
SimBA-DCT [10]	99.6%	836.3	729.0	99.7%	944.8	842.0	98.6%	1170.3	986.0	99.8%	735.7	661.0
MCG + SimBA-DCT	**99.8%**	**664.2**	**623.5**	**99.8%**	**845.5**	**768.00**	**98.9%**	**983.3**	**861.0**	**99.9%**	**601.9**	**543.5**
Signhunter [11]	**100.0%**	386.1	272.0	**100.0%**	465.1	323.0	**100.0%**	556.3	385.5	**100.0%**	320.9	232.0
MCG + Signhunter	**100.0%**	**267.3**	**124.5**	**100.0%**	**321.0**	**181.0**	**100.0%**	**399.1**	**210.0**	**100.0%**	**167.9**	**81.0**
Square [17]	99.6%	504.7	369.0	**100.0%**	624.8	471.0	**100.0%**	827.2	593.5	**100.0%**	400.1	301.0
MCG + Square	**100.0%**	**92.3**	**20.0**	**100.0%**	**133.1**	**33.0**	**100.0%**	**141.7**	**22.0**	**100.0%**	**55.0**	**17.0**
MetaAttack [20]	**100.0%**	1174.7	899.0	**100.0%**	1294.5	1106.0	99.0%	1721.4	1106.0	**100.0%**	1065.9	890.0
MCG + MetaAttack	**100.0%**	**757.3**	**669.0**	**100.0%**	**992.9**	**887.0**	**100.0%**	**1180.7**	**882.0**	**100.0%**	**718.3**	**669.0**

TABLE 2
Closed-set evaluation on the ImageNet dataset.

Target Model →	ResNet-18			VGG-16			WRN-50			Inception-V3		
Attack Method ↓	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
Untargeted Attack												
NES [12]	99.0%	2085.5	1597.0	98.2%	1554.0	1072.0	96.5%	2366.0	1681.0	95.8%	1111.6	526.0
MCG + NES	**99.7%**	**1057.5**	**1.0**	**99.1%**	**596.5**	**1.0**	**98.4%**	**1385.1**	**1.0**	**96.7%**	**889.9**	**390.5**
CG-Attack [53]	96.3%	272.0	**21.0**	96.5%	178.3	21.0	89.6%	323.0	**21.0**	94.1%	**251.1**	**21.0**
MCG + CG-Attack	**100.0%**	**95.4**	**21.0**	**99.7%**	**69.3**	**1.0**	**99.2%**	**247.3**	**21.0**	**97.0%**	271.1	**21.0**
SimBA-DCT [10]	**100.0%**	520.6	381	98.3%	486.1	350.0	**94.9%**	810.6	593.0	81.9%	496.9	290.5
MCG + SimBA-DCT	98.1%	**68.7**	**1.0**	**98.8%**	**73.6**	**1.0**	94.7%	**159.6**	**1.0**	**95.0%**	**129.2**	**9.5**
Signhunter [11]	**100.0%**	60.2	23.0	**100.0%**	69.8	34.0	**100.0%**	116.0	38.5	99.4%	178.3	49.0
MCG + Signhunter	**100.0%**	**29.6**	**1.0**	**100.0%**	**19.0**	**1.0**	**100.0%**	**62.8**	**1.0**	**100.0%**	**91.8**	**8.0**
Square [17]	**100.0%**	72.8	26.0	**100.0%**	82.6	28.0	**100.0%**	136.1	68.0	99.4%	210.8	64.0
MCG + Square	**100.0%**	**31.7**	**1.0**	**100.0%**	**24.8**	**1.0**	**100.0%**	**59.9**	**1.0**	**100.0%**	**123.8**	**24.0**
MetaAttack [20]	92.2%	3302.6	2641.0	95.4%	3075.2	2213.0	87.4%	3824.2	3309.0	94.7%	2634.8	1772.0
MCG + MetaAttack	**94.1%**	**1692.3**	**1.0**	**96.8%**	**1025.9**	**1.0**	**92.5%**	**2076.8**	**1.0**	**95.3%**	**2107.1**	**1324.0**
Targeted Attack												
NES [12]	67.9%	5734.8	5860.0	79.4%	4944.1	4621.0	33.0%	6137.9	6259.0	63.3%	4921.1	4726.0
MCG + NES	**75.3%**	**5499.6**	**5294.0**	**81.0%**	**4720.9**	**4559.0**	**37.5%**	**5839.1**	**5882.0**	**66.2%**	**4687.7**	**4370.0**
CG-Attack [53]	96.9%	2553.3	1801.0	**92.9%**	2447.7	1731.0	77.8%	2976.5	2191.0	91.0%	**2260.3**	1481.0
MCG + CG-Attack	**98.0%**	**2501.8**	**1781.0**	91.9%	**2374.4**	**1721.0**	**79.8%**	**2960.2**	**2101.0**	**94.9%**	2272.1	**1441.0**
SimBA-DCT [10]	56.0%	6927.2	7196.0	71.7%	6569.9	6507.0	41.0%	6795.4	7028.0	56.8%	6094.9	6107.0
MCG + SimBA-DCT	**66.8%**	**6460.8**	**6607.0**	**79.3%**	**6023.3**	**6038.0**	**60.6%**	**5744.8**	**5626.0**	**72.8%**	**5576.9**	**5564.0**
Signhunter [11]	**100.0%**	1332.7	922.0	**100.0%**	1115.8	734.0	99.4%	1786.8	1064.0	**100.0%**	1836.2	1063.0
MCG + Signhunter	**100.0%**	**1043.8**	**596.5**	**100.0%**	**788.8**	**446.0**	**99.7%**	**1428.4**	**782.0**	99.7%	**1486.7**	**865.0**
Square [17]	**100.0%**	1097.7	819.0	**100.0%**	1112.7	819.0	99.7%	1678.7	1242.0	99.0%	1789.0	1120.0
MCG + Square	**100.0%**	**797.6**	**518.5**	**100.0%**	**784.4**	**529.0**	**100.0%**	**1148.9**	**661.0**	**99.7%**	**1455.2**	**868.0**
MetaAttack [20]	33.8%	8202.6	8041.5	17.5%	8341.4	8806.0	10.3%	8408.4	8585.0	**19.5%**	7176.8	8141.5
MCG + MetaAttack	**45.6%**	**7315.1**	**8033.0**	**38.1%**	**6425.6**	**7721.0**	**20.6%**	**7029.9**	**7596.5**	**19.5%**	**6946.5**	**7706.0**

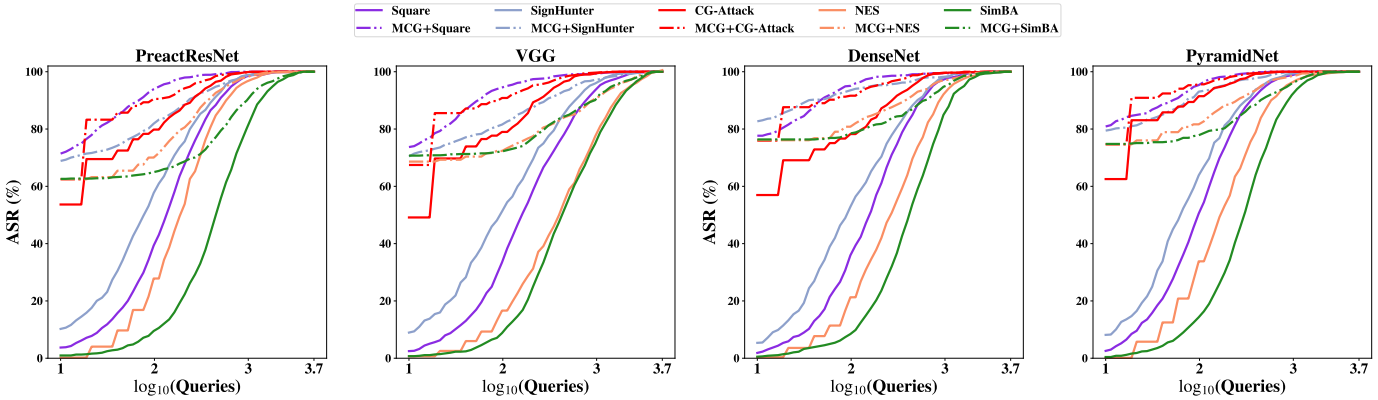


Fig. 6. Attack success rate (ASR%) *w.r.t.* the query number of untargeted attack on the CIFAR-10 dataset.

TABLE 3
Untargeted Attack against adversarial defended models on the ImageNet dataset.

Target Model → / Attack Method ↓	JPEG-Compress-WRN-50			SND-WRN-50			FastAdv-ResNet-50			FreeAdv-ResNet-50		
	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
NES [12]	13.2%	5682.1	3243.0	76.6%	3879.8	3308.5	**23.5%**	7501.5	7986.5	15.7%	7188.5	6322.0
MCG + NES	**98.9%**	**1412.1**	**1.0**	**81.7%**	**1795.7**	**1.0**	**23.5%**	**2415.4**	**715.0**	**22.4%**	**2104.1**	**685.0**
CG-Attack [53]	39.0%	2227.5	1161.0	**79.8%**	540.5	81.0	58.5%	1789.7	871.0	61.3%	2374.0	1121.0
MCG + CG-Attack	**91.7%**	**180.9**	**1.0**	**79.8%**	**331.5**	**1.0**	**64.3%**	**1634.4**	**801.0**	**64.7%**	**2305.1**	**1081.0**
SimBA-DCT [10]	14.3%	4688.3	4545.0	10.5%	608.3	22.0	**45.5%**	702.5	72.5	**52.7%**	**792.5**	245.0
MCG + SimBA-DCT	**67.1%**	**543.9**	**1.0**	**60.4%**	**498.3**	**1.0**	**45.5%**	**634.4**	**34.0**	**52.7%**	821.9	**77.0**
Signhunter [11]	**100.0%**	120.7	39.0	**89.4%**	133.4	31.0	**89.4%**	1101.5	37.5	**88.2%**	**931.3**	42.0
MCG + Signhunter	**100.0%**	**69.8**	**1.0**	89.1%	**65.6**	**1.0**	**89.4%**	**1083.5**	**36.5**	86.5%	1058.3	**40.0**
Square [17]	**100.0%**	140.9	71.0	88.9%	1487.8	154.0	**92.4%**	940.5	182.0	**91.4%**	871.3	170.0
MCG + Square	**100.0%**	**57.5**	**1.0**	**90.7%**	**307.4**	**1.0**	**92.4%**	**864.9**	**137.0**	**91.4%**	**724.9**	**123.5**
MetaAttack [20]	8.0%	3515.9	1557.0	11.7%	2449.1	1777.0	53.8%	4832.2	5063.5	55.3%	4614.9	4631.0
MCG + MetaAttack	**50.9%**	**117.2**	**1.0**	**53.1%**	**535.7**	**1.0**	**55.7%**	**4119.9**	**3969.5**	**56.3%**	**3580.2**	**3094.0**

TABLE 4
Comparison with combination-based methods on the ImageNet dataset

Target Model → / Attack Method ↓	ResNet-18			VGG-16			WRN-50			Inception-V3		
	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
Untargeted Attack												
TREMBA [23]	**100.0%**	664.5	169.0	99.1%	160.2	**1.0**	99.0%	697.6	148.0	96.9%	1232.7	211.0
AdvFlow [22]	**100.0%**	578.8	400.0	**100.0%**	693.3	420.0	**100.0%**	937.9	400.0	**97.2%**	716.3	200.0
MCG + CG-Attack	**100.0%**	**95.4**	**21.0**	99.7%	**69.3**	**1.0**	99.2%	**247.3**	**21.0**	97.0%	**271.1**	**21.0**
Targeted Attack												
TREMBA [23]	81.4%	3197.2	1997.5	91.1%	2493.5	1759.0	60.2%	5140.5	3529.0	13.3%	9462.1	9433.0
AdvFlow [22]	83.8%	4650.0	4400.0	81.6%	4407.8	4200.0	61.4%	5210.6	4980.0	81.8%	3734.1	3200.0
MCG + CG-Attack	**98.0%**	**2501.8**	**1781.0**	**91.9%**	**2374.4**	**1721.0**	**79.8%**	**2960.2**	**2101.0**	**94.9%**	**2272.1**	**1441.0**

to the original Square attack, the MCG+Square achieves the ASR of 100% with a significantly fewer number of queries. For example, the mean and median query numbers of MCG are over 4.6 times and 14.3 times less than those of the original Square attack.

Performance on ImageNet. ImageNet is a much larger dataset than CIFAR-10. The performance of both untargeted and targeted attacks on ImageNet is reported in Table 2.

We have a similar observation that the proposed MCG obtains consistent improvements when combined with different attack methods. Under the untargeted attack setting, the improvement of ASR in several cases can be apparently observed. For example, when attacking WRN-50 and VGG-16 with CG-Attack, the ASRs are 89.6% and 96.5%, respectively. Our MCG further improves CG-Attack by about 10% for attacking WRN-50 and 3% for attacking VGG-16. Besides, the efficiency improvements are also noticeable, especially for NES and SimBA. Under the targeted attack setting, the MCG brings in different degrees of improvements in almost all cases. The increase in ASR is more evident than the untargeted attack setting. Although when combined with SignHunter against InceptionV3, the ASR has been slightly reduced (0.3% lower) while the attack cost is saved near 20%.

All these results demonstrate that our MCG can provide an effective initial perturbation or a distribution of perturbation for various off-the-shelf black-box attack methods to boost their performance. Please note that the meta training procedure of the meta generator does not involve any target model or attack method. Hence, the meta generator only needs to be trained once, and it can be combined with different black-box attack methods to attack different black-box models without re-training, which corroborates the flexibility and generalization ability of the proposed framework.

Attacking Defended Models. To verify the effectiveness of the proposed method against adversarial defense models, we perform

TABLE 5
Comparison with Transfer-based methods on the ImageNet dataset.

Target model → Attack Method ↓	ResNet-18 ASR	VGG-16 ASR	WRN-50 ASR	Inception-V3 ASR
PGD [66]	35.5%	29.6%	36.6%	15.7%
PGD + MCG w/o surrogate	56.8%	70.0%	49.5%	**26.4%**
PGD + MCG w/ fixed surrogate	**57.9%**	**70.4%**	**50.1%**	**26.4%**
MI [34]	56.1%	62.0%	66.8%	26.4%
MI + MCG w/o surrogate	62.6%	71.6%	67.6%	45.1%
MI + MCG w/ fixed surrogate	**63.2%**	**73.6%**	**69.3%**	**45.4%**
TIMI [67]	56.7%	**63.4%**	**62.3%**	42.5%
TIMI + MCG w/o surrogate	66.4%	62.1%	58.3%	42.7%
TIMI + MCG w/ fixed surrogate	**66.7%**	**63.4%**	59.4%	**45.4%**
DI [68]	44.2%	42.0%	43.3%	20.1%
DI + MCG w/o surrogate	68.5%	**80.0%**	67.1%	49.6%
DI + MCG w/ fixed surrogate	**69.2%**	**80.0%**	**68.7%**	**50.7%**

experiments of attacking various defended models trained with different defense strategies, including JPEG-Compression [62], random noise perturbation [63], and adversarial training [64].

JPEG-Compression defense ($i.e.$, JPEG-Compress-WRN-50) attempts to remove the influence of adversarial examples through the compression process. Small-Noise-Defense ($i.e.$, SND-WRN-50) is specially designed for query-based attack via introducing additional random noise to hinder the attacker to estimate gradients correctly. As shown in Table 3, when attacking JPEG-Compress-WRN-50 and SND-WRN-50, the performance of NES, CG-Attack, SimBA-DCT, and MetaAttack drops sharply. In contrast, the combination with our framework greatly improves their performance in all the three metrics. For example, when attacking JPEG-Compress-WRN-50 with CG-Attack, the ASR is only 39% and the median query number is 1161, which fails in most attacks. Our framework boosts its ASR to 91.7% and reduces its median query number to 1. Besides, our method also reduces the mean and median query numbers for Square and Signhunter.

Adversarial training enlarges the difference of the classification boundaries between the robust model and the vanilla model and greatly limits the model-level adversarial transferability. As shown in Table 3, when attacking the models of adversarial training, the attack performance of all methods drops a lot compared to attacking the vanilla models. Our method can still improve the attack performance in most cases, though the improvements are not as significant as those of attacking the vanilla models.

Comparison with Transfer-based Methods. To verify the effectiveness of example-level adversarial transferability boosted by meta learning, we perform an untargeted attack experiment to compare our meta generator with several transfer-based attack methods in the transfer attack setting, $i.e.$, no information from the black-box target model is used for fine-tuning.

Our meta generator contains a pre-training stage of the c-Glow and the pre-training is performed by using an attack method to generate adversarial examples for training. The quality of the adversarial examples affects the performance a lot. In other experiments, the attack method is PGD. Since the transfer-based methods can generate better transferable adversarial examples than PGD, to fairly compare with the transfer-based methods, here we pre-train the meta generator with using the adversarial examples generated by the three transfer-based methods, respectively.

The results are shown in Table 5. 'X + MCG w/o surrogate' means that we use 'X' to pre-train the c-Glow and then directly use the meta generator to produce adversarial example without using the surrogate model. 'X + MCG w/ fixed surrogate' means that after pre-training we use the surrogate model to fine-tune the meta generator first and then use the meta generator to produce adversarial examples. As shown in Table 5, directly using our meta generator has better performance than the transfer-based methods in most cases. Fine-tuning the meta generator with the surrogate model can further improve the performance. Transfer-based methods may overfit the surrogate model due to that the adversarial example generation totally depends on the surrogate model. Differently, our meta generator captures the example-level transferability that can alleviate the overfitting issue. The generation is determined by both the learned prior of the meta generator and the surrogate model.

Comparison with Combination-based Methods. As our method can be treated as a combination of transfer-based and query-based method, we compare with two combination-based black-box attack methods, $i.e.$, TREMBA [23] and AdvFlow [22]. Since they take advantage of model-level adversarial transferability to improve the performance and are often combined with evolution methods to adjust their distribution mapping. For fairness, in the meta-test phase, we integrate the distribution adjustment method CG-Attack [53] into our framework for evaluation.

The results on ImageNet are shown in Table 4. Our method achieves the best performance under almost all attack cases in all the three metrics. Although other methods attempt to utilize the transferability between different models, they appear to be unstable in ASR when attacking different target models. For example, when attacking ResNet-18 and VGG-16 under targeted attack, the ASRs of TREMBA are 81.4% and 91.1%. But when attacking WRN-50 and Inception-V3, the ASRs drop to 60.2% and 13.3%. The results show that TREMBA cannot generalize well to attack different target models. Differently, our method can perform better in the generalization ability to attack different models. The experimental results on CIFAR-10 are presented in the Appendix Sec. B.1.

4.3 Experiments in Open-set Attack Scenario

In the closed-set attack scenario, the surrogate and target models share the same training dataset, *i.e.,* the training dataset of the target model is visible to attackers, which is hard to achieve in the real attack scenario. In real-world scenarios, the surrogate model will share less common knowledge with the target model. This situation strongly increases the difficulty of attack. Therefore, in this section, we verify the effectiveness of our method in the open-set attack scenario where the surrogate and target models are trained on disjoint training datasets, and there is no overlap between the output categories of the two models. In this open-set attack setting, it is quite challenging for the attacker to transfer the limited prior to the unknown areas. Specifically, in our experiments, we employ two datasets and train the surrogate model on one dataset and the target model on another. The training data of the meta generator is the same as the data used for the surrogate model. The results of training on CIFAR-10 and testing on CIFAR-100 are shown in Table 6. Please note that other experiments on ImageNet and OpenImage [69] are presented in the Appendix Sec. B.2.

Since the surrogate and target models are trained from different training sets with different categories, the effectiveness of model-level adversarial transferability is limited, which decreases the performance compared with the results in Table 1. Nevertheless, our framework still works in boosting the existing black-box attack methods in the open-set setting. MCG can reduce the query cost of attacks and improve the ASR in most cases, which demonstrates

TABLE 6
Untargeted Attack trained on the CIFAR-10 dataset and tested on the CIFAR-100 dataset in the open-set scenario.

Target model →	ResNet-18			VGG-16			WRN-50			Inception-V3		
Attack Method ↓	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
NES [12]	96.1%	2054.0	1555.0	84.3%	1691.0	904.0	89.5%	2229.6	1534.0	94.6%	2044.7	1471.0
MCG + NES	**97.0%**	**773.1**	**1.0**	**88.5%**	**910.5**	**1.0**	**94.0%**	**833.7**	**1.0**	**95.7%**	**1025.2**	**1.0**
CG-Attack [53]	99.4%	226.2	**1.0**	**98.1%**	294.7	**1.0**	98.4%	285.2	**1.0**	99.1%	330.8	**1.0**
MCG + CG-Attack	**99.5%**	**135.1**	**1.0**	97.8%	**244.8**	**1.0**	**98.7%**	**163.5**	**1.0**	**99.6%**	**196.5**	**1.0**
SimBA-DCT [10]	**99.2%**	222.8	84.0	96.8%	321.4	66.0	97.4%	268.1	86.5	98.5%	231.6	76.0
MCG + SimBA-DCT	**99.2%**	**119.5**	**1.0**	**97.0%**	**218.3**	**1.0**	**98.0%**	**155.2**	**1.0**	**98.9%**	**184.1**	**1.0**
SignHunter [11]	**100.0%**	88.2	28.0	**99.6%**	136.7	22.0	**99.8%**	134.5	28.0	**100.0%**	117.8	32.0
MCG + SignHunter	**100.0%**	**57.8**	**1.0**	**99.6%**	**110.2**	**1.0**	**99.8%**	**94.1**	**1.0**	**100.0%**	**92.0**	**1.0**
Square [17]	**99.9%**	103.8	17.0	**99.5%**	213.9	24.0	**99.8%**	178.1	16.0	99.8%	121.2	15.0
MCG + Square	**99.9%**	**47.9**	**1.0**	**99.5%**	**92.7**	**1.0**	**99.8%**	**83.1**	**1.0**	**99.9%**	**75.6**	**1.0**
MetaAttack [20]	**100.0%**	518.9	443.0	99.7%	591.4	443.0	99.9%	610.1	447.0	**100.0%**	523.6	445.0
MCG + MetaAttack	99.9%	**227.4**	**1.0**	**99.9%**	**396.4**	**1.0**	**100.0%**	**303.6**	**1.0**	**100.0%**	**291.4**	**1.0**

TABLE 7
Untargeted Attack against *Imagga* tagging API.

Method	Baseline			Combined with MCG		
	ASR	Mean	Median	ASR	Mean	Median
NES [12]	44.0%	35.3	21.0	**67.0%**	**8.8**	**1.0**
CG-Attack [53]	74.0%	33.5	21.0	**81.0%**	**21.4**	**1.0**
SimBA-DCT [10]	45.0%	93.8	56.0	**57.0%**	**45.5**	**1.0**
SignHunter [11]	51.0%	45.3	20.5	**82.0%**	**19.5**	**1.0**
Square [17]	49.0%	50.8	15.5	**69.0%**	**13.7**	**1.0**
MetaAttack [20]	16.0%	230.1	95.0	**61.0%**	**101.4**	**1.0**

that the meta generator can be fast-adapted to attack different target models across different datasets.

4.4 Experiments of Attacking Real-World API

To verify the effectiveness of our framework in real-world scenarios, we perform an experiment of attacking the Imagga Tagging API[1]. The model of Imagga is trained on an unknown dataset of over $3,000$ types of daily-life objects. Given a query image, the API will return a list of possible labels as well as the corresponding confidence scores. We randomly select 100 images from the validation set of ImageNet and set the query limit to 500. We define the goal of untargeted attack as removing the top-3 labels of the benign images. As the images are from ImageNet, the pre-trained surrogate model can be used to fine-tune the meta generator. $\mathcal{L}_{adv}$ is set as the maximal score of the top-3 labels. The adapted generator is then used to generate an initial perturbation for existing black-box attack methods to attack the API. As shown in Table 7, the performance of all methods is significantly improved through the combination with our framework. Specifically, the median query numbers are decreased to 1s for all methods and the mean query numbers are also highly improved. These results demonstrate that our framework is applicable in real-world scenarios.

4.5 Ablation Study

To verify the effectiveness of the meta training and the fine-tuning stages in the meta-test, we conduct experiments of untargeted attacks on ImageNet for ablation study. The results are shown in Table 8. 'Flow' is the method that directly uses the perturbations to learn a conditional glow (c-Glow) model, rather than using the adversarial loss or the parameter update strategy in meta learning. The perturbations are generated by applying Projected Gradient Descent (PGD) [66] attack to the surrogate model. During testing, no fine-tuning is conducted. 'MCG w/ fixed surrogate' means that during testing we fine-tune the meta generator with a fixed surrogate model. 'MCG w/ fine-tuned surrogate' means that during testing we update the surrogate model first by exploiting the query feedback from the target model, and then we use the updated surrogate model to fine-tune the meta generator. All these methods are combined with Square attack to query the target model. To evaluate the effectiveness of the initial perturbations, we use the first Attack Success Rate (FASR) as the metric instead of ASR. FASR means the success rate of straightforwardly using the perturbation generated by the generators to attack the target model.

TABLE 8
Ablation study on the ImageNet dataset. All methods in the table are combined with Square attack for untargeted attack.

Target Model →	ResNet-18			VGG-16		
Attack Method ↓	FASR	Mean	Median	FASR	Mean	Median
Flow	35.2%	48.6	13.0	46.1%	39.1	4.0
MCG w/ fixed surrogate	57.9%	35.3	**1.0**	70.4%	30.0	**1.0**
MCG w/ fine-tuned surrogate	**60.1%**	**31.7**	**1.0**	**71.3%**	**24.8**	**1.0**

As shown in Table 8, compared to Flow, MCG w/ fixed surrogate achieves much better performance in all the three metrics. The FASRs and the median query numbers are significantly improved. The results demonstrate the effectiveness of the meta training formulation, which improves the example-level transferability by capturing more effective generic prior of how to attack different samples. The provided initial perturbation is better than that from Flow. Moreover, compared to MCG w/ fixed surrogate, MCG w/ fine-tuned surrogate gains further improvements in the FASR and the attack efficiency, which corroborates the effectiveness of the historical attack information, *i.e.,* we can get a better-adapted generator by transferring the information of the target model to the surrogate model.

5 Conclusion

We propose a novel framework for black-box attack by formulating it as a meta-learning problem to improve the example-level adversarial transferability as well as the efficiency of attack. As the architecture and parameters of the black-box target model

1. https://imagga.com/solutions/auto-tagging

are unknown, we propose to perform the meta training with a surrogate by leveraging the model-level adversarial transferability. Since the standard meta-test process cannot be applied to the black-box attack, we propose a three-stage attack pipeline to fine-tune the meta model, including fine-tuning the surrogate model with historical attack information of the target model, fine-tuning the meta generator for the benign image with the updated surrogate model, and serving as the initialization to boost off-the-shelf black-box attack methods. Comprehensive experiments, including the closed-set and open-set scenarios as well as attacking online APIs, demonstrate the effectiveness of the proposed model.

Acknowledgement

This work was partially supported by the National Natural Science Foundation of China under Grant No.U1903213 and the Shenzhen Key Laboratory of Marine IntelliSense and Computation under Grant No.ZDSYS20200811142605016. Baoyuan Wu is supported by the Natural Science Foundation of China under grant No.62076213, Shenzhen Science and Technology Program under grants No.RCYX20210609103057050 and No.ZDSYS20211021111415025, the university development fund of the Chinese University of Hong Kong, Shenzhen under grant No.01001810, and CCF-Tencent Open Fund.

REFERENCES

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent.*, 2015.

[2] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 1765–1773.

[3] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Artificial intelligence safety and security*. Chapman and Hall/CRC, 2018, pp. 99–112.

[4] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*. IEEE, 2017, pp. 39–57.

[5] C. Laidlaw and S. Feizi, "Functional adversarial attacks," in *Proc. Adv. Neural Inform. Process. Syst.*, vol. 32, 2019.

[6] R. Duan, X. Ma, Y. Wang, J. Bailey, A. K. Qin, and Y. Yang, "Adversarial camouflage: Hiding physical-world attacks with natural styles," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 1000–1008.

[7] W. Feng, B. Wu, T. Zhang, Y. Zhang, and Y. Zhang, "Meta-attack: Class-agnostic and model-agnostic physical adversarial attack," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2021, pp. 7787–7796.

[8] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 284–293.

[9] S. T. Jan, J. Messou, Y.-C. Lin, J.-B. Huang, and G. Wang, "Connecting the digital and physical world: Improving the robustness of adversarial attacks," in *Proc. of the AAAI Conf. on Artif. Intell.*, vol. 33, no. 01, 2019, pp. 962–969.

[10] C. Guo, J. R. Gardner, Y. You, A. G. Wilson, and K. Q. Weinberger, "Simple black-box adversarial attacks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2484–2493.

[11] A. Al-Dujaili and U. O'Reilly, "Sign bits are all you need for black-box attacks," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[12] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2137–2146.

[13] A. Ilyas, L. Engstrom, and A. Madry, "Prior convictions: Black-box adversarial attacks with bandits and priors," in *Proc. Int. Conf. Learn. Represent.*, 2019.

[14] 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*, 2017, pp. 15–26.

[15] C.-C. Tu, P. Ting, P.-Y. Chen, S. Liu, H. Zhang, J. Yi, C.-J. Hsieh, and S.-M. Cheng, "Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks," in *Proc. of the AAAI Conf. on Artif. Intell.*, vol. 33, no. 01, 2019, pp. 742–749.

[16] S. Liu, P.-Y. Chen, X. Chen, and M. Hong, "signsgd via zeroth-order oracle," in *Proc. Int. Conf. Learn. Represent.*, 2018.

[17] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: a query-efficient black-box adversarial attack via random search," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 484–501.

[18] S. Cheng, Y. Dong, T. Pang, H. Su, and J. Zhu, "Improving black-box adversarial attacks with a transfer-based prior," in *Proc. Adv. Neural Inform. Process. Syst.*, vol. 32, 2019.

[19] Y. Guo, Z. Yan, and C. Zhang, "Subspace attack: Exploiting promising subspaces for query-efficient black-box attacks," in *Proc. Adv. Neural Inform. Process. Syst.*, 2019, pp. 3820–3829.

[20] J. Du, H. Zhang, J. T. Zhou, Y. Yang, and J. Feng, "Query-efficient meta attack to deep neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[21] Y. Li, L. Li, L. Wang, T. Zhang, and B. Gong, "NATTACK: learning the distributions of adversarial examples for an improved black-box attack on deep neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3866–3876.

[22] H. Mohaghegh Dolatabadi, S. Erfani, and C. Leckie, "Advflow: Inconspicuous black-box adversarial attacks using normalizing flows," in *Proc. Adv. Neural Inform. Process. Syst.*, 2020, pp. 15 871–15 884.

[23] Z. Huang and T. Zhang, "Black-box adversarial attack with transferable model-based embedding," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[24] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," in *Proc. Int. Conf. Learn. Represent.*, 2018.

[25] Y. Dong, H. Su, B. Wu, Z. Li, W. Liu, T. Zhang, and J. Zhu, "Efficient decision-based black-box adversarial attacks on face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 7714–7722.

[26] A. Rahmati, S.-M. Moosavi-Dezfooli, P. Frossard, and H. Dai, "Geoda: a geometric framework for black-box adversarial attacks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 8446–8455.

[27] W. Chen, Z. Zhang, X. Hu, and B. Wu, "Boosting decision-based black-box adversarial attacks with random sign flip," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 276–293.

[28] J. Chen and Q. Gu, "Rays: A ray searching method for hard-label adversarial attack," in *Proc. Knowledge Discovery and Data Mining*, 2020, pp. 1739–1747.

[29] Y. Liu, S. Moosavi-Dezfooli, and P. Frossard, "A geometry-inspired decision-based attack," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 4889–4897.

[30] J. Chen, M. I. Jordan, and M. J. Wainwright, "Hopskipjumpattack: A query-efficient decision-based attack," in *IEEE Symposium on Security and Privacy*. IEEE, 2020, pp. 1277–1294.

[31] H. Li, X. Xu, X. Zhang, S. Yang, and B. Li, "QEBA: query-efficient boundary-based blackbox attack," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 1218–1227.

[32] M. Cheng, T. Le, P. Chen, H. Zhang, J. Yi, and C. Hsieh, "Query-efficient hard-label black-box attack: An optimization-based approach," in *Proc. Int. Conf. Learn. Represent.*, 2019.

[33] M. Cheng, S. Singh, P. H. Chen, P. Chen, S. Liu, and C. Hsieh, "Sign-opt: A query-efficient hard-label adversarial attack," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[34] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 9185–9193.

[35] J. Lin, C. Song, K. He, L. Wang, and J. E. Hopcroft, "Nesterov accelerated gradient and scale invariance for adversarial attacks," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[36] N. Papernot, P. D. McDaniel, and I. J. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," *arXiv preprint arXiv:1605.07277*, 2016.

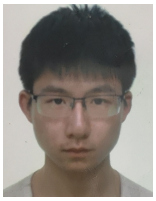
[37] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," in *Proc. Int. Conf. Learn. Represent.*, 2017.

[38] N. Inkawhich, K. J. Liang, B. Wang, M. Inkawhich, L. Carin, and Y. Chen, "Perturbing across the feature hierarchy to improve standard and strict blackbox attack transferability," in *Proc. Adv. Neural Inform. Process. Syst.*, 2020, pp. 20 791–20 801.

[39] D. Wu, Y. Wang, S. Xia, J. Bailey, and X. Ma, "Skip connections matter: On the transferability of adversarial examples generated with resnets," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[40] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "The space of transferable adversarial examples," *arXiv preprint arXiv:1704.03453*, 2017.

[41] Z. Yang, L. Li, X. Xu, S. Zuo, Q. Chen, B. Rubinstein, C. Zhang, and B. Li, "Characterizing adversarial transferability via gradient orthogonality and smoothness," in *Proc. Int. Conf. Mach. Learn. Worksh.*, 2020.
[42] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli, "Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks," in *28th USENIX security symposium*, 2019, pp. 321–338.
[43] X. Wang, J. Ren, S. Lin, X. Zhu, Y. Wang, and Q. Zhang, "A unified approach to interpreting and boosting adversarial transferability," in *Proc. Int. Conf. Learn. Represent.*, 2021.
[44] X. Wang and K. He, "Enhancing the transferability of adversarial attacks through variance tuning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2021, pp. 1924–1933.
[45] Q. Huang, I. Katsman, H. He, Z. Gu, S. Belongie, and S.-N. Lim, "Enhancing adversarial example transferability with an intermediate level attack," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 4732–4741.
[46] N. Inkawhich, K. Liang, L. Carin, and Y. Chen, "Transferable perturbations of deep feature distributions," in *Proc. Int. Conf. Learn. Represent.*, 2019.
[47] Y. Qin, Y. Xiong, J. Yi, and C.-J. Hsieh, "Training meta-surrogate model for transferable adversarial attack," *arXiv preprint arXiv:2109.01983*, 2021.
[48] Z. Yuan, J. Zhang, Y. Jia, C. Tan, T. Xue, and S. Shan, "Meta gradient adversarial attack," in *Proc. Int. Conf. Comput. Vis.*, 2021, pp. 7728–7737.
[49] Y. Lu and B. Huang, "Structured output learning with conditional generative flows," in *Proc. of the AAAI Conf. on Artif. Intell.*, 2020, pp. 5005–5012.
[50] E. Tabak and E. Vanden-Eijnden, "Density estimation by dual ascent of the log-likelihood," *Communications in Mathematical Sciences*, vol. 8, no. 1, pp. 217–233, 2010.
[51] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.
[52] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015.
[53] Y. Feng, B. Wu, Y. Fan, L. Liu, Z. Li, and S. Xia, "Boosting black-box attack with partially transferred conditional adversarial distribution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 15 095–15 104.
[54] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.
[55] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, 2015.
[56] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 630–645.
[57] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 2261–2269.
[58] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015.
[59] D. Han, J. Kim, and J. Kim, "Deep pyramidal residual networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 6307–6315.
[60] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Brit. Mach. Vis. Conf.*, 2016.
[61] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 2818–2826.
[62] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, "Countering adversarial images using input transformations," in *Proc. Int. Conf. Learn. Represent.*, 2018.
[63] J. Byun, H. Go, and C. Kim, "Small input noise is enough to defend against query-based black-box attacks," *Proc. Int. Conf. Learn. Represent.*, 2021.
[64] A. Shafahi, M. Najibi, M. A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!" in *Proc. Adv. Neural Inform. Process. Syst.*, 2019, pp. 3353–3364.
[65] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," in *Proc. Int. Conf. Learn. Represent.*, 2019.
[66] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. Int. Conf. Learn. Represent.*, 2018.
[67] Y. Dong, T. Pang, H. Su, and J. Zhu, "Evading defenses to transferable adversarial examples by translation-invariant attacks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 4312–4321.
[68] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille, "Improving transferability of adversarial examples with input diversity," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 2730–2739.
[69] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov *et al.*, "The open images dataset v4," *Int. J. Comput. Vis.*, pp. 1956–1981, 2020.



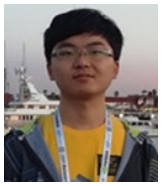
Fei Yin is currently a master student in Tsinghua Shenzhen International Graduate School, Tsinghua University. His current research interests include multimedia and computer vision.

Yong Zhang received the Ph.D. degree in pattern recognition and intelligent systems from the Institute of Automation, Chinese Academy of Sciences in 2018. From 2015 to 2017, he was a Visiting Scholar with the Rensselaer Polytechnic Institute. He is currently with the Tencent AI Lab. His research interests include computer vision and machine learning.

Baoyuan Wu is an Associate Professor of School of Data Science, the Chinese University of Hong Kong, Shenzhen (CUHK-Shenzhen). He is also the director of the Secure Computing Lab of Big Data, Shenzhen Research Institute of Big Data (SBRID). He received the PhD degree from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, on June 2014. From November 2016 to August 2020, he was a Senior and Principal Researcher at Tencent AI lab. His research interests are AI security and privacy, machine learning, computer vision and optimization.

Yan Feng is currently a master student in Tsinghua Shenzhen International Graduate School, Tsinghua University. His current research interests include computer vision and AI security.

Jingyi Zhang is currently a master student in School of Computer Science and Engineering, University of Electronic Science and Technology of China. His current research interests include multimedia and computer vision.



Yanbo Fan is currently a Senior Researcher at Tencent AI Lab. He received his Ph.D. degree from Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing, China, in 2018, and his B.S. degree in Computer Science and Technology from Hunan University in 2013. His research interests are computer vision and machine learning.

Yujiu Yang Yujiu Yang (Member, IEEE) received the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences. He is an Associate Professor with the Tsinghua Shenzhen International Graduate School, Tsinghua University. His research interests include natural language processing and computer vision.

Supplemental Material for Generalizable Black-Box Adversarial Attack with Meta Learning

Fei Yin*, Yong Zhang*, Baoyuan Wu*†, *Member, IEEE,* Yan Feng, Jingyi Zhang, Yanbo Fan, Yujiu Yang†, *Member, IEEE*

APPENDIX A
METHOD ANALYSIS

A.1 Comparison with Methods Directly Using the Surrogate Model

Our framework can combine the learned prior with different types of off-the-shelf query-based black-box attack methods in the meta-test phase and significantly boost their performance in terms of attack efficiency as well ASR.

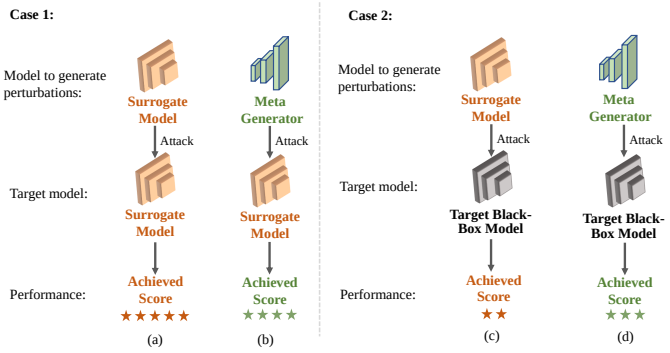


Fig. 1. Two cases for transfer attack.

For a fair comparison with the transfer-based methods directly using the surrogate model, we only use the trained generator for evaluation. Figure 1 presents the comparison between the surrogate model and our meta generator in two cases in the scenario of transfer attack, *i.e.*, no query of the unknown target model. In Case 1, the surrogate model is treated as the target model. Perturbations are generated by the surrogate model and our meta generator, respectively. In this case, using the surrogate model to generate perturbation achieves much better performance than our meta generator (***e.g.,* ASR: PGD-100 100% v.s. Ours 74%. Surrogate model: ResNet-50, Target model: ResNet-50**). Because the unknown target model is the same as the surrogate model. Though our meta model is learned using the gradient from the surrogate, it does not try to learn mapping exactly from a sample to the gradient but learns the sample-dependent perturbation distribution by attacking a large set of samples, *i.e.,* it captures some common properties among samples. Given a sample, our meta generator can provide a perturbation distribution that tells the probability of a sampled perturbation to be effective. It cannot predict the exact perturbation as the surrogate model. Therefore, in Case 1, the surrogate model wins.

In Case 2, the unknown target model is different from the surrogate model. Given a sample, its generated perturbation is totally determined by the sample and the model itself. The perturbation seems to 'overfit' the surrogate model as the gradient exactly comes from the surrogate model. Differently, our meta model generates the perturbation according to the sample and the learned prior (*i.e.*, common properties among samples). Hence, the perturbation is generated by considering not one sample but a set of samples. It always generalizes better than the surrogate model in this case (***e.g.,* ASR: PGD-100 35.5% v.s. Ours 56.8%. Surrogate model: ResNet-50, Target model: ResNet-18**).

A.2 Working Principle of MCG

We provide an illustration of how our method works for better understanding in Figure 2. Figure 2 (a) presents the attack by using the surrogate model via '**PGD**'. The perturbation is generated according to the surrogate model and the clean example itself. Figure 2 (b) shows that our meta generator captures the sample-dependent conditional distribution by performing a large set of attacking tasks involving a number of samples. The perturbation distribution is denoted by the black dash circle. Figure 2 (c) shows that the meta generator is directly used to attack the unknown target model by sampling a perturbation with a high probability. It is a transfer attack. Figure 2 (d) shows the meta update of the meta generator. The meta generator is combined with a query-based attack method. The query feedback is used to update the meta generator. Hence, the perturbation distribution is shifted from the black dash circle to the black dash circle. The black one is better as the update uses the feedback information about the target model. Figure 2 (e) shows the usage of the updated meta generator. When combining with a query-based method that requires a perturbation as initialization, we can sample a perturbation with a high probability as the initialization. When

- *Fei Yin, Yong Zhang and Baoyuan Wu are co-first authors.*
- *Baoyuan Wu (wubaoyuan@cuhk.edu.cn) and Yujiu Yang (yang.yujiu@sz.tsinghua.edu.cn) are corresponding authors.*
- *Fei Yin, Yan Feng and Yujiu Yang are with Tsinghua Shenzhen International Graduate School, Tsinghua University. Baoyuan Wu is with School of Data Science, Shenzhen Research Institute of Big Data, the Chinese University of Hong Kong, Shenzhen. Yong Zhang and Yanbo Fan are with Tencent AI Lab. Jingyi Zhang is with the Center for Future Media and the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China.*

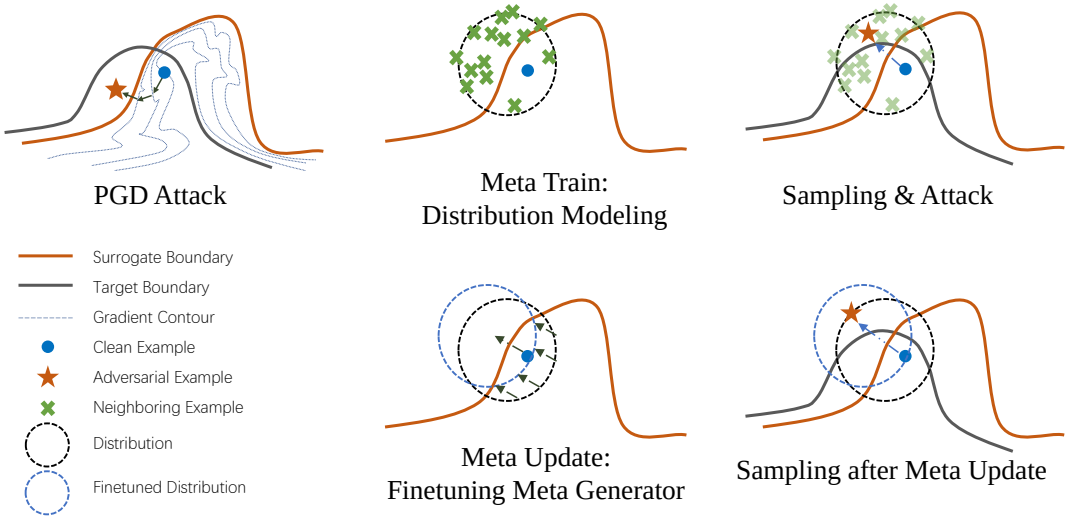


Fig. 2. Illustration of how the proposed method works.

combining with a query-based method that requires a distribution as initialization, then we can use the distribution represented by the generator as the initialization.

According to Figure 2, our method learns the prior knowledge in the meta-train phase and updates the generator in the meta-test phase. It can be combined with off-the-shelf query-based methods. The meta update involves the query feedback that improves the meta generator. Our framework can boost the query-based methods in attack efficiency as well as ASR.

A.3 Advantages of Using c-Glow as Meta Generator

c-Glow can model the exact log-likelihood of the underlying distribution, making it feasible to directly minimize the KL divergence between the approximated and real conditional adversarial distributions (CAD), rather than only optimizing the lower bound as in VAE models or finding an approximate maximum point in CAD as in learning-to-learn methods.

In learning-to-learn methods, CNN and RNN generators are optimized based on the gradients of the classifiers, which means attackers can only pre-train these generators on surrogate classifiers, and then fully transfer their parameters for black-box attacks. As pointed out in [1], such a *fully-transfer mechanism* will introduce the so-called *surrogate bias* (due to differences in architectures and training datasets between surrogate and target models), which inevitably harms black-box attack performance. In contrast, as c-Glow consists of two parts of parameters, $i.e.$, Gaussian and mapping parameters, we can utilize the *partial transfer mechanism* to alleviate the surrogate bias.

A.4 Visualization of Adversarial Perturbations

We present some visualization examples of five attack methods in Fig. 3. The experimental settings are as follows. We perform untargeted black-box attack with ResNet-50 as the surrogate model and ResNet-18 as the target model. Five attack methods are evaluated, including MCG (pure transfer), CG-Attack, MCG + CG-Attack, Square, and MCG + Square. The perturbation limit is set to $\ell_\infty \leq 0.05$. It is interesting to see that our proposed MCG is likely to generate nearly symmetric and rhombus-like patterns (see the top row in Fig. 1). Considering the extremely high transferability of these perturbations, they may provide good instances to analyze the characteristics of highly transferable perturbations. However, we realize that these perturbations will vary across different clean images and different models. It requires more comprehensive evaluations and ingenious analysis tools/approaches to reveal some general characteristics. It will be explored in our future work.

Appendix B
Additional Experiment Results

B.1 Comparison with Combination-based Methods on the CIFAR-10 dataset

In Table 4 of the manuscript, we present the comparison with combination-based methods on the ImageNet dataset. Here, we provide the comparison results on the CIFAR-10 dataset. Competing methods are TREMBA [2] and AdvFlow [3]. The results are shown in Table 1. On CIFAR-10, our method achieves the best performance under all attack cases in all three metrics, which is consistent with the results on ImageNet.

B.2 Additional Experiments in Open-set Attack Scenario

In real-world scenarios, the surrogate model will share less common knowledge with the target model. This situation strongly increases the difficulty of attacking. In Sec.4.3 of the manuscript, we verify the adaptability of the proposed method across datasets via training on CIFAR-10 and testing on CIFAR-100. Here, we provide additional experiments on ImageNet [9] and OpenImage [10].

Experiments on ImageNet. To simulate the open-set scenarios, we first randomly select 10 classes from the $1,000$ classes of ImageNet and split the 10 classes into two groups evenly. We train the surrogate model on the training set of the one group of classes and train the target model on the training set of the other group of classes. The training data of the meta generator is the same as the data used for the surrogate model. The testing data is the validation set corresponding to the training categories of the target model. The results of the untargeted attack on ImageNet are shown in Table 2. The open-set attack on ImageNet is more challenging than that on CIFAR-10 as the median query numbers of several attack methods are relatively high, *e.g.,* NES, SimBA-DCT, and

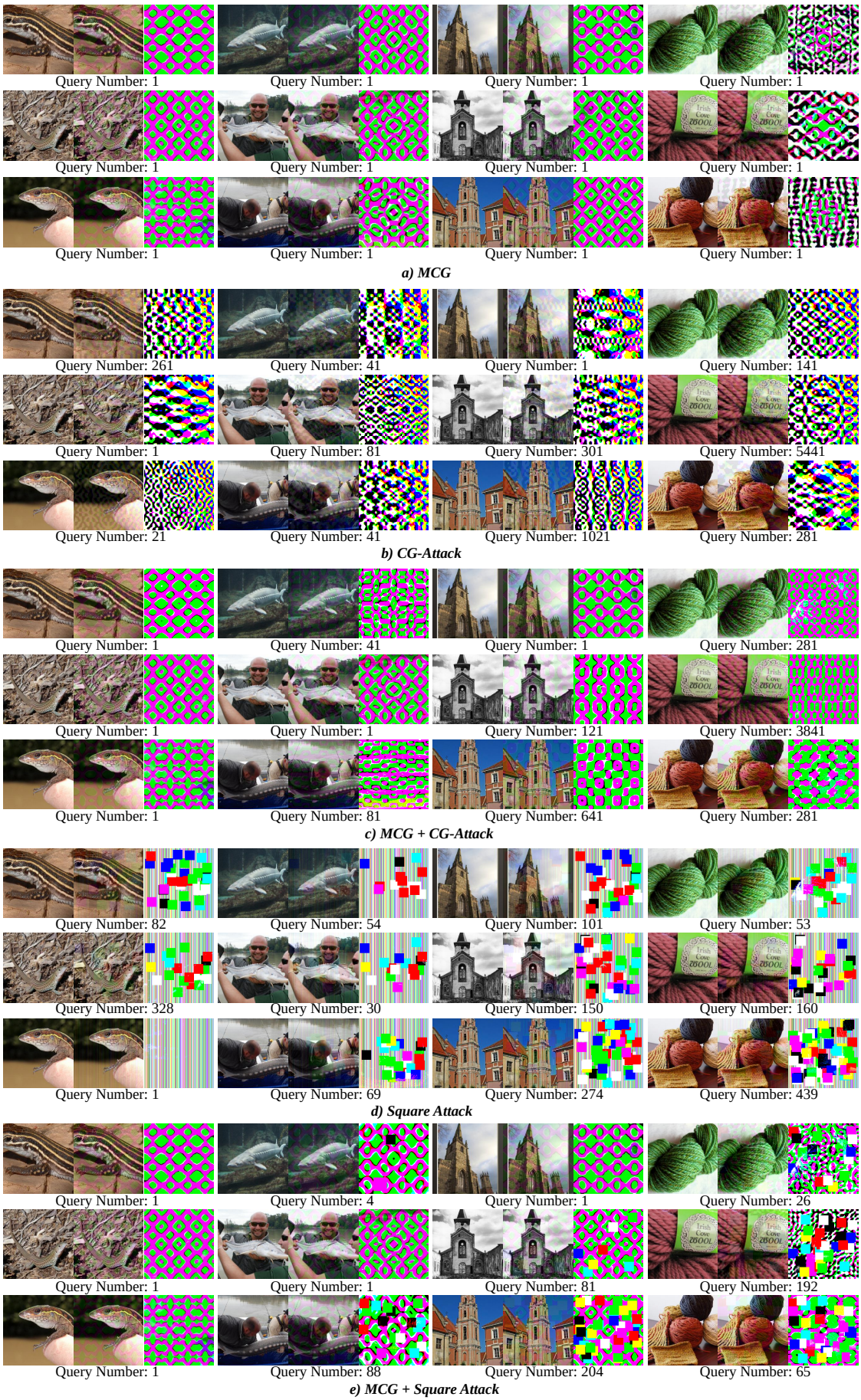


Fig. 3. Visualization of the generated perturbations of the ImageNet dataset. In each triple block, each column represents the benign image, the adversarial example, and the perturbation, respectively. The origin range of the perturbation is $[-0.05, 0.05]$ and we scale it to the range $[0, 255]$ for better visualization.

TABLE 1
Comparison with combination-based methods on the CIFAR-10 dataset

Target model →	ResNet-PreAct-110			DenseNet-121			VGG-19			PyramidNet-110		
Attack Method ↓	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
Untargeted Attack												
TREMBA [2]	90.9%	120.7	64.0	97.8%	126.4	66.0	97.7%	125.5	63.0	97.9%	82.3	39.0
AdvFlow [3]	97.2%	841.4	600.0	**100.0%**	1025.3	740.0	98.2%	1079.1	860.0	99.7%	857.5	560.0
MCG + CG-Attack	**100.0%**	**41.8**	**1.0**	**100.0%**	**21.3**	**1.0**	**100.0%**	**38.8**	**1.0**	**100.0%**	**12.8**	**1.0**
Targeted Attack												
TREMBA [2]	91.2	1125.3	868.0	92.3	1123.4	879.0	96.5	1331.5	1142.0	98.1	1082.4	759.0
AdvFlow [3]	98.6	911.7	820.0	96.3	1021.5	860.0	**97.4**	1144.1	940.0	**100.0**	908.1	820.0
MCG + CG-Attack	**100.0**	**430.8**	**181.0**	**99.8**	**608.5**	**241.0**	**97.4**	**944.1**	**381.0**	**100.0**	**290.6**	**141.0**

TABLE 2
Untargeted attack on the ImageNet dataset in the open-set scenario.

Target Model →	ResNet-18			VGG-16			WRN-50			Inception-V3		
Attack Method ↓	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
NES [4]	66.7%	4684.9	4516.0	68.5%	4903.7	4778.5	74.0%	4199.8	3760.0	94.9%	1989.0	1439.5
MCG + NES	**77.5%**	**4113.1**	**3845.0**	**77.4%**	**4392.6**	**4517.0**	**78.9%**	**3503.1**	**3089.0**	**96.8%**	**1773.5**	**1251.5**
CG-Attack [1]	50.2%	524.8	61.0	36.3%	476.3	**41.0**	55.1%	615.6	141.0	64.4%	**497.3**	**41.0**
MCG + CG-Attack	**57.7%**	**446.2**	**21.0**	**41.2%**	**468.6**	61.0	**55.2%**	**591.3**	**61.0**	**65.0%**	521.4	**41.0**
SimBA-DCT [5]	20.9%	2123.4	2002.0	21.4%	2782.8	3202.0	33.3%	2947.2	3187.0	53.3%	2485.2	2283.0
MCG + SimBA-DCT	**42.9%**	**1182.9**	**829.0**	**35.2%**	**1006.6**	**669.0**	**52.4%**	**1293.4**	**1335.0**	**60.0%**	**1402.7**	**1343.0**
Signhunter [6]	**100.0%**	**142.2**	**60.0**	98.5%	391.6	145.0	**100.0%**	157.7	49.0	**100.0%**	129.5	**45.0**
MCG + Signhunter	**100.0%**	161.0	64.0	**99.2%**	**364.6**	**137.5**	**100.0%**	**153.0**	**44.0**	**100.0%**	**121.9**	46.0
Square [7]	**100.0%**	288.1	196.0	**100.0%**	632.6	336.0	**100.0%**	272.5	156.0	**100.0%**	207.1	126.0
MCG + Square	**100.0%**	**220.1**	**119.0**	**100.0%**	**572.3**	**271.0**	**100.0%**	**220.3**	**112.0**	**100.0%**	**175.8**	**97.0**
MetaAttack [8]	78.8%	5646.7	5951.0	75.7%	5485.2	5293.0	76.2%	5346.7	5505.5	93.6%	4168.6	4198.5
MCG + MetaAttack	**79.2%**	**4360.4**	**4416.0**	**76.6%**	**5078.9**	**5619.0**	**82.4%**	**4555.6**	**4412.0**	**94.5%**	**3896.9**	**3974.0**

TABLE 3
Untargeted attack evaluation on the OpenImage dataset.

Target Model →	ResNet-18			VGG-16			WRN-50			Inception-V3		
Attack Method ↓	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
NES [4]	28.3%	4915.9	5041.0	40.4%	3894.7	3718.0	61.1%	4035.6	3823.0	86.9%	2671.2	1954.0
MCG + NES	**52.6%**	**1306.7**	**281.0**	**69.9%**	**764.3**	**1.0**	**74.0%**	**1327.8**	**1.0**	**92.9%**	**1712.8**	**23.0**
CG-Attack [1]	62.3%	2033.8	21.0	66.7%	2044.4	21.0	66.9%	1948.1	**81.0**	61.5%	2273.4	41.0
MCG + CG-Attack	**72.7%**	**1337.4**	**1.0**	**82.5%**	**1419.3**	**1.0**	**78.7%**	**1201.2**	101.0	**75.5%**	**1039.1**	**1.0**
SimBA-DCT [5]	90.4%	374.201	17	**94.9%**	254.7	10.0	**92.5%**	301.4	12.0	83.2%	320.180	11.0
MCG + SimBA-DCT	**90.8%**	**348.5**	**7.0**	94.7%	**226.2**	**1.0**	**92.5%**	**250.7**	**1.0**	**83.9%**	**249.6**	**1.0**
Signhunter [6]	**99.7%**	331.8	11.0	**100.0%**	149.2	7.0	**100.0%**	233.3	13.0	**100.0%**	259.9	8.0
MCG + Signhunter	98.6%	**267.9**	**6.0**	**100.0%**	**100.9**	**1.0**	**100.0%**	**159.9**	**1.0**	**100.0%**	**247.7**	**4.0**
Square [7]	**99.9%**	292.1	69.0	**100.0%**	247.0	68.5	**100.0%**	156.9	61.5	**100.0%**	178.9	56.0
MCG + Square	99.7%	**230.7**	**24.0**	**100.0%**	**180.9**	**1.0**	**100.0%**	**93.9**	**1.0**	**100.0%**	**123.9**	**8.5**
MetaAttack [8]	69.7%	4366.9	3964.0	79.9%	4029.4	3635.5	72.2%	4364.2	3754.0	81.4%	3689.8	3093.0
MCG + MetaAttack	**75.1%**	**3033.8**	**1670.0**	**84.7%**	**1637.9**	**1.0**	**76.9%**	**1722.0**	**1.0**	**86.9%**	**2351.1**	**41.0**

MetaAttack. In this setting, our method can still improve their performance, especially the ASR for NES and SimBA-DCT.

Experiments of training on ImageNet and testing on OpenImage. To further verify the adaptability across different datasets, we perform another untargeted attack experiment by training the meta generator on the ImageNet dataset and testing it on the OpenImage dataset. Target models (*i.e.*, ResNet-18, VGG-16, WRN-50, and Inception-V3) are trained with the training data of 10 randomly selected classes of OpenImage. The meta generator is trained with the training data of ImageNet guided by the surrogate model. The results are shown in Table 3. Our MCG can reduce the query cost of attacks and improve the ASR in most cases, which demonstrates that the meta generator can be fast-adapted to different target models across different datasets.

B.3 Comparison with CNN and RNN-based generators.

CNN and RNN-based generators can also be fine-tuned to mitigate the bias in our framework. In our framework the generator is used to capture the prior distribution of adversarial examples, which is a replaceable component. Other types of generators can be flexibly incorporated into the framework to replace the c-Glow. To compare the influence of different generators, we perform experiments by integrating the CNN or RNN-based generator into our framework to replace c-Glow.

Implementation details of the CNN-based generator. We follow [11] and [12] to re-implement the CNN-based generator. The backbone includes a feature extractor f, a generator network G, and a discriminator network D. We concatenate the feature

TABLE 4
Untargeted Attack comparison with CNN and RNN-based generators on the CIFAR-10 dataset.

Target model →	ResNet-PreAct-110			DenseNet-121			VGG-19			PyramidNet-110		
Attack Method ↓	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
Square	**100.0%**	227.3	144.5	**100.0%**	260.3	159.0	**100.0%**	342.0	175.5	**100.0%**	165.5	100.5
CNN MCG + Square	**100.0%**	67.4	**1.0**	**100.0%**	58.7	**1.0**	**100.0%**	120.8	24.0	**100.0%**	39.3	**1.0**
RNN MCG + Square	**100.0%**	49.9	**1.0**	**100.0%**	69.4	6.0	**100.0%**	97.9	20.0	**100.0%**	33.1	**1.0**
c-Glow MCG + Square	**100.0%**	**39.7**	**1.0**	**100.0%**	**47.6**	**1.0**	**100.0%**	**57.9**	**1.0**	**100.0%**	**29.6**	**1.0**

TABLE 5
Untargeted Attack comparison with MAML-based meta-learning strategy on the CIFAR-10 dataset.

Target model →	ResNet-PreAct-110			DenseNet-121			VGG-19			PyramidNet-110		
Attack Method ↓	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median	ASR	Mean	Median
Square [7]	**100.0%**	227.3	144.5	**100.0%**	260.3	159.0	**100.0%**	342.0	175.5	**100.0%**	165.5	100.5
MCG-MAML + Square	**100.0%**	42.2	**1.0**	**100.0%**	**45.7**	**1.0**	**100.0%**	107.1	18.0	**100.0%**	**27.2**	**1.0**
MCG-REPTILE + Square	**100.0%**	**39.7**	**1.0**	**100.0%**	47.6	**1.0**	**100.0%**	**57.9**	**1.0**	**100.0%**	29.6	**1.0**

TABLE 6
Validation of the extension with SGM. Untargeted attack evaluation on the ImageNet dataset.

Target model →	ResNet-18			VGG-16			WRN-50			Inception-V3		
Attack Method ↓	FASR	Mean	Median	FASR	Mean	Median	FASR	Mean	Median	FASR	Mean	Median
MCG + Square	60.1%	31.7	**1.0**	71.3%	24.8	**1.0**	51.3%	59.9	**1.0**	30.0%	123.8	24.0
SGM + MCG + Square	**67.9%**	**22.9**	**1.0**	**76.2%**	**22.6**	**1.0**	**65.5%**	**41.0**	**1.0**	**43.9%**	**88.2**	**6.0**

$f(x)$ of image x and a noise vector sampled from learnable mean parameters z. Then we feed the concatenation to the generator G. The generator G predicts an adversary perturbation x_{adv} corresponding to x. The discriminator D distinguishes the output distribution of the generator with the real distribution. The adversarial loss of the surrogate model (Eq. 4 of the manuscript) is also used. To bound the magnitude of perturbation, we minimize L_{inf} bound norm of adversary perturbation. The loss function is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{GAN}} + \alpha\mathcal{L}_{adv} + \beta\mathcal{L}_{inf}, \tag{1}$$

where

$$\mathcal{L}_{GAN} = \mathbb{E}_x[\log \mathcal{D}(x) + \mathbb{E}_x \log(1 - \mathcal{D}(x + G(z, f(x)))], \tag{2}$$

$$\mathcal{L}_{adv} = \mathbb{E}_x[g_w(x + G(z, f(x)), t)], \tag{3}$$

$$\mathcal{L}_{inf} = \mathbb{E}_x \|G(z, f(x))\|_\infty. \tag{4}$$

t is the target class and g_w denotes the surrogate classifier.

Implementation details of the RNN-based generator. We follow [13] to re-implement the RNN-based model. We flatten the input benign images into one-dimension feature vectors and directly feed the features into the RNN network G. The initial hidden state h for the input sequence is sampled from the learnable mean parameters z. We reshape the output sequence from G back to the corresponding spatial dimension and achieve the adversarial perturbation. Similarly, the adversarial loss $\mathcal{L}_{adv}$ and the bound loss $\mathcal{L}_{inf}$ are used to optimize the generator. The loss function is defined as:

$$\mathcal{L} = \mathcal{L}_{adv} + \lambda\mathcal{L}_{inf}. \tag{5}$$

For the rest training and testing strategy, we keep the settings the same as in our manuscript.

Experimental results. The experimental results are shown in Tab. 4. We use Square [7] as the baseline method. It can be observed that both CNN-based and RNN-based generators can improve the performance of the baseline method considerably. The results demonstrate the scalablility of our framework, *i.e.,* the c-Glow generator can be replaced by other types of generators. Comparing the three types of generator, Flow-based MCG achieves better performance than the other two. Moreover, overall the RNN-based generator performs slightly better than the CNN-based generator.

B.4 Comparison with MAML-based Meta-learning Methods.

Both MAML and REPTILE are powerful meta-learning algorithms aiming at optimizing for an initial representation that can be effectively fine-tuned. MAML unrolls and differentiates through the computation graph of the gradient descent algorithm, while Reptile simply performs stochastic gradient descent on each task, which makes Reptile take less computation and memory than MAML. We perform an experiment to compare REPTILE-based MCG with MAML-based MCG in the untargeted attack scenario on the CIFAR-10 dataset. The baseline method is Square [7]. Results are shown in Tab. 5. It can be observed that 'MCG-MAML + Square' and 'MCG-REPTILE + Square' achieve close performance. This comparison also demonstrates the flexibility of using different meta-learning algorithms in the proposed framework.

B.5 Extended Experiments with Skip Gradient Method

Skip Gradient Method (SGM) [14] is a transfer-based attack method that excavates the internal gradient flow of skip-connection branches to generate more transferable perturbation. Since we utilize the model-level adversarial transferability through the surrogate model, we can introduce the strategy of SGM into our framework to boost the attack performance. Similar to SGM, we change the backward gradient weights of skip-connection branches of our surrogate model to train the generator. During the attacking process, we apply the same strategy to the surrogate model to fine-tune our meta generator. We perform an experiment on ImageNet in the untargeted attack scenario with ResNet-50 as the surrogate model. The results are shown in Table 6. 'MCG + Square' is our original method. 'SGM + MCG + Square' means that we additional introduce the strategy of SGM. The results show the SGM strategy

helps our model achieve improvements in both FASR and the query number.

REFERENCES

[1] Y. Feng, B. Wu, Y. Fan, L. Liu, Z. Li, and S. Xia, "Boosting black-box attack with partially transferred conditional adversarial distribution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 15 095–15 104.

[2] Z. Huang and T. Zhang, "Black-box adversarial attack with transferable model-based embedding," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[3] H. Mohaghegh Dolatabadi, S. Erfani, and C. Leckie, "Advflow: Inconspicuous black-box adversarial attacks using normalizing flows," in *Proc. Adv. Neural Inform. Process. Syst.*, 2020, pp. 15 871–15 884.

[4] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2137–2146.

[5] C. Guo, J. R. Gardner, Y. You, A. G. Wilson, and K. Q. Weinberger, "Simple black-box adversarial attacks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2484–2493.

[6] A. Al-Dujaili and U. O'Reilly, "Sign bits are all you need for black-box attacks," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[7] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: a query-efficient black-box adversarial attack via random search," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 484–501.

[8] J. Du, H. Zhang, J. T. Zhou, Y. Yang, and J. Feng, "Query-efficient meta attack to deep neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[9] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, 2015.

[10] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov *et al.*, "The open images dataset v4," *Int. J. Comput. Vis.*, pp. 1956–1981, 2020.

[11] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," in *Proc. Int. Joint Conf. on Artif. Intell.*, 2018, pp. 3905–3911.

[12] S. Jandial, P. Mangla, S. Varshney, and V. Balasubramanian, "Advgan++: Harnessing latent layers for adversary generation," in *Proc. Int. Conf. Comput. Vis. Worksh.*, 2019, pp. 2045–2048.

[13] Y. Xiong and C.-J. Hsieh, "Improved adversarial training via learned optimizer," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 85–100.

[14] D. Wu, Y. Wang, S.-T. Xia, J. Bailey, and X. Ma, "Skip connections matter: On the transferability of adversarial examples generated with resnets," in *Proc. Int. Conf. Learn. Represent.*, 2020.