Improving Query Efficiency of Black-box Adversarial Attack

Yang Bai[1,4,⋆], Yuyuan Zeng[2,4,⋆],
Yong Jiang[1,2,4,†], Yisen Wang[3,†], Shu-Tao Xia[2,4], and Weiwei Guo[5]

[1] Tsinghua Berkeley Shenzhen Institute, Tsinghua University
[2] Tsinghua Shenzhen International Graduate School, Tsinghua University
[3] Shanghai Jiao Tong University
[4] PCL Research Center of Networks and Communications, Peng Cheng Laboratory
[5] vivo AI Lab

Abstract. Deep neural networks (DNNs) have demonstrated excellent performance on various tasks, however 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 (inaccessible target model) are of 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. Extensive experiments show that NP-Attack could greatly decrease the query counts under the black-box setting. Code is available at `https://github.com/Sandy-Zeng/NPAttack`.

Keywords: Black-box Adversarial Attack, Adversarial Distribution, Query Efficiency, Neural Process

1 Introduction

Deep neural networks (DNNs) have been deployed on many real-world complex tasks and demonstrated excellent performance, such as computer vision [12], speech recognition [28], and natural language processing [7]. However, DNNs are found vulnerable to adversarial examples, *i.e.*, DNNs will make incorrect predictions confidently when inputs are added with some well designed imperceptible perturbations [11], thus various adversarial defense methods are proposed [2, 29,

---

⋆ Equal contribution. ({y-bai17, zengyy19}@mails.tsinghua.edu.cn)
† Corresponding authors: Yisen Wang (eewangyisen@gmail.com) and Yong Jiang (jiangy@sz.tsinghua.edu.cn).

30]. Adversarial examples can be crafted following either a white-box setting (the adversary has full access to the target model) or a black-box setting (the adversary has no information of the target model). White-box methods such as Fast Gradient Sign Method (FGSM) [11], Projected Gradient Decent (PGD) [21], Carlini & Wagner (CW) [3] and other universal attacks [23, 19] only pose limited threats to DNN models which are usually kept secret in practice. Meanwhile transferability-based black-box attacks, *e.g.*, momentum boosting [8] and skip gradient [33], need to train a surrogate model separately and only obtain a moderate attack success rate.

As modern machine learning systems, *e.g.*, Google Cloud Vision API2 (`https://cloud.google.com/vision/`) and Google Photos3 (`https://photos.google.com/`), are often provided as a kind of service, one common scenario is that we can query the system in a number of times and get the output results [1]. Based on this, *query*-based black-box attacks that directly generate adversarial examples on the target model are proposed such as ZOO [4]. These query-based methods could bring almost 100% attack success rates while their query complexity is quite high (not acceptable). Therefore, how to significantly reduce the query complexity while maintaining the attack success rate simultaneously is still an open problem. There are several existing work to reduce the query complexity. For examples, AutoZOOM [26] compresses the attack dimension while QL [13] adopts an efficient gradient estimation strategy. Different from the above example-wise adversarial example generation, $\mathcal{N}$Attack [18] proposes to model the adversarial distribution and sample from it to generate adversarial examples, which indeed reduces the query counts. However, the distribution is based on the simple pixel-wise mapping functions (*e.g.*, *tanh*), which is the bottleneck of its query complexity.

Inspired by the above observations, in this paper, we introduce the structure information of the image into consideration to further reduce the required query counts when modeling the distribution of adversarial examples. To be specific, the structure is characterized by a Neural Process (NP) [10], an efficient auto-encoder method to model a distribution over regression functions with a deterministic variable focusing on local information and a latent variable focusing on global information. Based on NP, we then propose a Neural Process based black-box attack, named NP-Attack. NP can be pre-trained on benign examples, and then used to reconstruct adversarial examples through optimizing above mentioned variables, which is definitely efficient than previous pixel-wise operations in $\mathcal{N}$Attack. As a proof-of-concept, we conduct an experiment to compare these two distribution-based attacks: $\mathcal{N}$Attack and NP-Attack. Both attacks are conducted by limiting the same maximal query count, and the maximal distortion of L_∞ is set to 0.05 for CIFAR10 and 0.2 for MNIST. The percentage of adversarial examples returned at the maximal query count is plotted in Figure 1. We can see that the adversarial distribution optimized in NP-Attack (three variants) contains a higher percentage of adversarial examples under the same query counts compared to $\mathcal{N}$Attack, which illustrates NP-Attack is more efficient in optimizing and modeling the adversarial distributions. Figure 1 also gives some
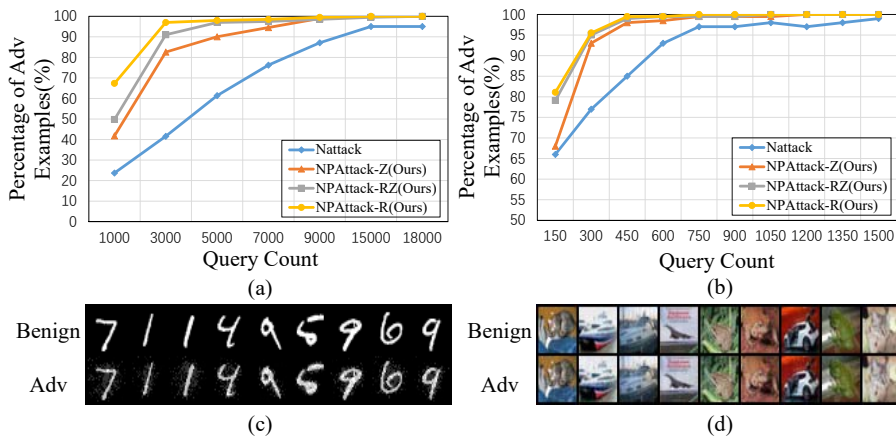
Fig. 1. Comparison of two distribution-based black-box attacks: $\mathcal{N}$Attack and our proposed NP-Attack. Percentage of adversarial examples with different query counts generated by $\mathcal{N}$Attack and NP-Attack on MNIST (a) and CIFAR10 (b), and the corresponding NP-Attack generated adversarial examples and benign examples of MNIST (c) and CIFAR10 (d). It is demonstrated that NP-Attack is more query efficient compared to $\mathcal{N}$Attack when generating adversarial examples.

examples of adversarial examples generated in our NP-Attack, which are also visual integrity. Our main contributions could be summarized as follows:

– We propose a distribution based black-box attack, Neural Process based black-box attack (NP-Attack), which uses the image structure information for modeling adversarial distributions and reduces the required query counts.
– NP-Attack has several optimization variants due to the variables in NP. The optimization on deterministic variable focuses more on the local information, while optimization on latent variable focuses more on the global information. The different optimization variants have different effects on the location of adversarial perturbations, which brings more flexibility for NP-Attack.
– Extensive experiments demonstrate the superiority of our proposed NP-Attack. On both untargeted and targeted attacks, NP-Attack greatly reduces the needed query counts under the same attack success rate and distortion, compared with the state-of-the-art query-based black-box attacks.

2   Related Work

Existing black-box attacks can be categorized into two groups: 1) transferability-based method that transfers from attacking a surrogate model; and 2) query-based method that directly generates adversarial examples on the target model.

For transferability-based black-box attacks, adversarial examples are crafted on a surrogate model then applied to attack the target model. There are several techniques to improve the transferability of black-box attacks. For example,

Momentum Iterative boosting (MI) [8] incorporates a momentum term into the gradient to boost the transferability. Diverse Input (DI) [34] proposes to craft adversarial examples using gradient with respect to the randomly-transformed input example. Skip Gradient Method (SGM) [33] uses more gradients from the skip connections rather than the residual modules via a decay factor to craft adversarial examples with high transferability. However, they usually cannot obtain the 100% attack success rate, which is closely restricted by the dependency between the surrogate model and the target model.

For query-based black-box attacks, they could be further classified as decision-based (query results are one-hot labels) [5] or score-based ones (query results are scores). The score-based attacks are investigated in this paper. This kind of method estimates the gradient of the target model via a large number of queries, which is then used to generate adversarial examples. ZOO [4] explores gradient estimation methods by querying the target model as an oracle. They use zeroth-order stochastic coordinate descent along with dimension reduction, hierarchical attack and importance sampling techniques, to directly estimate the gradients of the targeted model for generating adversarial examples. However, ZOO requires numerous queries to estimate the gradients with respect to all pixels. Further, AutoZOOM [26] operates the gradient estimation in latent space, using an offline pre-trained auto-encoder or a bilinear mapping function to compress the attack dimension. It then applies an adaptive random gradient estimation strategy to balance query counts and distortion, which improves the query efficiency by a great deal. Meta Attack [9] pre-trains a meta attacker model to estimate the black-box gradient, which efficiently reduces the query counts. Beyond zeroth-order optimization-based approaches, QL [13] proposes to use a Natural Evolution Strategy (NES) [31] to enhance query efficiency. Bandits [14] further introduces time and data priors under NES. $\mathcal{N}$Attack [18] is another kind of black-box attack that explicitly models the adversarial distribution with a Gaussian distribution. The adversarial attack is hence formalized as an optimization problem, which searches the Gaussian distribution under the guidance of increasing the attack success rate of target models.

Following the general NES structure, we explore the adversarial distribution in a more efficient way using some high-level information of images in this paper.

3 Proposed Neural Process-based Black-box Attack

In this section, we propose a Neural Process-based Black-box Attack (NP-Attack) with significantly reduced query counts. NP-Attack models the distribution of adversarial examples efficiently by a Neural Process that utilizes the high-level structure information of images rather than the pixel-level information.

3.1 Preliminaries of Neural Process

Neural Process (NP) [10] is a combination of the best from neural networks and Gaussian Process [22, 32], which could efficiently estimate the uncertainty in
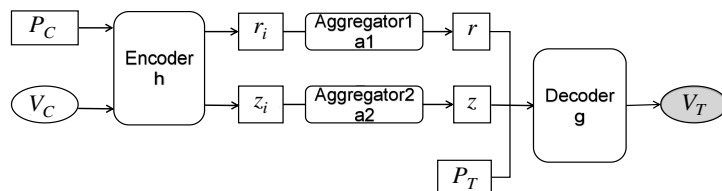
Fig. 2. The structure of NP model trained on pixels of one image. Here, V_C and V_T are the context and target pixel values, then P_C and P_T are the corresponding pixel positions. z_i and r_i are pixel-wise latent variables and deterministic variables.

the predictions. It could be expanded into an attentive version, called Attentive Neural Process (ANP) [15, 27], which is applied in our NP-Attack[§].

As pixel values in one image subject to a Gaussian Process, NP is applied to reconstruct images by predicting their pixel values. As shown in Figure 2, the structure of NP consists of the following three parts: 1) *Encoder h*, whose inputs are context pixel pairs concatenating pixel positions and values; 2) *Aggregator a*, which summarizes the outputs of the encoder as latent variable z from a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ and deterministic variable r; 3) *Decoder g*, which takes the sampled latent variable z, deterministic variable r and target pixel position P_T as inputs and predicts the target pixel value V_T. Overall, NP models the distributions as:

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

Here P, V represents pixel position and value, and C, T are random subsets of context and target, s_C is the distribution modeled over (P_C, V_C). In Eq. 1, Latent variable z accounts for uncertainty in the predictions of V_T for observed (P_C, V_C), while r is calculated by a deterministic function which aggregates (P_C, V_C) into a finite dimensional representation with permutation invariance in C. The interpretation of the latent path is that z gives rise to correlations in the marginal distribution of the target predictions V_T, modeling the global structure of the stochastic process realization, whereas the deterministic path r models the fine-grained local structure.

3.2   Pre-training of Neural Process

The NP model in NP-Attack is pre-trained on the pixels of given benign images by maximizing the following Evidence Lower Bound (ELBO)[¶]:

$$\log p\left(V_T|P_T, P_C, V_C\right) \geq \mathbb{E}_{q(z|s_T)}\left[\log p\left(V_T|P_T, r, z\right)\right] - D_{KL}\left(q\left(z|s_T\right)\|q\left(z|s_C\right)\right). \tag{2}$$

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

[¶] The implementation details of ANP are shown in the Appendix A

Like s_C defined above, s_T represents the distribution modeled over (P_T, V_T). To be specific, the s_C and s_T are both the whole pixels in one image during the pre-training process in NP-Attack. Once trained, NP shares the same encoder and decoder on the same image set even on different selected images, varying on latent variables and deterministic variables corresponding to different images. Note that such pre-training process is independent from the main attack part on the benign examples, which means the encoder and decoder are fixed after pre-training and could keep some structure information of images.

The pre-trained NP models a distribution by variables z and r, where z is sampled from a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, thus the latent variables and deterministic variable are in fact (μ, σ) and r. As NP models the pixel-wise distribution in one image, its structure information is kept in such NP. Adversarial examples and benign examples are imperceptible to human eyes, sharing a similar visual structure. Thus 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.

3.3   Overview of the Proposed NP-Attack

Equipped with the pre-trained NP, we can propose the distribution-based NP-Attack. The key idea of distribution-based attack is to model the adversarial distribution around the small region of one natural example x, such that a sample drawn from this distribution is likely an adversarial example. Compared to previous distribution-based $\mathcal{N}$Attack, the difference lies on the method of modeling the adversarial distribution. $\mathcal{N}$Attack models such adversarial distribution just on pixel level with a Gaussian distribution focusing on pixels independently, not considering any other information on the structure of pixel values in one image. However, our proposed NP-Attack utilizes the decoder of a pre-trained NP model and only optimizes latent variables or deterministic variables, in which case the fixed decoder could hold some structure information of pixels in one image, improving the optimization efficiency in modeling adversarial distributions.

Specifically, the objective function L in our NP-Attack is defined on S, an intersection of a latent region modeled by NP and a L_p-ball centered around the benign example x, $i.e.$, $S = S_p(x) \bigcap \mathrm{NP}(x)$. Given l as the loss defined on a given example point, the objective L with regard to adversarial distribution in our optimization criterion is:

$$L((\mu, \sigma), r|x) := \int_{x-\epsilon}^{x+\epsilon} l(x_{rec}) g(x_{rec}|(\mu, \sigma), r) h((\mu, \sigma), r|x) dx_{rec} \qquad (3)$$

where g and h are the decoder and encoder of pre-trained NP, μ, σ, r denote for the variables to be optimized in NP-Attack, x_{rec} denotes the image reconstructed by NP, ϵ denotes the L_p ball restriction. By optimizing on such objective L, we could achieve our aim to model the latent manifold of adversarial examples.

As the target model could return scores in each query, the original loss function is defined as:

$$l(x) := \begin{cases} \max(0, \max\limits_{c \neq y} \log F(x)_c - \log F(x)_y), & \text{targeted,} \\ \max(0, \log F(x)_y - \max\limits_{c \neq y} \log F(x)_c), & \text{untargeted.} \end{cases} \tag{4}$$

where $F(x)$ 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.

In summary, the procedures on modeling and sampling from distribution in NP-Attack are shown in Algorithm 1: 1) Feed a benign example x, and compute r and (μ, σ) from the *Encoder h* of NP; 2) Sample z from $\mathcal{N}(\mu, \sigma^2)$; and 3) Reconstruct x_{rec} with the optimized (r, z) from the *Decoder g* of NP (shown in the following Section 3.4), then project x_{rec} back into the L_p-ball centered at x.

3.4   Optimization of NP-Attack

As the reconstructed image x_{rec} is mainly dependent on the sampled r, z and pre-trained *Decoder g* of NP, there are three optimization options: NP-Attack-R/Z/RZ corresponding to the optimized variables r, z or both. As the three branches share the similar optimization function, we take **NP-Attack-R** as an example here, and other optimization options are shown in the Appendix B.

In this case, r is a variable in NP, cooperating with z to reconstruct an image. The difference of r from z is that r is a deterministic variable. That is, z shares the same distribution in one image for all pixels, while r is independent for different pixels thus with better freedom. As adversarial perturbations are computed and added on pixels, the optimization on r might be more reasonable. The benefit of optimizing r than $\mathcal{N}$Attack is that r is in the latent space with a larger latent dimension (128 dimension in our NP-Attack) for each pixel, which owns more capacity to perturb.

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

$$r_{t+1} \leftarrow r_t - \frac{\eta}{b} \sum_{i=1}^{b} l(g(R_i)) \nabla_{r_t} \log \mathcal{N}(R_i \mid r_t, \sigma'^2), \tag{5}$$

where b is the batch size. Given that $R_i = r_t + p_i \sigma'$, where p_i is sampled from the standard Gaussian distribution $\mathcal{N}(0, I)$, and σ' is a hyper-parameter, not essentially equal to σ in z, $\nabla_{r_t} \log \mathcal{N}(R_i \mid r_t, \sigma'^2) \propto \sigma'^{-1} p_i$.

3.5   Discussion

As the adversarial examples in our NP-Attack is sampled and reconstructed through the decoder of the benign pre-trained NP, we study on the examples

---

Algorithm 1 NP-Attack

Input: natural image 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

1: Compute the variables from *Encoder h* on image x: $\mathcal{N}(\mu, \sigma^2), r \leftarrow h(x)$

2: **for** $t = 0$ **to** $T - 1$ **do**

3:    **if** F(x) $\neq$ y (untargeted) or F(x) = y (targeted) **then**

4:      attack success.

5:    **else**

6:      Sample perturbations from Gaussian distribution: $p_i \sim \mathcal{N}(0, I), i = 1, 2, ..., b$.

7:      Add the perturbation p_i in NP-Attack-R/Z/RZ, specifically on μ or r or both,

$$\begin{cases} z_i \sim \mathcal{N}(\mu, \sigma^2), r_i = r + p_i\sigma, & \text{NP-Attack-R,} \\ z_i \sim \mathcal{N}(\mu + p_i\sigma, \sigma^2), r_i = r, & \text{NP-Attack-Z,} \\ z_i \sim \mathcal{N}(\mu + p_i\sigma, \sigma^2), r_i = r + p_i\sigma, & \text{NP-Attack-RZ.} \end{cases}$$

8:      Reconstruct the image from *Decoder g*, and use project function P to restrict its maximal distortion: $x_i = P(g(z_i, r_i))$.

9:      Compute the losses of these reconstructed image series x_i under targeted or untargeted setting,

$$l_i = \begin{cases} \max(0, \max\limits_{c \neq y} \log F(x_i)_c - \log F(x_i)_y), & \text{targeted,} \\ \max(0, \log F(x_i)_y - \max\limits_{c \neq y} \log F(x_i)_c), & \text{untargeted.} \end{cases}$$

The corresponding loss $l_i = l_i - mean(l)$.

10:      Update μ or r or both as optimization:

$$\begin{cases} r_{t+1} \leftarrow r_t - \dfrac{\eta}{b\sigma} \sum\limits_{i=1}^{b} l_i p_i, & \text{NP-Attack-R,} \\\\ \mu_{t+1} \leftarrow \mu_t - \dfrac{\eta}{b\sigma} \sum\limits_{i=1}^{b} l_i p_i, & \text{NP-Attack-Z,} \\\\ \mu_{t+1} \leftarrow \mu_t - \dfrac{\eta}{b\sigma} \sum\limits_{i=1}^{b} l_i p_i, r_{t+1} \leftarrow r_t - \dfrac{\eta}{b\sigma} \sum\limits_{i=1}^{b} l_i p_i, & \text{NP-Attack-RZ.} \end{cases}$$

11:    **end if**

12: **end for**

---

reconstructed by NP. Figure 3 shows that the adversarial examples and benign ones could both be reconstructed by benign pre-trained NP, sharing the same decoder, optimized with different latent variables. However, noised examples (added with slight Gaussian noises) could absolutely not be reconstructed by such NP. This phenomenon contributes to the query efficiency of NP-Attack compared to $\mathcal{N}$Attack, as the appliance of NP filters the noised examples away, the distributions modeled by NP are with higher percentages of adversarial examples, which keep consistency with the Figure 1.

Moreover, we discuss the difference between NP and other auto-encoder models, like VAE [16]. Note that the variables in NP are different from VAE. In VAE, the latent variable z is on the whole image set, which means a different sample would refer to the reconstruction of a different image. In NP, the distribution

(a) Noise examples with random Gaussian noise
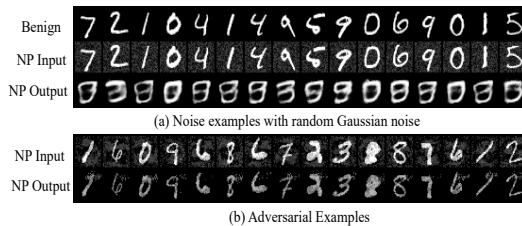
(b) Adversarial Examples

Fig. 3. 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.

of z is modeled independently on one image. So the optimization of z in VAE to model adversarial distribution could lead to a collapse on distributions with high possibility. On the contrary, the Gaussian distribution of z in NP could be optimized independently on different images towards adversarial distribution. Moreover, NP has another variable r, which is also modeled on one image and different from z, bringing more flexibility.

4   Experiment

In this section, we evaluate our method on three benchmark image classification datasets MNIST [20], CIFAR10 [17] and ImageNet [6]. 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, CIFAR10 and ImageNet to show the superiority of our method in reducing query counts.

Experimental Setups For MNIST, a MLP model is trained with three fully connected layers, achieving 98.50% accuracy. For CIFAR10, the WideRes-Net [35] is adopted, achieving 94.10% accuracy. For ImageNet, we use the pre-trained Inception-V3 model [25], achieving 78% accuracy. The baseline query based black-box techniques are ZOO [4], AutoZOOM [26], QL [13] and $\mathcal{N}$Attack [18]. The results of those compared methods are reproduced by the code released in the original paper with default settings.

NP Pre-training For NP-Attack, NP models are pre-trained on the train set of MNIST, CIFAR10 and ImageNet respectively which takes around 5, 45 and 72 hours on a single NVIDIA GTX 1080 TI GPU. The dimensions of r and z are both 128. Regarding the computational overhead of the pre-training, we would like to point out that the pre-training of NP is over the whole dataset while the black-box attack is performed on a single image per time. The pre-trained model is used for free to perform black-box attack, when the pre-training is complete.

Thus, the pre-training is a off-line operation, which is totally separated from the on-line black-box attack. The overhead of pre-training will not affect the black-box attack phase.

4.1   Empirical Understanding of NP-Attack

We evaluate the performance of NP-Attack under different experimental settings, including various sample sizes, maximum distortion, and optimization methods (optimizing r, z or both). Experiments are conducted on MNIST with 200 correctly classified images selected from the test set.

   Optimization Method We first investigate the performance of our NP-Attack with different optimization methods, $i.e.$, optimizing over the variable r, z or both. The experimental results are summarized in Table 1. The L_∞ distortion is set to 0.2 for MNIST and 0.05 for CIFAR10. Optimizing r is more query efficient, for the reason that r is modeled independent for different pixels in NP while z is modeled as a same Gaussian distribution over all pixels in one image. NP-Attack-R and NP-Attack-Z show the upper and lower bound of our NP based attack. The adversarial examples generated by different optimization methods are shown in Appendix C.

Table 1. Evaluation of NP-Attack under untargeted setting by optimizing the variables r, z or both on 200 correctly classified images from MNIST and CIFAR-10.

Attack Method	MNIST			CIFAR10		
	ASR	L_2 Dist	Query Count	ASR	L_2 Dist	Query Count
NP-Attack-R	100%	3.07	**1,190**	100%	1.75	**96**
NP-Attack-Z	100%	3.55	1,665	100%	1.68	150
NP-Attack-RZ	100%	3.65	1,460	100%	1.80	98

   Sample Size We test the sample size $b \in \{10, 20, 30, 40, 50\}$ and set the L_∞ distortion to 0.2. 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. The changes of ASR with the maximal query count are plotted in Figure 4(a). It shows that our NP-Attack could achieve over 90% ASR with different sample sizes when the maximal query count is larger than 7000. The ASR, L_2 distortion and average query count when the maximal query count is 7000 are reported in Table 2. We can see that, with a larger sample size, the iteration T is reduced along with larger average query counts. However, we can get higher ASRs and smaller L_2 distortion by increasing the sample size, illustrating a more accurate estimation of the perturbation direction.

   Maximal Distortion It is a trade-off between the distortion and query counts in black-box attack. We set the sample size $b = 10$ and test the L_∞ distortion $\epsilon \in \{0.10, 0.15, 0.20, 0.25, 0.30\}$. The change of ASRs with various
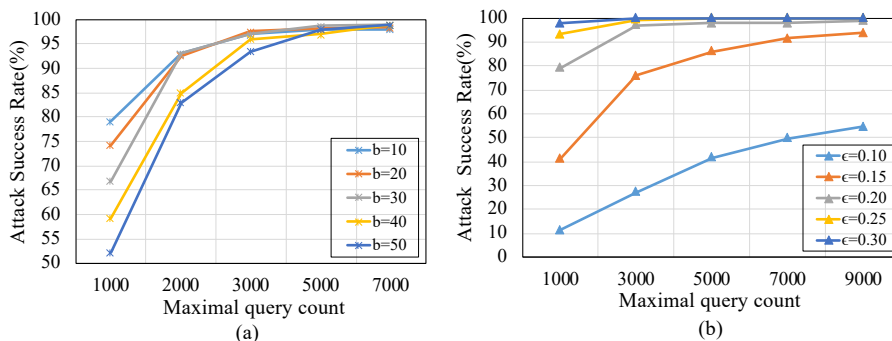
Fig. 4. Change of ASRs when limiting the query counts on MNIST with NP-Attack-R. (a) Different curves represent performance under different sample sizes. (b) Different curves represent restricting different maximal distortion.

Table 2. Evaluation of NP-Attack-R under untargeted setting on MNIST with different sample sizes. The maximal query count is 7000 and L_∞ distortion is restricted to 0.2.

Sample Size (b)	Iteration Nums (T)	ASR	L_2 Dist	Avg Query Count
10	700	98.01%	4.3707	**723**
20	350	98.51%	3.9478	786
30	233	**99.00%**	3.6675	940
40	175	**99.00%**	3.4675	1,074
50	140	**99.00%**	**3.3277**	1,203

maximal query counts is shown in Figure 4(b). When $\epsilon = 0.3$, our NP-Attack achieves 98% ASR with only 1000 maximal query counts. With the 5000 query counts, NP-Attack can achieve nearly 100% ASR when L_∞ is larger than 0.2.

4.2   Evaluation on MNIST and CIFAR10

For both MNIST and CIFAR10, we randomly select 1000 correctly classified images from the test set for untargeted attack. For targeted attack, 100 correctly classified images are selected from test set, for each image the target labels are set to the other 9 classes and a total 900 attacks are performed. In our experiments, the L_∞ distortion is restricted to 0.2 for MNIST and 0.05 for CIFAR10 following the setting of [14]. For untargeted attack, the maximal iteration is $T = 900$ and sample size is $b = 30$ while for targeted attack $T = 2000$, $b = 50$. The learning rate is $\eta = 0.01$ in default setting. A total query count is obtained by multiplying the number of iterations and the query count per iteration.

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.

Table 3. Adversarial evaluation of black-box attacks on MNIST.

Attack Method	Untargeted Attack				Targeted Attack			
	ASR	L_2	L_∞	Query Count	ASR	L_2	L_∞	Query Count
ZOO	100%	1.12	0.21	107,264	100%	1.64	0.29	128,768
AutoZOOM-BiLIN	100%	2.01	0.50	9,129	99.89%	2.78	0.62	9,401
AutoZOOM-AE	100%	2.62	0.67	10,202	99.89%	3.74	0.87	10,380
QL	96.62%	3.38	0.20	2,549	99.67%	3.09	0.20	2,693
$\mathcal{N}$Attack	95.09%	2.14	0.20	4,357	98.22%	3.33	0.20	5,981
NP-Attack-R(Ours)	100%	3.09	0.20	**1,226**	100%	3.72	0.20	2,693
NP-Attack-Z(Ours)	99.90%	3.55	0.20	1,680	100%	3.94	0.20	**2,605**

Table 4. Adversarial evaluation of black-box attacks on CIFAR10.

Attack Method	Untargeted Attack				Targeted Attack			
	ASR	L_2	L_∞	Query Count	ASR	L_2	L_∞	Query Count
ZOO	100%	0.12	0.02	208,384	99.52%	0.19	0.02	230,912
AutoZOOM-BiLIN	100%	1.56	0.15	8,113	100%	2.13	0.21	8,266
AutoZOOM-AE	100%	1.88	0.16	7,113	100%	2.78	0.24	8,217
QL	98.40%	1.91	0.05	857	99.55%	2.11	0.05	616
$\mathcal{N}$Attack	99.89%	2.61	0.05	183	100%	2.61	0.05	1,151
NP-Attack-R(Ours)	100%	1.74	0.05	**94**	100%	1.85	0.05	**589**
NP-Attack-Z(Ours)	100%	1.67	0.05	144	100%	1.78	0.05	936

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 with 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 which quickly attack the model successfully and generate adversarial examples with quite 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.

The experimental results are shown in Tables 3 and 4. We report the ASRs, L_2 distortion, L_∞ distortion and average query counts. As mentioned in Section 4.1, performance of NP-Attack-RZ is between NP-Attack-R and NP-Attack-Z. Thus, in the following experiments, we only show the results of NP-Attack-R and NP-Attack-Z to benchmark the upper and lower bound of our NP based attack. Compared to ZOO, our NP-Attack-R can reduce the query count by 98.86% and 99.95% on MNIST and CIFAR10 respectively under untargeted attack setting, while for targeted attack, NP-Attack-R reduces the query count by 96.91% and 99.74% on MNIST and CIFAR10. 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 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_∞ distortion. QL utilizes NES algorithm to estimate the gradient, thus the performance of such technique hinges on the quality of the estimated gradient. QL shows its superiority in targeted attack 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 QL, our NP-Attack-R could achieve much better results in untargeted attack and comparable results in targeted attack. 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, NP-Attack obtains the best trade-off between query counts and distortion. More experimental results on various architectures such as ResNet18 [12] and VGG16 [24] are reported in Appendix D.

4.3   Evaluation 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 true one out of 1000 classes is selected to serve as the target. The L_∞ distortion is restricted to 0.05 following the setting of [13, 14], $T = 600$, $b = 100$ and the learning rate $\eta = 0.005$. 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. To be specific, we resize the images to $32 \times 32 \times 3$ to train the NP model, then we add perturbations on each $32 \times 32 \times 3$ patch, which is cut independently from the original images. Although using the attack dimension reduction, our NP-Attack method still outperforms.

   The experimental results are summarized in Table 5. Due to the large image sizes, ZOO suffers from low ASRs and tremendous model evaluations especially for targeted attack. For other three compared attacks, they achieve 100% ASRs at the cost of over 10,000 queries in targeted attack and over 1,000 queries in untargeted attack. While for our NP-Attack, NP-Attack-R can achieve 100% ASR with only 867 queries, yielding a 94.45% query reduction ratio compared to ZOO in untargeted attack. In targeted attack, NP-Attack-R and NP-Attack-Z get over 98% ASRs and reduce the query counts to 8,001 and 11,383 respectively,

which exceed all the compared methods. This demonstrates that our NP-Attack can scale to ImageNet set. The generated adversarial examples by NP-Attack-R are shown in Appendix E.

Table 5. Adversarial evaluation of black-box attacks on ImageNet.

Attack Method	Untargeted Attack			Targeted Attack		
	ASR	L_2 Dist	Query Count	ASR	L_2 Dist	Query Count
ZOO	90%	1.20	15,631	78%	3.43	2.11×10^6
AutoZOOM-BiLIN	100%	9.34	3,024	100%	11.26	14,228
QL	100%	17.72	3,985	100%	17.39	33,360
$\mathcal{N}$Attack	100%	24.01	2,075	100%	24.14	14,229
Bandits	100%	–	1,165	100%	–	25,341
NP-Attack-R(Ours)	100%	10.96	**867**	98.02%	14.38	**8,001**
NP-Attack-Z(Ours)	96.04%	12.37	1,236	98.02%	14.60	11,383

5   Conclusions

In this paper, we focus on improving the query efficiency in black-box attack by modeling the high-level distribution of adversarial examples. By considering the structure information of pixels in one image rather than individual pixels, which is realized by a decoder of benign pre-trained Neural Process model, we propose the Neural Process-based black-box attack (NP-Attack) to greatly reduce the required query complexity. Evaluated on MNIST, CIFAR10 and ImageNet with regard to query count and attack success rate, our proposed NP-Attack achieves the state-of-the-art results, showing its efficiency and superiority under the black-box adversarial attack setting. Moreover, when pre-training the NP model on adversarial image examples instead of benign examples, we believe the query efficiency could be further improved, which is left for our future work.

6   Acknowledgement

References

1. Bai, J., Chen, B., Li, Y., Wu, D., Guo, W., Xia, S.T., Yang, E.h.: Targeted attack for deep hashing based retrieval. In: ECCV (2020)
2. Bai, Y., Feng, Y., Wang, Y., Dai, T., Xia, S.T., Jiang, Y.: Hilbert-based generative defense for adversarial examples. In: ICCV (2019)
3. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: S&P (2017)
4. Chen, P.Y., Zhang, H., Sharma, Y., Yi, J., Hsieh, C.J.: Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. arXiv preprint arXiv:1708.03999 (2017)
5. Chen, W., Zhang, Z., Hu, X., Wu, B.: Boosting decision-based black-box adversarial attacks with random sign flip. In: ECCV (2020)
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)
8. Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., Li, J.: Boosting adversarial attacks with momentum. In: CVPR (2018)
9. Du, J., Zhang, H., Zhou, J.T., Yang, Y., Feng, J.: Query-efficient meta attack to deep neural networks. In: ICLR (2020)
10. Garnelo, M., Schwarz, J., Dan, R., Viola, F., Teh, Y.W.: Neural processes. In: ICLR (2018)
11. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: ICLR (2015)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
13. Ilyas, A., Engstrom, L., Athalye, A., Lin, J.: Black-box adversarial attacks with limited queries and information. In: ICML (2018)
14. Ilyas, A., Engstrom, L., Madry, A.: Prior convictions: Black-box adversarial attacks with bandits and priors. arXiv preprint arXiv:1807.07978 (2018)
15. Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., Teh, Y.W.: Attentive neural processes. In: ICLR (2019)
16. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: ICLR (2014)
17. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Technical Report, University of Toronto (2009)
18. Li, Y., Li, L., Wang, L., Zhang, T., Gong, B.: Nattack: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. In: ICML (2019)
19. Liu, X., Bai, Y., Xia, S.T., Jiang, Y.: Self-adaptive feature fool. In: ICASSP (2020)
20. Lécun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
21. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. In: ICLR (2018)
22. Matthews, A., Rowland, M., Hron, J., Turner, R., Ghahramani, Z.: Gaussian process behaviour in wide deep neural networks. In: ICLR (2018)
23. Moosavi-Dezfooli, S.M., Fawzi, A., Fawzi, O., Frossard, P.: Universal adversarial perturbations. In: CVPR (2017)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Computer Science (2014)

25. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR (2016)
26. Tu, C.C., Ting, P., Chen, P.Y., Liu, S., Zhang, H., Yi, J., Hsieh, C.J., Cheng, S.M.: Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. In: AAAI (2019)
27. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
28. Wang, Y., Deng, X., Pu, S., Huang, Z.: Residual convolutional ctc networks for automatic speech recognition. arXiv preprint arXiv:1702.07793 (2017)
29. Wang, Y., Ma, X., Bailey, J., Yi, J., Zhou, B., Gu, Q.: On the convergence and robustness of adversarial training. In: ICML (2019)
30. Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., Gu, Q.: Improving adversarial robustness requires revisiting misclassified examples. In: ICLR (2020)
31. Wierstra, D., Schaul, T., Peters, J., Schmidhuber, J.: Natural evolution strategies. In: CEC (2008)
32. Wistuba, M., Schilling, N., Schmidt-Thieme, L.: Scalable gaussian process-based transfer surrogates for hyperparameter optimization. Machine Learning **107**(1), 43–78 (2018)
33. Wu, D., Wang, Y., Xia, S.T., Bailey, J., Ma, X.: Skip connections matter: On the transferability of adversarial examples generated with resnets. In: ICLR (2020)
34. Xie, C., Zhang, Z., Zhou, Y., Bai, S., Wang, J., Ren, Z., Yuille, A.L.: Improving transferability of adversarial examples with input diversity. In: CVPR (2019)
35. Zagoruyko, S., Komodakis, N.: Wide residual networks. arXiv preprint arXiv:1605.07146 (2016)