Pattern Recognition 133 (2023) 109037

Contents lists available at ScienceDirect

Pattern Recognition

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

Query efficient black-box adversarial attack on deep neural networks

Yang Bai [a,b,1], Yisen Wang [c,d,1,*], Yuyuan Zeng [b], Yong Jiang [b,e], Shu-Tao Xia [b,e,*]

[a] Tencent Security Zhuque Lab, China
[b] Tsinghua University, China
[c] Key Laboratory of Machine Perception (MoE), School of Intelligence Science and Technology, Peking University, China
[d] Institute for Artificial Intelligence, Peking University, China
[e] Peng Cheng Laboratory, China

ARTICLE INFO

Article history:
Received 13 December 2021
Revised 23 August 2022
Accepted 6 September 2022
Available online 11 September 2022

Keywords:
Black-box adversarial attack
Adversarial distribution
Query efficiency
Neural process

ABSTRACT

Deep neural networks (DNNs) have demonstrated excellent performance on various tasks, yet they are under the risk of adversarial examples that can be easily generated when the target model is accessible to an attacker (white-box setting). As plenty of machine learning models have been deployed via online services that only provide query outputs from inaccessible models (*e.g.*, Google Cloud Vision API2), black-box adversarial attacks raise critical security concerns in practice rather than white-box ones. However, existing query-based black-box adversarial attacks often require excessive model queries to maintain a high attack success rate. Therefore, in order to improve query efficiency, we explore the distribution of adversarial examples around benign inputs with the help of image structure information characterized by a Neural Process, and propose a Neural Process based black-box adversarial attack (NP-Attack) in this paper. Our proposed NP-Attack could be further boosted when applied with surrogate models or tiling tricks. Extensive experiments show that NP-Attack could greatly decrease the query counts under the black-box setting.

1. Introduction

Deep neural networks (DNNs) have been widely deployed in various practical tasks ranging from computer vision (CV) [1,2], natural language processing (NLP) [3] to autonomous driving [4], etc. Thus the security of deep learning shows its salient necessity. Recently the existence of adversarial examples [5] has highlighted some potential threats. Adversarial examples reveal the vulnerability of DNNs to well-designed adversarial perturbations, which could mislead DNNs with high confidence. In general, they can be crafted with or without accessing the target models and thus be categorized into white-box and black-box attack methods respectively. White-box adversarial attack methods [5,6] have access to the true gradient and generate perturbations by backward propagating on the target model. In contrast, another series of attack methods have no access to the parameters as well as the gradients of the target model, dubbed black-box adversarial attack methods [7,8]. Existing black-box attack methods could be divided into two major series according to their research methodologies. The transfer-based attack methods [9,10] generate adversarial examples on the (white-box) surrogate model, while the query-based attack methods [11–13] optimize adversarial perturbations on the (black-box) target model directly with the returned query results. Considering the wide deployment of online models in practical scenarios, *e.g.*, Google Cloud V2 [2], query-based attack methods highlight the essence of adversarial robustness.

To be specific, those query-based black-box adversarial attack methods return query results in the way of hard labels or scores. Therefore, the query-based black-box attack methods could be further divided into the decision-based ones [14–16] (*i.e.*, returning one-hot hard labels) and the score-based ones [11,13,17] (*i.e.*, returning classification scores). How to efficiently use the returned information of black-box target models is the main challenge to improve query efficiency, especially for score-based black-box attack methods. Existing methods [11,12] use the zeroth order optimization by adding perturbations in each query independently. Developed from adding those independent perturbations, some use the evolution strategy to optimize the distribution of adversarial perturbations [13,17]. They formulate the adversarial perturbations

* Corresponding authors at: Key Laboratory of Machine Perception (MoE), School of Intelligence Science and Technology, Peking University, China (Y. Wang) and Tsinghua University, China (S.-T. Xia).
E-mail addresses: yisen.wang@pku.edu.cn (Y. Wang), xiast@sz.tsinghua.edu.cn (S.-T. Xia).

[1] Equal Contribution. For Yang Bai, the work was done when she was a Ph.D. student at Tsinghua University and an intern at Tencent Security Zhuque Lab.

[2] https://cloud.google.com/vision/

https://doi.org/10.1016/j.patcog.2022.109037
0031-3203/

(added in each query) as samplings from the same distribution which can be optimized. However, such optimization over distribution is on pixel-level, which is time-consuming under the high dimension of images.

As adversarial examples share similar visual expressions with benign images, optimizing in the compressed latent space with some encoder-decoder model is possible. However, this kind of method comes up with some new challenges, for example, adversarial perturbations are instance-wise while normal encoder-decoder models usually characterize the distributions. Therefore, normal encoder-decoder models cannot be adopted while we find one specific encoder-decoder model that is well suitable in the black-box adversarial attack setting, *i.e.*, Neural Process (NP) [18], and propose the NP-Attack to generally improve the query efficiency, especially for those evolution strategy methods. Our contributions could be summarized as follows.

- We propose a distribution-based black-box attack, *i.e.*, Neural Process based black-box attack (NP-Attack), which uses the image structure information to model adversarial distributions and reduce the required query counts.
- The proposed NP-Attack has several optimization variants due to the variables in NP. The optimization on the deterministic variable focuses more on the local information, while the optimization on the latent variable focuses more on the global information. Different optimization variants have different effects on the location of adversarial perturbations, which brings more flexibility for NP-Attack.
- On both untargeted and targeted attacks, NP-Attack greatly reduces the needed query counts under the same attack success rate and distortion on the benchmark datasets.

A preliminary version of this work has been presented as a conference version [19]. In the current work, we incorporate the following additional contents:

- We introduce surrogate (white-box classification) models in NP-Attack to provide the best initial point for NP latent variables, and further improve query efficiency while maintaining satisfactory attack success rates.
- We explore deploying differently pre-trained NP models, which are pre-trained either with partial training data or external training data, and further discover the efficiency of our proposed NP-Attack.
- We add some tiling tricks on NP-Attack to improve query efficiency, which are well suited for high-dimensional data or attacking robust target models. Moreover, we also conduct comprehensive ablation study experiments on tiling parameters.
- We also evaluate NP-Attack on adversarially trained models to further discuss the capability of our method. Extensive and comprehensive experiments on benchmark demonstrate that our NP-Attack still outperforms existing evolution strategy methods in these black-box attack tasks.

2. Related work

As mentioned above, query-based black-box adversarial attacks could be further divided into decision- or score-based ones, depending on the black-box models that return (hard) labels or (soft) confidence scores. Decision-based attack methods generate adversarial examples by initializing with images sampled from the target class, and optimizing to minimize their pixel-level distances to the ground truth images. Score-based ones generate adversarial examples by initializing with images sampled from the ground truth class, and optimizing to minimize their returned score results to the target class. Since our proposed NP-Attack leverages the (soft) confidence scores returned by black-box models, we mainly present related work on score-based adversarial attacks. More specifically, they can be further divided into zeroth order optimization series, evolution strategy series and random/local search series, etc.

Zeroth Order Optimization Zeroth order optimization-based adversarial attack methods, which are also called derivative-free methods, generate adversarial perturbations by estimating the gradient in the way of adding independent noises. *e.g.*, ZOO [11] optimizes the target perturbations on pixel level by adding noises which are independently sampled from Gaussian distributions. Developed from ZOO, AutoZOOM [12] accelerates such optimization with VAE [20] by adding the perturbations in the latent space.

Evolutionary Strategy Considering the interaction between independent perturbations, evolutionary strategy (ES) methods formulate adversarial attacks as optimizing the distribution of adversarial perturbations instead of adding individual noises. Therefore, the perturbation of the next sampling is often more effectively evaluated by adversarial performance, which can improve query efficiency. This optimization is handled by re-parameterizing skills. Typical methods are QL [21], Bandits [22] and $\mathcal{N}$Attack [13], etc. QL samples perturbations from a distribution on a benign image. Bandits work by adding some priors related to time or data. $\mathcal{N}$Attack optimizes the adversarial perturbations by updating their mean and variance on benign images. Some other methods also update the distribution of adversarial perturbations on specific surrogate (white-box) models following evolutionary strategy formulations, such as P-RGF [23], TREMBA [24], etc. P-RGF provides a mathematical formula that balances the prior gradient of (white-box) surrogate model and the estimated gradient of (black-box) target model. TREMBA explores priors from another angle, by introducing some adversarial priors from the surrogate model with a pre-trained VAE model. In this paper, our proposed NP-Attack also belongs to the ES family of methods, so we mainly compare NP-Attack with such methods as baselines.

Random/Local Search Random/Local search method is developed based on the knowledge of FGSM [5] in white-box adversarial attacks. FGSM generates adversarial examples which are distributed on the surface of a given ϵ-ball centered on a benign example, implying that perturbing the pixel-level gradient signs is sufficient to generate strong adversarial examples. Typical methods are SignHunter [25], Parsimonious [26], DFO [27] and Square Attack [28], etc. Such methods flip the gradient sign of perturbations on the pixel level, and then add perturbations of the same magnitude under L_∞ norm. Thus the resulting (black-box) adversarial examples are all distributed on the ϵ-ball around the benign example. To be more specific, SignHunter proposes a divide-and-conquer algorithm to estimate the gradient sign bits. Parsimonious explores the gradient sign bits' distribution by slicing images into chunks. DFO improves Parsimonious by deploying some tiling and evolution strategy skills. Square Attack adds perturbations with the same gradient sign in each block, showing excellent performance.

3. Proposed method

3.1. Preliminary

Neural Process (NP) [18] is a combination of the best from neural networks and Gaussian Process [29,30], which could be regarded as the encoder-decoder model estimating the pixel value of the target position. Its attentive version, Attentive Neural Process (ANP) [31,32], is applied in our NP-Attack[3]. It is worth noting that the NP model works differently from other encoder-decoder models, that is, it optimizes the distribution in the latent space

[3] We still use NP in the following without ambiguity

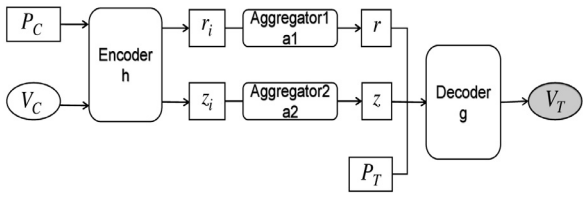


Fig. 1. The structure of NP model trained on pixels of one image. Here, V_C and V_T are the context and target pixel values, then P_C and P_T are the corresponding pixel positions. z_i and r_i are pixel-wise latent variables and deterministic variables.

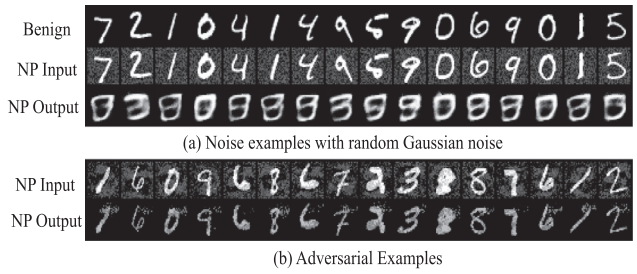


Fig. 2. The reconstruction of pre-trained NP. Given noise examples with random Gaussian noises (a) and adversarial examples generated by PGD attack (b) under $L_\infty = 0.2$ as the input of the benign pre-trained NP model, the adversarial examples could be reconstructed while noised examples could not.

instance-wise. Though the parameters in NP are shared on the whole data set, the latent variables vary among instances. Once trained, NP shares the same encoder and decoder on the same dataset (regardless of different selected images), yet varying on latent variables corresponding to different images.

As shown in Fig. 1, the structure of NP consists of the following three parts: encoder h, aggregator a, and decoder g. Overall, NP models the distributions as:

$$p\left(V_T \middle| P_C, P_T, V_C\right) = \int p(V_T|P_T, \boldsymbol{r}, \boldsymbol{z}) q(\boldsymbol{z}|P_C, V_C) d\boldsymbol{z}, \tag{1}$$

where (P, V) represent the pixel position and the pixel value, (C, T) are the given context and target pixel (to be predicted) subsets. Specifically, during the pre-training NP process in our proposed NP-Attack, whole pixels from the given image are used as both context pixel subsets and target pixel subsets. The optimization is to maximize the following Evidence Lower Bound (ELBO)[4]:

$$\begin{aligned} \log p(V_T|P_T, P_C, V_C) \geq & \, \mathrm{E}_{q(\boldsymbol{z}|P_T, V_T)} \left[\log p(V_T|P_T, \boldsymbol{r}, \boldsymbol{z})\right] \\ & - D_{\mathrm{KL}}\left(q(\boldsymbol{z}|P_T, V_T) \| q(\boldsymbol{z}|P_C, V_C)\right). \end{aligned} \tag{2}$$

where (P_C, V_C) and (P_T, V_T) are both the whole pixel position and value pairs in one image.

The pre-trained NP models a distribution by variables $\boldsymbol{z}$ and $\boldsymbol{r}$, where $\boldsymbol{z}$ is sampled from a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, thus the latent variable and deterministic variable are in fact (μ, σ) and $\boldsymbol{r}$. Furthermore, latent variable $\boldsymbol{z}$ accounts for uncertainty in the predictions of V_T for observed (P_C, V_C), while $\boldsymbol{r}$ is calculated by a deterministic function which aggregates (P_C, V_C) into a finite dimensional representation with permutation invariance in C.

Note that such pre-training operation is offline and independent from the main online attack, which means the encoder and decoder are fixed after pre-training and could keep some structured information of images. Then we utilize the pre-trained NP and find that adversarial examples could be reconstructed successfully by optimizing the above mentioned variables for a new distribution and sampling from this optimized distribution.

[4] The implementation details of ANP are shown in Tables 1, 2 and 3

3.2. Pipeline of the proposed NP-attack

Equipped with the pre-trained NP, we propose NP-Attack, which is one kind of evolution strategy black-box attack methods. The key idea of the evolution strategy attack is to use the re-parameterization skills to model the adversarial distribution around the small region of one benign example $\boldsymbol{x}$, such that a sample drawn from this distribution is likely an adversarial example. The basic setting of our NP-Attack is score-based, that is, the instance-wise loss function l over one specific image $\boldsymbol{x}$ is defined as:

$$l(\boldsymbol{x}) := \begin{cases} \max(0, \max_{c \neq y} \log F(\boldsymbol{x})_c - \log F(\boldsymbol{x})_y), & \text{targeted}, \\ \max(0, \log F(\boldsymbol{x})_y - \max_{c \neq y} \log F(\boldsymbol{x})_c), & \text{untargeted}. \end{cases} \tag{3}$$

$F(\boldsymbol{x})$ denotes the softmax outputs, y is the true label in untargeted attack or the target label in targeted attack, and c is other labels except y.

The proposed NP-Attack improves query efficiency, not only due to it optimizes the adversarial distribution in the way of evolution strategy, but also benefits from utilizing the structure information of images. To be specific, the optimization of adversarial distribution is defined through the latent variables of pre-trained NP, which means the adversarial examples are generated in S, the intersection of a latent region modeled by NP and a L_p-ball centered around the benign example $\boldsymbol{x}$, *i.e.*, $S = S_p(\boldsymbol{x}) \bigcap \mathrm{NP}(\boldsymbol{x})$. The objective function L in NP-Attack with regard to adversarial distribution in our optimization criterion is:

$$L((\mu, \sigma), \boldsymbol{r}|\boldsymbol{x}) := \int_{\boldsymbol{x}-\epsilon}^{\boldsymbol{x}+\epsilon} l(\boldsymbol{x}_{rec}) g(\boldsymbol{x}_{rec}|(\mu, \sigma), \boldsymbol{r}) h((\mu, \sigma), \boldsymbol{r}|\boldsymbol{x}) d\boldsymbol{x}_{rec} \tag{4}$$

where g and h are the decoder and encoder of pre-trained NP, $\mu, \sigma, \boldsymbol{r}$ denote for the variables to be optimized in NP-Attack, $\boldsymbol{x}_{rec}$ denotes the image reconstructed by NP, and ϵ denotes the L_p-ball restriction.

In summary, the procedures of modeling and sampling from distribution in NP-Attack are as follows: (1) feed a benign example $\boldsymbol{x}$, and compute $\boldsymbol{r}$ and (μ, σ) from the encoder h of NP; (2) sample $\boldsymbol{z}$ from $\mathcal{N}(\mu, \sigma^2)$; and (3) reconstruct $\boldsymbol{x}_{rec}$ with the optimized $(\boldsymbol{r}, \boldsymbol{z})$ from the decoder g of NP (shown in the following Section 3.3), then project $\boldsymbol{x}_{rec}$ back into the L_p-ball centered at $\boldsymbol{x}$.

3.3. Optimization variants

As the reconstructed image $\boldsymbol{x}_{rec}$ is mainly dependent on the sampled $\boldsymbol{r}$, $\boldsymbol{z}$ and pre-trained decoder g of NP, there are three optimization options: NP-Attack-R/Z/RZ corresponding to the optimized variables $\boldsymbol{r}$, $\boldsymbol{z}$ or both.

NP-Attack-R In this case, $\boldsymbol{r}$ is a variable in NP, cooperating with $\boldsymbol{z}$ to reconstruct an image. We define the loss L on some search distribution $\Pi(\boldsymbol{R}|\boldsymbol{r})$. Omitting $\boldsymbol{z}$ while inheriting decoder g from the pre-trained NP, the estimated loss function in our NP-Attack could be considered as: $L = \mathbb{E}_{\Pi(\boldsymbol{R}|\boldsymbol{r})} l(g(\boldsymbol{R}))$. To simplify the optimization, we give $\boldsymbol{R} \sim \mathcal{N}(\boldsymbol{r}, \boldsymbol{\sigma}'^2)$, where $\boldsymbol{\sigma}'$ is the hyper-parameter. Then the optimization of $\boldsymbol{r}$ could be implemented by adding some random Gaussian noises and using NES. The optimization function could be computed as:

$$\boldsymbol{r}_{t+1} \leftarrow \boldsymbol{r}_t - \frac{\eta}{b} \sum_{i=1}^{b} l(g(\boldsymbol{R}_i)) \nabla_{\boldsymbol{r}_t} \log \mathcal{N}(\boldsymbol{R}_i|\boldsymbol{r}_t, \boldsymbol{\sigma}'^2), \tag{5}$$

where b is the batch size. Given that $\boldsymbol{R}_i = \boldsymbol{r}_t + \boldsymbol{p}_i \boldsymbol{\sigma}'$, where $\boldsymbol{p}_i$ is sampled from the standard Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$ and $\boldsymbol{\sigma}'$ does not essentially equal to $\boldsymbol{\sigma}$ in $\boldsymbol{z}$, we have

$\nabla_{\boldsymbol{r}_t} \log \mathcal{N}(\boldsymbol{R}_i|\boldsymbol{r}_t, \boldsymbol{\sigma}'^2) \propto \frac{\boldsymbol{p}_i}{\boldsymbol{\sigma}'}$. *NP-Attack-Z* In this case, we fix $\boldsymbol{r}$ and optimize the distribution of $\boldsymbol{z}$, in order to sample $\boldsymbol{z}$ with adversary. Under the guidance of score outputs in each query, to be simplified, we just optimize the predictive mean $\boldsymbol{\mu}$, keeping $\boldsymbol{\sigma}$ fixed. The optimization of our NP-Attack is inherited from NES, which could enhance query efficiency compared with those vector-wise gradient estimation strategies. Omitting the fixed $\boldsymbol{r}$ while inheriting decoder g from the pre-trained NP, the estimated loss function in our NP-Attack could be considered as $L = \mathbb{E}_{\mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu},\boldsymbol{\sigma}^2)} l(g(\boldsymbol{z}))$. The optimization of $\boldsymbol{\mu}$ is implemented by adding some random Gaussian noises and further using NES to estimate the gradient. Those added Gaussian noises are of the same variance with $\boldsymbol{z}$ in pre-trained NP. The reason comes as that the same variance could keep $\boldsymbol{z}$ of a simpler representation for the variance after adjustment in each iteration, which benefits the optimization later. The optimization function could be computed as: $\boldsymbol{\mu}_{t+1} \leftarrow \boldsymbol{\mu}_t - \eta \nabla_{\boldsymbol{\mu}} L|_{\boldsymbol{\mu}_t}$, where η denotes a learning rate. To guarantee such optimization being more accurate, in practice the optimization is over a mini-batch of sample size b, which means we add b random perturbations independently during each iteration and operate on those outputs. So the updating operation is on the average values instead:

$$\boldsymbol{\mu}_{t+1} \leftarrow \boldsymbol{\mu}_t - \frac{\eta}{b} \sum_{i=1}^{b} l(g(\boldsymbol{z}_i)) \nabla_{\boldsymbol{\mu}_t} \log \mathcal{N}(\boldsymbol{z}_i|\boldsymbol{\mu}_i, \boldsymbol{\sigma}^2). \tag{6}$$

Given that $\boldsymbol{\mu}_i = \boldsymbol{\mu}_t + \boldsymbol{p}_i\boldsymbol{\sigma}$, $\boldsymbol{z}_i = \boldsymbol{\mu}_i + \boldsymbol{p}_i\boldsymbol{\sigma} = \boldsymbol{\mu}_t + 2\boldsymbol{p}_i\boldsymbol{\sigma}$, where $\boldsymbol{p}_i$ is sampled from the standard Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$, and $\boldsymbol{\sigma}$ is multiplied to keep a simple representation for variance, then we have $\nabla_{\boldsymbol{\mu}_t} \log \mathcal{N}(\boldsymbol{z}_i|\boldsymbol{\mu}_i, \boldsymbol{\sigma}^2) \propto \frac{\boldsymbol{p}_i}{\boldsymbol{\sigma}}$. *NP-Attack-RZ* Similarly, we propose NP-Attack-RZ as a third choice by optimizing both $\boldsymbol{r}$ and $\boldsymbol{z}$ simultaneously.

In summary, the operations in three variants are shown in Algorithm 1. To be specific, NP-Attack-R refers to only $\boldsymbol{r}$ is adjustable, NP-Attack-Z refers to only $\boldsymbol{z}$ is adjustable, and NP-Attack-RZ refers to both $\boldsymbol{r}$ and $\boldsymbol{z}$ are adjustable following the given optimization.

Algorithm 1 NP-Attack

Input: benign image $\boldsymbol{x}$, label y, target neural network F, pre-trained NP model (encoder h, decoder g, aggregator a), maximal optimization iteration T, sample size b, projecting function P, learning rate η

Compute the variables from encoder h on image $\boldsymbol{x}$: $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2), \boldsymbol{r} \leftarrow h(\boldsymbol{x})$
for $t = 0$ **to** $T - 1$ **do**
  if $F(\boldsymbol{x}) \neq y$ (untargeted) or $F(\boldsymbol{x}) = y$ (targeted) **then**
    attack success.
  else
    Sample perturbations from Gaussian distribution: $\boldsymbol{p}_i \sim \mathcal{N}(0, \boldsymbol{I}), i = 1, 2, ..., b$.
    Add the perturbation $\boldsymbol{p}_i$ in NP-Attack-R/Z/RZ, specifically on $\boldsymbol{\mu}$ or $\boldsymbol{r}$ or both,
    $$\begin{cases} \boldsymbol{r}_i = \boldsymbol{r} + \boldsymbol{p}_i\boldsymbol{\sigma}, & \text{if } \boldsymbol{r} \text{ is adjustable,} \\ \boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2), & \text{if } \boldsymbol{z} \text{ is adjustable.} \end{cases}$$
    Reconstruct the image from decoder g, and use project function P to restrict its maximal distortion: $\boldsymbol{x}_i = P(g(\boldsymbol{z}_i, \boldsymbol{r}_i))$.
    Compute the losses of these reconstructed image series $\boldsymbol{x}_i$ under targeted or untargeted setting following Eq. (3), then formulate the corresponding loss as $l_i = l_i - \text{mean}(l)$.
    Update $\boldsymbol{\mu}$ or $\boldsymbol{r}$ or both as follows:
    $$\begin{cases} \boldsymbol{r}_{t+1} \leftarrow \boldsymbol{r}_t - \frac{\eta}{b\boldsymbol{\sigma}} \sum_{i=1}^{b} l_i \boldsymbol{p}_i, & \text{if } \boldsymbol{r} \text{ is adjustable,} \\ \boldsymbol{\mu}_{t+1} \leftarrow \boldsymbol{\mu}_t - \frac{\eta}{b\boldsymbol{\sigma}} \sum_{i=1}^{b} l_i \boldsymbol{p}_i, & \text{if } \boldsymbol{z} \text{ is adjustable.} \end{cases}$$
  end if
end for

3.4. Improved versions of NP-attack

The basic version of the proposed NP-Attack is to optimize latent variables following the three variants mentioned in Section 3.3. The pre-trained NP is trained on benign examples. This basic version could directly and essentially show the efficiency of our proposed method as we do not add other tricks or surrogate models. In this section, we explore more on NP-Attack methods by adding some tricks, *e.g.*, with transferable priors or tiling skills.

3.4.1. NP-attack with transferable priors

The optimization procedure of $\boldsymbol{r}$, $\boldsymbol{z}$ in the previous section is designed without any surrogate model. However, in many scenarios, surrogate (white-box) models are allowed and could further improve query efficiency. A similar idea has been deployed in P-RGF and TREMBA, which could also benefit our NP-Attack as well. To be specific, we use the surrogate models to update the latent distribution variables $\boldsymbol{z}$ and $\boldsymbol{r}$ in backward propagation. With such guidance, the optimized starting point of $\boldsymbol{z}$ and $\boldsymbol{r}$ could gain some adversarial bias. Here, for simplicity, we take one surrogate model as an example to show how we incorporate the transferable bias into the pre-trained NP model. We samples $\boldsymbol{z}^0 \sim \mathcal{N}(\boldsymbol{\mu}^0, \boldsymbol{\sigma}^2)$ and $\boldsymbol{r}_0$ and reconstruct the image with the decoder g of the pre-trained NP model as $\boldsymbol{x}_{rec} = g(\boldsymbol{z}^0, \boldsymbol{r}^0)$ and feed it into the surrogate model. With F_s as the softmax output of the surrogate model, the loss functions are defined as:

$$l_s(\boldsymbol{x}_{rec}) := \begin{cases} \max(0, \max_{c \neq y} \log F_s(\boldsymbol{x}_{rec})_c - \log F_s(\boldsymbol{x}_{rec})_y), & \text{(a)}, \\ \max(0, \log F_s(\boldsymbol{x}_{rec})_y - \max_{c \neq y} \log F_s(\boldsymbol{x}_{rec})_c), & \text{(b)}, \end{cases} \tag{7}$$

where (a) for targeted attacks and (b) for untargeted attack, y is the true label in untargeted attack or the target label in targeted attack, and c is other labels except y. We can update $\boldsymbol{r}$ and $\boldsymbol{z}$ during the iterable backward propagation on the surrogate model as:

$$\begin{aligned} \boldsymbol{r}^{i+1} &= \boldsymbol{r}^i - \eta \tfrac{\partial}{\partial \boldsymbol{r}^i} l_s(\boldsymbol{x}_{rec}), & \text{if } \boldsymbol{r} \text{ is adjustable,} \\ \boldsymbol{\mu}^{i+1} &= \boldsymbol{\mu}^i - \eta \tfrac{\partial}{\partial \boldsymbol{\mu}^i} l_s(\boldsymbol{x}_{rec}), & \text{if } \boldsymbol{z} \text{ is adjustable.} \end{aligned} \tag{8}$$

With multiple surrogate models, we can feed the reconstructed images to all of them and sum up the loss function l_s for each individual one, then update $\boldsymbol{r}$ and $\boldsymbol{z}$. Note that the updating of $\boldsymbol{z}$ is not executable as the sampling operation of $\boldsymbol{z}$ is non-differential. So that we update the latent variables $\boldsymbol{r}$ and $\boldsymbol{\mu}$, and apply them as the start points $\boldsymbol{r}$ and $\boldsymbol{\mu}$ in the following NP-Attack. The further improvement confirms our NP-Attack also outperforms in the setting with surrogate models. The NP-Attack with transferable prior is introduced in Algorithm 2.

3.4.2. NP-attack with tiling tricks

We would like to further accelerate the query efficiency in NP-Attack by adopting the tiling tricks. The reasons come as follows: (1) the compression of searching space by our introduced NP-Attack works well compared with $\mathcal{N}$Attack [13], (2) adversarial perturbations tend to be similar on nearby pixels [28]. Similar query efficiency improvements are achieved in black-box attacks by Bandits [22]. Square Attack [28] also presents the superiority of square/patch-wise attacks. The tiling trick shows its remarkable efficiency, especially when using NP-Attack on robust models or high-dimensional images. Because robust models tend to focus more on low-frequency information, they are less sensitive to downsampling operations. Furthermore, high-dimensional images usually incur expensive computational and time costs for NP-Attack, which can be greatly alleviated by the tiling trick. Specifically, we first resize and downsample the original image. Next, we pre-train the NP model on these downsampled images, optimizing the distribution of the downsampled NP latent variables. We then

Algorithm 2 NP-Attack with Transferable Prior.

Input: benign image $\boldsymbol{x}$, label y, target neural network F, surrogate neural networks $\{F_{s^1}, \ldots F_{s^n}\}$, pre-trained NP model (encoder h, decoder g, aggregator a), learning rate η, transferable optimization iteration t

1: Compute the variables from encoder h on image $\boldsymbol{x}$: $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2), \boldsymbol{r} \leftarrow h(\boldsymbol{x})$.
2: Initial $\boldsymbol{r}^0 = \boldsymbol{r}$ and $\boldsymbol{\mu}^0 = \boldsymbol{\mu}$
3: **for** $i = 0$ **to** t **do**
4: Reconstruct the image with decoder g:$\boldsymbol{x}_{rec} = g(\boldsymbol{z}^i, \boldsymbol{r}^i)$, where $\boldsymbol{z}^i \sim \mathcal{N}(\boldsymbol{\mu}^i, \boldsymbol{\sigma}^2)$.
5: Compute the losses l_{sj} with the reconstructed image $\boldsymbol{x}_{rec}$ on surrogate models.
6: Sum up the losses:$L_s = \sum_{j=0}^{n} l_{sj}$.
7: Update $\boldsymbol{\mu}$ and $\boldsymbol{r}$:
$$\begin{cases} \boldsymbol{r}^{i+1} = \boldsymbol{r}^i - \eta \frac{\partial}{\partial \boldsymbol{r}^i} L_s(\boldsymbol{x}_{rec}), \\ \boldsymbol{\mu}^{i+1} = \boldsymbol{\mu}^i - \eta \frac{\partial}{\partial \boldsymbol{\mu}^i} L_s(\boldsymbol{x}_{rec}). \end{cases}$$
8: **end for**
9: Update the latent variables in pre-trained NP: $\boldsymbol{\mu}' = \boldsymbol{\mu}^t$ and $\boldsymbol{r}' = \boldsymbol{r}^t$.
10: Apply NP-Attack with $\mathcal{N}(\boldsymbol{\mu}', \boldsymbol{\sigma}^2), \boldsymbol{r}'$ in Line 1 in Algorithm 1.

upsample the optimized images back to the original images before feeding them into the classifier. With these tricks, NP-Attack can be significantly accelerated.

4. Experiment

In this section, we evaluate our method on three benchmark image classification datasets MNIST [33], CIFAR-10 [34] and ImageNet [35]. We compare our proposed NP-Attack with several score-based black-box attack techniques with regard to distortion (ϵ), average query count (*i.e.*, the number of evaluations on black-box model) and attack success rate (ASR). We firstly provide a comprehensive understanding of NP-Attack where we test the performance of our NP-Attack under different experimental settings. After that, we conduct untargeted and targeted attacks on benchmark datasets MNIST, CIFAR-10 and ImageNet to show the superiority of our method in reducing query counts. We denote NP-Attack-R as optimizing $\boldsymbol{r}$, NP-Attack-Z as optimizing $\boldsymbol{z}$, NP-Attack-RZ as optimizing both $\boldsymbol{r}$ and $\boldsymbol{z}$ simultaneously. The NP model with adversarial prior in transferable NP-Attack is optimized on WideResNet [36] or DenseNet [37] or ResNet18 [38], with one as the target model the other two as surrogate models. *Experimental Setups.* For MNIST, an MLP model is trained with three fully connected layers, achieving 98.50% accuracy. For CIFAR-10, WideResNet [36], DenseNet [37] and ResNet18 [38] are adopted, achieving 94.10%, 95.32% and 94.33% accuracy respectively. For ImageNet, we use the pre-trained Inception-V3 model [39], achieving 78% accuracy. The baseline query-based black-box techniques are ZOO [11], AutoZOOM [12], QL [21] and $\mathcal{N}$Attack [13]. The results of those compared methods are reproduced by the code released in their original paper with default settings.

NP Pre-training. For NP-Attack, NP models are firstly pre-trained on the whole training dataset of MNIST, CIFAR-10, and ImageNet, taking around 5, 45 and 72 hours on a single NVIDIA GTX 1080 TI GPU respectively. In addition, we also pre-train an NP model on the partial training dataset of CIFAR-10, so as to further evaluate the efficiency of our proposed NP-Attack. The dimensions of $\boldsymbol{r}$ and $\boldsymbol{z}$ in our pre-trained NP models are both 128. Multihead attention is applied in NP models. The optimizer is Adam with the learning rate set to be 1e-3 and the total training epoch set to be 50. Regarding the computational overhead of the pre-training, we would like to point out that the procedure of pre-training NP is over a set of the training data, while the black-box attack procedure is performed on a single image per time. The pre-trained model is used for free to perform black-box attacks when the pre-training is complete. Thus, the pre-training is an off-line operation, which is totally separated from the online black-box attack. The overhead of pre-training will not affect the black-box attack phase.

Table 1
Structure of the deterministic part of encoder h. (The input of Linear1 includes the pixel position: 3072 ($32 \times 32 \times 3$) $\times$ 3, the pixel value in RGB respectively: 3072 $\times$ 1, thus with a size of 3072 $\times$ 4 in total.).

Layer	Input	Output	Activation function
Linear1	3072 × 4	3072 × 128	ReLU
Linear2	3072 × 128	3072 × 128	ReLU
Linear3	3072 × 128	3072 × 128	ReLU
SelfAtt	3072 × 128	3072 × 128	-
CrossAtt	3072 × 128	3072 × 128	-
Linear4	3072 × 128	3072 × 128	-

Table 2
Structure of the sampled latent part of encoder h.

Layer	Input	Output	Activation function
Linear1	3072 × 4	3072 × 128	ReLU
Linear2	3072 × 128	3072 × 128	ReLU
Linear3	3072 × 128	3072 × 128	ReLU
SelfAtt	3072 × 128	3072 × 128	-
Mean	3072 × 128	3072 × 128	-
Linear4	3072 × 128	3072 × 128	-

Table 3
Structure of the latent part of decoder g. (The input of Linear1 includes the deterministic path: 3072 $\times$ 128, the sampled latent path: 3072 $\times$ 128 and the target pixel position: 3072 $\times$ 3, thus with a size of 3072 $\times$ 259 in total.).

Layer	Input	Output	Activation function
Linear1	3072 × 259	3072 × 128	ReLU
Linear2	3072 × 128	3072 × 128	ReLU
Linear3	3072 × 128	3072 × 128	ReLU
Linear4	3072 × 128	3072 × 1	ReLU

4.1. Empirical understanding of NP-attack

We evaluate the performance of NP-Attack under different experimental settings, including different **optimization methods** (optimizing $\boldsymbol{r}$, $\boldsymbol{z}$ or both), various **sample sizes, maximal iterations, maximal distortion, pre-trained NP models, transferable priors, tiling tricks** and **time costs**. Experiments are conducted on some correctly classified images selected from MNIST, CIFAR-10, or ImageNet test set. Experiments include both targeted and untargeted settings. (If not specifically mentioned, for convenience, we show untargeted results as default, and the targeted results will keep consistency.) **Optimization Methods.** As introduced in Section 3.3, the proposed NP-Attack could be divided into different variants regarding whether $\boldsymbol{r}$ or $\boldsymbol{z}$ or both is adjustable. Experimental results are summarized in Table 4. By default, the L_∞ norm distortion is set to 0.2 for MNIST and 0.031 for CIFAR-10, the sample size is set as $b = 10$ for MNIST and $b = 30$ for CIFAR-10, and the maximal iteration is set to 800 for MNIST and 400 for CIFAR-10 respectively. We select 200 images from each test set. For MNIST, NP-Attack-R performs better compared to the other two, while for CIFAR-10, the three variants perform very similarly. Adversarial examples generated by different optimization methods are shown in Fig. 3. More interestingly, we find that (black-box) adversarial perturbations exhibit clearly different patterns when optimizing $\boldsymbol{z}$ or $\boldsymbol{r}$. That is, adversarial perturbations generated by NP-Attack-R tend

Table 4
Evaluation of NP-Attack under untargeted setting by optimizing variables $\boldsymbol{r}$, $\boldsymbol{z}$ or both on 200 correctly classified images from MNIST and CIFAR-10 test sets.

Attack Method	Dataset	ASR	L_2 Dist	Query Count
NP-Attack-R	MNIST	100%	3.07	**1,190**
NP-Attack-Z		100%	3.55	1665
NP-Attack-RZ		100%	3.65	1460
NP-Attack-R	CIFAR-	100%	1.37	139
NP-Attack-Z	10	100%	1.32	**131**
NP-Attack-RZ		99.90%	1.40	140

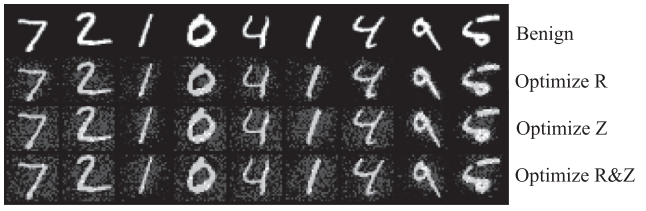


Fig. 3. Adversarial examples generated by different optimization methods under untargeted setting (NP-Attack-R/Z/RZ).

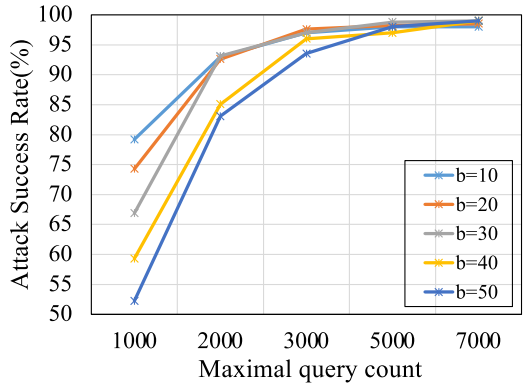


Fig. 4. Change of ASRs when limiting the query counts on MNIST with NP-Attack-R. Different curves represent for performances under different sample sizes.

to be centered around the main digits, while adversarial perturbations generated by NP-Attack-Z are scattered in the background. Furthermore, both the visual pattern of adversarial perturbations and the attack success rate performance in NP-Attack-RZ are somehow in a moderate degree between NP-Attack-R and NP-Attack-Z.

Sample Sizes. The updating of latent variables $\boldsymbol{z}$ and $\boldsymbol{r}$ highly depends on the average losses returned by the sampled examples in each optimization in Algorithm 1. The larger batch size in each iteration could provide more accurate gradients in each batch-wise optimization. Here we explore how the sample size will affect attack success rates (ASRs) and the query efficiency in NP-Attack on MNIST. The L_∞ norm distortion is set to 0.2. We test the sample size $b \in \{10, 20, 30, 40, 50\}$. Noted that query count is linearly related to sample size, as $Q = T \times b$, where T represents iteration count and Q represents query count. As the query count and attack success rate are highly related, we first plot the change of ASRs with the maximal query count in Fig. 4. It shows that our NP-Attack could achieve over 90% ASRs with different sample sizes when the maximal query count is larger than 7,000. Thus we count the ASR, L_2 norm distortion, and average query count setting the maximal query count to 7000 in Table 5. We can see that, with a larger sample size, the iteration T is reduced along with lower ASRs and larger average query counts. However, we can get smaller L_2 norm distortion by increasing the sample size.

Maximal iterations. The larger maximal iterations often imply more optimization steps and a larger chance to find an adversarial example. Here we use the L_∞ norm restriction for adversarial examples on MNIST. We set the distortion $\epsilon = 0.2$ and the sample size $b = 30$. Then we test the maximal iteration $T \in \{300, 400, 500, 600\}$. We count the ASR, L_2 norm distortion and average query count in Table 6. Experimental results show that a larger maximal iteration is more likely to increase both attack success rates (ASRs) and query counts.

Table 5
Evaluation of NP-Attack-R under untargeted setting on MNIST with different sample sizes. The maximal query count is 7000 and L_∞ norm distortion is restricted to 0.2. T refers to the number of iteration.

Sample Size (b)	T	ASR	L_2 Dist	Avg Query Count
10	700	**97.01%**	3.0146	**1,439**
20	350	94.03%	2.5684	1487
30	233	94.53%	2.4082	1577
40	175	92.54%	2.3279	1657
50	140	90.55%	**2.2694**	1869

Table 6
Evaluation of NP-Attack-R under untargeted setting on MNIST with different maximal iterations. The L_∞ norm distortion is restricted to 0.2 and sample size b is set to 30.

Maximal Iteration (T)	ASR	L_2 Dist	Avg Query Count
300	94.37%	3.04	997
400	95.65%	2.98	1045
500	97.01%	3.01	1139
600	**100%**	3.09	1226

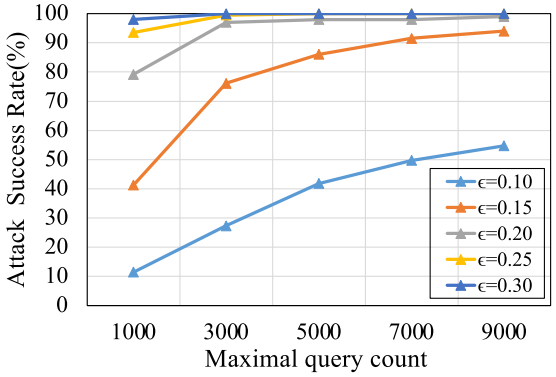


Fig. 5. Change of ASRs when limiting the query counts on MNIST with NP-Attack-R. Different curves represent different maximal distortion restrictions.

Maximal distortion. It is highly correlated between the distortion and query counts in black-box attack. The larger distortion implies the larger space to search for adversarial examples, thus the less difficulty in black-box attacks, which represents for the higher attack success rates and lower query counts. Here we use the L_∞ norm restriction for adversarial examples on MNIST. We set the distortion $\epsilon \in \{0.1, 0.15, 0.2, 0.25, 0.3\}$ and the sample size $b = 10$. The change of ASRs with various maximal query counts is shown in Fig. 5. It is found that in general, the larger distortion ϵ returns the higher ASR. To be specific, when $\epsilon = 0.3$, our NP-Attack achieves the 90% ASR with only 1000 maximal query counts. Increasing query counts could improve the ASR. Specifically, with 9000 query counts, NP-Attack can achieve nearly 100% ASRs when ϵ is set to be larger than 0.2.

Ablation Study on Pre-trained NP Models. We briefly study the effectiveness of NP-Attack when NP models are pre-trained on external or only a small fraction of training data. To be more specific, we try to use the NP models pre-trained on ImageNet or on 10% training data of CIFAR-10 in NP-Attack. We perform untargeted attacks on CIFAR-10 test set for ResNet18. We restrict the L_∞ norm distortion ϵ to 0.031, learning rate $\eta = 0.1$, sample size

Table 7
Evaluation of untargeted NP-Attack for ResNet18 on CIFAR-10, using differently pre-trained NP models.

Attack Method	Pre-trained NP Models	ASR	#Query
NP-Attack-R (ours)	Whole CIFAR-10 (default)	100%	226
	ImageNet	100%	304
	Partial CIFAR-10	100%	383
NP-Attack-Z (ours)	Whole CIFAR-10 (default)	100%	222
	ImageNet	100%	452
	Partial CIFAR-10	100%	575
NP-Attack-RZ (ours)	Whole CIFAR-10 (default)	100%	245
	ImageNet	100%	391
	Partial CIFAR-10	100%	393
ZOO	–	100%	222,976
AutoZOOM-BiLIN	–	99.70%	5760
AutoZOOM-AE	–	99.60%	5880
QL	–	83.05%	1829
$\mathcal{N}$Attack	–	100%	368

Table 8
Evaluation of untargeted NP-Attack for WideResNet (WRN) w/wo transferable prior on CIFAR-10.

	Attack Method	Trans Prior	ASR	#Query
Untargeted	NP-Attack-R	×	100%	139
		✓	100%	**31**
	NP-Attack-Z	×	100%	131
		✓	100%	**30**
	NP-Attack-RZ	×	100%	140
		✓	100%	**37**
Targeted	NP-Attack-R	×	100%	897
		✓	100%	**453**
	NP-Attack-Z	×	100%	827
		✓	100%	**376**
	NP-Attack-RZ	×	100%	905
		✓	99.78%	**518**

$b = 30$. Experimental results are shown in Table 7. We find that even though these two NP models are not as effective as the original NP model pre-trained on the whole training data of CIFAR-10, they show better results than some baseline methods. Experimental results demonstrate the availability of our proposed NP-Attack in real scenarios, even when without full knowledge of the whole training data. For example, the query count of NP-Attack-R using an NP model pre-trained with ImageNet is 304, which is better than SOTA $\mathcal{N}$Attack.

Ablation study on transferable priors. We briefly compare the experimental results of NP-Attack with and without the transferable prior. We perform untargeted and targeted attacks on CIFAR-10 test set with WideResNet (WRN) as the target model. To introduce transferable prior, the standard trained ResNet18 and DenseNet-BC-100-12 are deployed as surrogate models. We restrict the L_∞ norm distortion ϵ to 0.031, learning rate $\eta = 0.1$, for untargeted attacks sample size $b = 30$, for targeted attacks sample size $b = 50$. The results of attack success rate (ASR) and average query count of different versions of NP-Attack are reported in Table 8. The results show that the ASR keeps the same as the highest 100% in the two versions of NP-Attack, however, with transferable prior, the query efficiency is reduced by over 80% for untargeted attacks and 50% for targeted attacks. The performance in the latter experiments confirms that the NP-Attack with transferable prior could greatly outperform other transferable black-box attack methods.

Ablation study on tiling tricks. Then we would like to further explore the parameters in tiling tricks. Here, the NP model is pre-trained on the training data of CIFAR-10 downsampled by the tiling size. The standard trained (STD) and adversarially trained (ADV) ResNet18 are deployed as the black-box target models. The tiling step is set to be $1\times$ 1 (without tiling tricks), 2×2, and 4 $\times$ 4 to explore the effects of the patch size. With respect to the hyper-parameters of black-box attacks, the learning rate is set as $\eta = 0.1$ and the distortion is set as $\epsilon = 0.031$. For STD model, we set sample size as $b = 30$, maximal iteration as $T = 400$, while for ADV model, we set sample size as $b = 100$, maximal iteration as $T = 100$. The results in Table 9 show that these models show different tendencies. NP-Attack-R could benefit a lot from the tiling trick when attacking the robust model, as the robust model by adversarial training [6,40–42] prefers to capture more low-frequency features [43,44]. The query count is improved from 1923 to 1146 when applying the 2×2 tiling trick, while the ASR is improved from 39.06% to 41.26%. In the contrast, the tiling tricks could harm on the non-robust model, as the models by standard training rely more on the high-frequency features and the tiling trick could compress the noises added in the high-frequency domain, which restricts the perturbing effect. The ASR is slightly decreased from 100% to 99.20%, while the query count increases from 226 to 983. When testing with a tiling size of 4 $\times$ 4, both the performance on STD and ADV models degenerate for the reason that compressing the images by 4 times smaller than the original one results in excessive information loss for CIFAR-10.

Table 9
Evaluation of untargeted NP-Attack for ResNet18 w/wo tiling tricks on CIFAR-10.

Attack Method	Model	Tiling Size	ASR	Query Count
NP-Attack-R	STD	1 × 1	100%	**226**
		2 × 2	99.20%	983
		4 × 4	87.91%	1470
	ADV	1 × 1	39.06%	1923
		2 × 2	41.26%	**1,146**
		4 × 4	30.87%	2108

Table 10
Evaluation of untargeted NP-Attack w/wo tiling tricks on ImageNet for STD Inception-V3 model. 'Time' refers to the reconstruction time cost per image.

Attack Method	Tiling Size	ASR	#Query	Time (s)
NP-Attack-R	1 × 1	100%	733	83
	2 × 2	100%	600	30
	4 × 4	100%	**574**	12
	8 × 8	100%	708	3

Furthermore, we also show the superiority of tiling tricks in reducing reconstruction times for high-dimensional data such as ImageNet. We conduct untargeted attack on ImageNet with a standard trained (STD) Inception-V3 as the target model. The tiling step is set to 1 $\times$ 1, 2 $\times$ 2, 4 $\times$ 4, and 8 $\times$ 8. The learning rate η is 0.05, sample size b is 100, maximal iteration T is 600 and the L_∞ norm distortion is set to 0.05. We report the attack success rate (ASR), the average number of query (#Query), and average time (s) to reconstruct adversarial examples with different tiling sizes in Table 10. Experimental results show that for high-dimensional data, appropriate tiling tricks can both improve the query efficiency and greatly reduce the reconstruction time. Tiling size of 4 $\times$ 4 on ImageNet owes the best trade-off between the query efficiency and the reconstruction time.

Ablation study on time costs. We further study the time costs of our proposed NP-Attack in online generating adversarial examples. Note that the procedure of pre-training NP is over a set of the training data and is an off-line operation, which has been discussed above. Here we use the L_∞ norm distortion for 200 untargeted adversarial examples on CIFAR-10. We set the distortion $\epsilon = 0.031$ and the sample size $b = 30$. Then we test the maximal iteration $T = 400$. We count the time costs per image per iteration in Table 11. Experimental results show that the time costs of our proposed method are acceptable, as the online optimization procedure with pre-trained NP models is quite efficient.

Table 11
Time Costs (per image, per iteration) of untargeted NP-Attack for WideResNet (WRN) w/wo transferable prior on CIFAR-10.

Attack Method	Trans Prior	(On-line) Time Costs (s)
NP-Attack-R	×	0.43
	✓	0.34
NP-Attack-Z	×	0.42
	✓	0.32
NP-Attack-RZ	×	0.56
	✓	0.54
ZOO	–	0.54
AutoZOOM-BiLIN	–	0.68
AutoZOOM-AE	–	0.93
QL	–	0.45
$\mathcal{N}$Attack	–	0.22

4.2. Comprehensive evaluations on MNIST and CIFAR-10

To comprehensively evaluate NP-Attack on both MNIST and CIFAR-10, we randomly select 1000 correctly classified images from the test set for untargeted attack. For targeted attack, 100 correctly classified images are selected from the test set, for each image the target labels are set to the other 9 classes, and a total of 900 attacks are performed. In our experiments, the L_∞ norm distortion is restricted to 0.2 for MNIST and 0.031 for CIFAR-10 following the setting of [22]. For untargeted attack, the maximal iteration is $T = 600$ and sample size is $b = 30$ while for targeted attack $T = 600$, $b = 50$. The learning rate is $\eta = 0.1$. A total query count is obtained by multiplying the number of iterations and the query count per iteration. It is worth mentioning that for a fair comparison to the counterparts, in this section we conduct experiments with the plain NP-Attack without introducing surrogate models and tiling tricks.

Note that the query count per iteration varies in different black-box attack techniques. ZOO uses the parallel coordinate-wise estimation with a batch of 128 pixels, resulting in 256 query counts per iteration. In AutoZOOM, the attack stages could be divided into initial attack success and post-success fine-tuning. In each iteration, the number of random vector is set to 1 at the first stage to find the initial attack success examples and then set to q at the second stage to reduce the distortion at the same level as other techniques. Thus the query count for AutoZOOM is $q + 1$ per iteration. For QL, $\mathcal{N}$Attack and our NP-Attack, query count in each iteration is the sample size b. For a fair comparison, we set the same sample size for these three NES algorithm-based methods. Besides, both ZOO and AutoZOOM are optimization-based methods that quickly attack the model successfully and generate adversarial examples with quite a large distortion and continuously perform post-success fine-tuning to reduce the distortion. We report the final fine-tuning query count and the distortion after fine-tuning for ZOO and AutoZOOM.

Experimental results are shown in Tables 12 and 13. We report the ASRs, L_2 norm distortion, L_∞ norm distortion, and average query counts. For CIFAR-10, we adopt three different architectures, *i.e.*, WideResNet, DenseNet and ResNet18, as the targeted models. Among all three architectures, our NP-Attack can outperform compared baseline methods. Compared to ZOO, our NP-Attack can reduce the query count by 98.86% and 99.95% on MNIST and CIFAR-10 respectively under untargeted attack setting, while for targeted attack, NP-Attack reduces the query count by 98.34% and 99.64% on MNIST and CIFAR-10. The comparison shows that though ZOO achieves 100% ASR, it is far from query efficient, requiring over 100,000 queries to generate considerable adversarial examples, implying high costs in computation and time. When it comes to AutoZOOM, it could significantly reduce the query count by proposing an adaptive random gradient estimation strategy. AutoZOOM-BiLIN and AutoZOOM-AE leverage a simple bilinear resizer or auto-encoder as the decoder respectively to reduce the dimension of adversarial perturbations. AutoZOOM-based method can attack the model successfully with quite fewer initial success query counts, but the distortion is unacceptable at the initial success. It still requires much more queries to fine-tuning. Compared to AutoZOOM, our NP-Attack methods outperform not only in query counts but also in the L_∞ norm distortion. QL utilizes the NES algorithm to estimate the gradient, thus the performance of such a technique hinges on the quality of the estimated gradient. In Table 12, the ASR of QL in untargeted attack v.s. targeted attack is 96.62% v.s. 99.67%. While in Table 13, among different models, QL obtains higher ASRs under targeted attack settings. QL shows its superiority in targeted attacks for the reason that gradient-based methods are easier to find the targeted direction. In contrast, $\mathcal{N}$Attack and our NP-Attack both utilize NES to estimate the adversarial distribution. Compared to $\mathcal{N}$Attack, our NP-Attack could achieve higher ASRs and fewer query counts under both targeted or untargeted settings, due to the outstanding distribution modeling capacity of the NP model. In general, among all the compared methods, NP-Attack obtains the best trade-off between query counts and distortion.

4.3. Comprehensive evaluations on ImageNet

The dimension of the images in ImageNet is relatively larger, which requires more queries to generate adversarial examples. We randomly select 100 correctly classified images from the test set to perform untargeted and targeted black-box attacks on ImageNet. For each image in targeted attack, a random label except the ground truth one out of 1000 classes is selected to serve as the target. The L_∞ norm distortion is restricted to 0.05 following the setting of [21,22], $T = 600$, $b = 100$ for both untargeted attacks and targeted attacks and the learning rate $\eta = 0.05$. To introduce transferable prior, standard trained ResNet50 and DenseNet121 are adopted as the surrogate models and Inception-V3 as the black box target model. To improve the query efficiency on images with large sizes, ZOO and AutoZOOM utilize the techniques such as hierarchical attack and compressing dimension of attack space. For our NP-Attack, we also perform such compression with tiling tricks. The tiling size is 4×4 for untargeted attack and 2×2 for targeted attack. As the empirical results shown in Table 10, the tiling tricks can trade off query efficiency and the adversarial examples reconstruction times. To be specific, we resize the images to $32 \times 32 \times 3$ to pre-train the NP model, then we add perturbations on each $32 \times 32 \times 3$ patch, which is cut independently from the 4×4 downsampled images and finally upsampled to the original dimension of the images. Although using the attack dimension reduction, our NP-Attack method still outperforms.

Experimental results are summarized in Table 14. Due to the large image sizes, ZOO suffers from low ASRs and tremendous model evaluations, especially for targeted attack. For the other three compared attacks, they achieve 100% ASRs at the cost of over 10,000 queries in targeted attack and over 1000 queries in untargeted attack. While for our NP-Attack, NP-Attack-R can achieve 100% ASR with only 273 queries, yielding a 98.25% query reduction ratio compared to ZOO in untargeted attack. In targeted attack, NP-Attack-R/Z/RZ reduce the query counts to 6,270, 5,647, and 7690, respectively, which exceed all the compared methods. This demonstrates that our NP-Attack can scale well to ImageNet set. Adversarial examples generated by NP-Attack-R † are shown in Fig. 6. It is demonstrated that though we perturb each $32 \times 32 \times 3$ patch in the original images, the perturbation is still imperceptible.

Table 12
Adversarial evaluation of black-box attacks on MNIST. L_∞ norm distortion ϵ is 0.2. The attack success rate (ASR), L_2 norm distortion (L_2 Dist), and average query count (#Query) under untargeted and targeted attacks are reported.

Untargeted				Targeted			
Attack Method	ASR	L_2 Dist	#Query	Attack Method	ASR	L_2 Dist	#Query
ZOO	100%	1.12	107,264	ZOO	100%	1.64	128,768
AutoZOOM-BiLIN	100%	2.01	9129	AutoZOOM-BiLIN	99.89%	2.78	9401
AutoZOOM-AE	100%	2.62	10,202	AutoZOOM-AE	99.89%	3.74	10,380
QL	96.62%	3.38	2549	QL	99.67%	3.09	2693
$\mathcal{N}$Attack	95.09%	2.14	4357	$\mathcal{N}$Attack	98.22%	3.33	5,981
NP-Attack-R(Ours)	100%	3.09	**1,226**	NP-Attack-R(Ours)	100%	3.72	2693
NP-Attack-Z(Ours)	99.90%	3.55	1680	NP-Attack-Z(Ours)	100%	3.94	2605
NP-Attack-RZ(Ours)	100%	3.65	1460	NP-Attack-RZ(Ours)	99.78%	4.00	**2,136**

Table 13
Adversarial evaluation of black-box attacks on CIFAR-10. L_∞ norm distortion ϵ is 0.031. The attack success rate (ASR), L_2 norm distortion (L_2 Dist), and average query count (#Query) under untargeted and targeted attack are reported. WideResNet, DenseNet, and ResNet18 are adopted as the target model respectively.

	Attack Method	WRN			DenseNet			ResNet18		
		ASR	L_2 Dist	#Query	ASR	L_2 Dist	#Query	ASR	L_2 Dist	# Query
Untargeted	ZOO	100%	0.12	208,384	100%	0.15	220,160	100%	0.18	222,976
	AutoZOOM-BiLIN	100%	1.56	8113	100%	2.04	4529	99.70%	1.31	5760
	AutoZOOM-AE	100%	1.88	7113	100%	2.65	4611	99.60%	1.61	5880
	QL	96.05%	1.20	1487	92.23%	1.21	1817	83.05%	1.82	1829
	$\mathcal{N}$Attack	96.90%	1.53	315	99.70%	0.83	354	100%	0.86	368
	NP-Attack-R(Ours)	100%	1.37	139	100%	1.41	144	100%	1.38	226
	NP-Attack-Z(Ours)	100%	1.32	**131**	100%	1.36	**135**	100%	1.34	**222**
	NP-Attack-RZ(Ours)	99.90%	1.40	140	100%	1.44	142	100%	1.41	245
Targeted	ZOO	99.52%	0.19	230,912	100%	0.27	222,720	100%	0.03	198,912
	AutoZOOM-BiLIN	100%	2.13	8266	99.89%	2.30	8404	99.89%	2.29	8279
	AutoZOOM-AE	100%	2.78	8217	100%	3.02	8254	100%	3.00	8292
	QL	100%	1.43	963	99.78%	1.44	1286	99.55%	1.45	2094
	$\mathcal{N}$Attack	99.77%	0.95	1508	100%	1.25	784	99.59%	1.31	1276
	NP-Attack-R(Ours)	100%	1.41	897	100%	1.40	537	100%	1.42	930
	NP-Attack-Z(Ours)	100%	1.37	**827**	100%	1.36	**498**	100%	1.38	**849**
	NP-Attack-RZ(Ours)	100%	1.44	905	100%	1.44	551	100%	1.45	901

Table 14
Adversarial evaluation of black-box attacks on ImageNet for Inception-V3. † refers to the results of NP-Attack with transferable prior and tiling tricks.

Untargeted				Targeted			
Attack Method	ASR	L_2 Dist	#Query	Attack Method	ASR	L_2 Dist	#Query
ZOO	90%	1.20	15,631	ZOO	78%	3.43	$2.11\text{x}10^6$
AutoZOOM-BiLIN	100%	9.34	3024	AutoZOOM-BiLIN	100%	11.26	14,228
QL	100%	17.72	3985	QL	100%	17.39	33,360
$\mathcal{N}$Attack	100%	24.01	2075	$\mathcal{N}$Attack	100%	24.14	14,229
Bandits	100%	–	1165	Bandits	100%	–	25,341
NP-Attack-R†(Ours)	100%	14.01	**273**	NP-Attack-R†(Ours)	100%	19.52	6270
NP-Attack-Z†(Ours)	100%	13.43	346	NP-Attack-Z†(Ours)	100%	15.88	**5,647**
NP-Attack-RZ†(Ours)	100%	14.46	318	NP-Attack-RZ†(Ours)	99.01%	20.73	7690

4.4. Further evaluations on robust models

The evaluation in the previous section all refers to the non-robust models, *i.e.*, models by standard training (STD). However, robust models tend to show different performance from the non-robust ones in both white-box and black-box attacks. So in this section, we conduct experiments on adversarially trained (ADV) models for example, which are one kind of public acknowledged robust models. We apply an adversarially trained ResNet18 as the black-box target model. The standard trained ResNet50, VGG16, and WideResNet models are applied as surrogate models. The maximal iteration is set as $T = 100$, sample size is set as $b = 100$, learning rate is set as $\eta = 0.1$. The baseline methods are Bandits with the tiling patch set to be 2×2. P-RGF and TREMBA both use the same surrogate models and tiling tricks as our NP-Attack. Meta Attack uses the pre-trained meta model.

Experimental results in Table 15 show that our method could overall outperform under the help of the same surrogate models. On robust models, the attack success rate is hard to perform well for other baseline methods, which is greatly reduced from the white-box attack results. The ASR of our NP-Attack with the transferable prior and tiling tricks could achieve a similar result to white-box attack ASR, *i.e.*, 42.36% with fewer query counts, com-

Table 15
Adversarial evaluations on CIFAR-10 for robust ResNet18. † refers to the results of NP-Attack with the transferable prior and tiling tricks.

Attack Method	ASR	L_2 Dist	#Query
Bandits	22.87%	1.65	2,803
P-RGF	13.45%	1.76	4,601
TREMBA	31.45%	–	3417
MetaAttack	36.33%	–	2,918
NP-Attack-R†(Ours)	**42.36%**	1.59	**1,201**
NP-Attack-Z†(Ours)	38.46%	1.58	1,875
NP-Attack-RZ†(Ours)	42.06%	1.59	1294

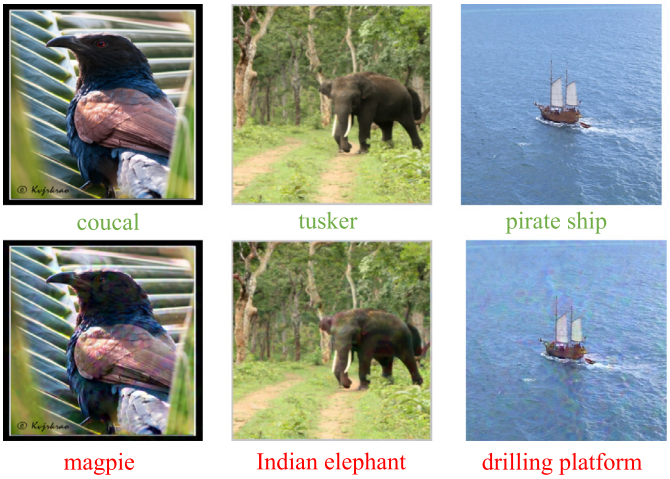


Fig. 6. Adversarial examples of ImageNet, which are generated by NP-Attack-R† under untargeted attack setting and are bounded by L_∞=0.05. Top row shows the original images with ground-truth labels, while the bottom row are the adversarial examples with (wrong) predicted labels.

pared to the best baseline method, Bandits with 22.87% ASR and 2803 query counts. Note that all the experiments are conducted with the same surrogate models, which means we compare the efficiency of transfer combined methods in a fair way. The superiority mainly comes from the query efficiency and the better usage of surrogate models' gradients in our NP-Attack.

5. Conclusion

In this paper, we propose NP-Attack by introducing the neural process (NP) model into black-box attacks. NP returns an instance-wise latent distribution for every image and can better preserve its structure or high-level information. Then (black-box) adversarial perturbations can be optimized on the latent distribution with compressed dimensions, instead of the original optimization on the high-dimensional pixel level. Experimental results demonstrate the superiority of both query efficiency and attack success rates (ASRs) in our NP-Attack. More evaluations on robust models or more data sets confirm the superiority of our proposed method among the evolution strategy series methods. Note that the surrogate models are not essential in our NP-Attack when considering the improvement effect, although it could be further improved with such surrogate models.

Future work. We have highlighted the effectiveness of NP-Attack in score-based (black-box) adversarial attack setting. Notably, NP model shows its superiority mainly because it could model the sample-wise data distribution, and is more suitable for adversarial attacks compared with normal auto-encoder (AE) models. Recently, besides black-box attacks, a new series of adversarial attack methods have been proposed, dubbed no-box adversarial attacks, which require to generate adversarial examples without any returned information from the target model or without a local classification surrogate model. Existing works, in order to estimate the adversarial perturbations, either add some predefined and regular noise patterns [45] on benign images or use AE models [46] to generate worse noises. In future work, we will try to deploy the NP model to optimize the manifold of adversarial examples and improve the attack efficiency of no-box adversarial attacks.

Data availability

Query Efficient Black-box Adversarial Attack on Deep Neural Networks (Mendeley Data)

Declaration of Competing Interest

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

Acknowledgments

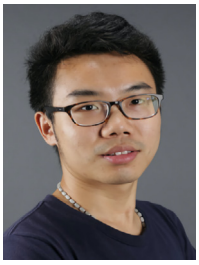
Yisen Wang is partially supported by the NSF China (No. 62006153), Project 2020BD006 supported by PKU-Baidu Fund, and Open Research Projects of Zhejiang Lab (No. 2022RC0AB05). Shu-Tao Xia is supported in part by the National Natural Science Foundation of China under Grant 62171248, and the PCNL KEY project (PCL2021A07).

References

[1] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: IEEE International Conference on Computer Vision, 2017.
[2] X. Ma, Y. Niu, L. Gu, Y. Wang, Y. Zhao, J. Bailey, F. Lu, Understanding adversarial attacks on deep learning based medical image analysis systems, Pattern Recognition 110 (2021) 107332.
[3] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, Y. Bengio, End-to-end attention-based large vocabulary speech recognition, in: International Conference on Acoustics, Speech and Signal Processing, 2016.
[4] Z. Xiong, W. Li, Q. Han, Z. Cai, Privacy-preserving auto-driving: a gan-based approach to protect vehicular camera data, in: IEEE International Conference on Data Mining, 2019.
[5] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, in: International Conference on Learning Representations, 2015.
[6] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu, Towards deep learning models resistant to adversarial attacks, in: International Conference on Learning Representations, 2018.
[7] S. Liu, P.-Y. Chen, X. Chen, M. Hong, signsgd via zeroth-order oracle, in: International Conference on Learning Representations, 2018.
[8] L. Huang, S. Wei, C. Gao, N. Liu, Cyclical adversarial attack pierces black-box deep neural networks, Pattern Recognition (2022) 108831.
[9] D. Wu, Y. Wang, S.-T. Xia, J. Bailey, X. Ma, Skip connections matter: On the transferability of adversarial examples generated with resnets, in: International Conference on Learning Representations, 2020.
[10] X. Wang, J. Ren, S. Lin, X. Zhu, Y. Wang, Q. Zhang, A unified approach to interpreting and boosting adversarial transferability, in: International Conference on Learning Representations, 2021.
[11] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, C.-J. Hsieh, Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models, arXiv preprint arXiv:1708.03999 (2017).
[12] C.-C. Tu, P. Ting, P.-Y. Chen, S. Liu, H. Zhang, J. Yi, C.-J. Hsieh, S.-M. Cheng, Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks, in: AAAI Conference on Artificial Intelligence, 2019.
[13] Y. Li, L. Li, L. Wang, T. Zhang, B. Gong, Nattack: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks, in: International Conference on Machine Learning, 2019.
[14] H. Li, X. Xu, X. Zhang, S. Yang, B. Li, Qeba: Query-efficient boundary-based blackbox attack, in: IEEE Conference on Computer Vision and Pattern Recognition, 2020.
[15] J. Chen, Q. Gu, Rays: A ray searching method for hard-label adversarial attack, in: ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020.
[16] C. Ma, X. Guo, L. Chen, J.-H. Yong, Y. Wang, Finding optimal tangent points for reducing distortions of hard-label attacks, Advances in Neural Information Processing Systems, 2021.
[17] J. Du, H. Zhang, J.T. Zhou, Y. Yang, J. Feng, Query-efficient meta attack to deep neural networks, in: International Conference on Learning Representations, 2020.
[18] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D.J. Rezende, S. Eslami, Y.W. Teh, Neural processes, arXiv preprint arXiv:1807.01622 (2018).
[19] Y. Bai, Y. Zeng, Y. Jiang, Y. Wang, S.-T. Xia, W. Guo, Improving query efficiency of black-box adversarial attack, in: European Conference on Computer Vision, 2020.
[20] D.P. Kingma, M. Welling, Auto-encoding variational bayes, in: International Conference on Learning Representations, 2014.
[21] A. Ilyas, L. Engstrom, A. Athalye, J. Lin, Black-box adversarial attacks with limited queries and information, in: International Conference on Machine Learning, 2018.
[22] A. Ilyas, L. Engstrom, A. Madry, Prior convictions: Black-box adversarial attacks with bandits and priors, arXiv preprint arXiv:1807.07978 (2018).
[23] S. Cheng, Y. Dong, T. Pang, H. Su, J. Zhu, Improving black-box adversarial attacks with a transfer-based prior, Advances in Neural Information Processing Systems, 2019.
[24] Z. Huang, T. Zhang, Black-box adversarial attack with transferable model-based embedding, in: International Conference on Learning Representations, 2019.

[25] A. Al-Dujaili, U.-M. O'Reilly, Sign bits are all you need for black-box attacks, in: International Conference on Learning Representations, 2019.
[26] S. Moon, G. An, H.O. Song, Parsimonious black-box adversarial attacks via efficient combinatorial optimization, in: International Conference on Machine Learning, 2019.
[27] L. Meunier, J. Atif, O. Teytaud, Yet another but more efficient black-box adversarial attack: tiling and evolution strategies, arXiv preprint arXiv:1910.02244 (2019).
[28] M. Andriushchenko, F. Croce, N. Flammarion, M. Hein, Square attack: a query–efficient black-box adversarial attack via random search, in: European Conference on Computer Vision, 2020.
[29] A.G.d.G. Matthews, J. Hron, M. Rowland, R.E. Turner, Z. Ghahramani, Gaussian process behaviour in wide deep neural networks, in: International Conference on Learning Representations, 2018.
[30] M. Wistuba, N. Schilling, L. Schmidt-Thieme, Scalable gaussian process-based transfer surrogates for hyperparameter optimization, Machine Learning 107 (1) (2018) 43–78.
[31] H. Kim, A. Mnih, J. Schwarz, M. Garnelo, A. Eslami, D. Rosenbaum, O. Vinyals, Y.W. Teh, Attentive neural processes, in: International Conference on Learning Representations, 2018.
[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in Neural Information Processing Systems, 2017.
[33] Y. Lécun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.
[34] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images, Technical Report, University of Toronto (2009).
[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009.
[36] S. Zagoruyko, N. Komodakis, Wide residual networks, arXiv preprint arXiv:1605.07146 (2016).
[37] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017.
[38] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016.
[39] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016.
[40] D. Wu, S.-T. Xia, Y. Wang, Adversarial weight perturbation helps robust generalization, Advances in Neural Information Processing Systems, 2020.
[41] Y. Wang, D. Zou, J. Yi, J. Bailey, X. Ma, Q. Gu, Improving adversarial robustness requires revisiting misclassified examples, in: International Conference on Learning Representations, 2020.
[42] Y. Wang, X. Ma, J. Bailey, J. Yi, B. Zhou, Q. Gu, On the Convergence and Robustness of Adversarial Training, in: International Conference on Machine Learning, 2019.
[43] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F.A. Wichmann, W. Brendel, Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness, in: International Conference on Learning Representations, 2018.
[44] J. Ren, D. Zhang, Y. Wang, L. Chen, Z. Zhou, Y. Chen, X. Cheng, X. Wang, M. Zhou, J. Shi, et al., A unified game-theoretic interpretation of adversarial robustness, Advances in Neural Information Processing Systems, 2021.
[45] Q. Zhang, C. Zhang, C. Li, J. Song, L. Gao, H.T. Shen, Practical no-box adversarial attacks with training-free hybrid image transformation, arXiv preprint arXiv:2203.04607 (2022).
[46] Q. Li, Y. Guo, H. Chen, Practical no-box adversarial attacks against dnns, Advances in Neural Information Processing Systems, 2020.

Yang Bai received the Bachelor degree from Department of Electronic Engineering, Tsinghua University in 2017. She is currently a Ph.D. student in Tsinghua University. Her research interests include trusty AI studies in machine learning and computer vision.



Dr. Yisen Wang is an assistant professor at the School of Intelligence Science and Technology, Peking University. He got his Ph.D. degree from Tsinghua University. His research interests are adversarial machine learning and weakly/self-supervised learning.

Yuyuan Zeng received the B.S. degree in Computer Science from Nankai University, Tianjin, China, in 2019. Currently she is pursuing the M.S. degree in Tsinghua Shenzhen International Graduate School, Tsinghua University, Guangdong, China. Her research interests include deep learning, machine learning and computer vision.

Dr. Yong Jiang received his M.S. and Ph.D. degrees in computer science from Tsinghua University, China, in 1998 and 2002, respectively. Since 2002, he has been with the Graduate School of Shenzhen of Tsinghua University, Guangdong, China, where he is currently a full professor. His research interests include Internet architecture and its protocols, IP routing technology, machine learning, etc.

Dr. Shu-Tao Xia received the B. S. degree in mathematics and the Ph.D. degree in applied mathematics from Nankai University, Tianjin, China, in 1992 and 1997, respectively. Since January 2004, he has been with the Graduate School at Shenzhen of Tsinghua University, Guangdong, China. He is now a full professor there. From March 1997 to April 1999, he was with the research group of information theory, Department of Mathematics, Nankai University, Tianjin, China. From September 1997 to March 1998 and from August to September 1998, he visited the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong. His current research interests include coding and information theory, networking, and machine learning.