QE-DBA: Query-Efficient Decision-Based Adversarial Attacks via Bayesian Optimization

Zhuosheng Zhang
Stevens Institute of Technology
zzhang97@stevens.edu

Noor Ahmed
Air Force Research Laboratory
nosidahmed@gmail.com

Shucheng Yu
Stevens Institute of Technology
syu19@stevens.edu

***Abstract*—With the widespread popularity of mobile internet, an increasing number of IoT devices can use cloud services to invoke deep learning to accomplish computer vision tasks. Decision-based attacks (DBA), wherein attackers perturb inputs to spoof learning algorithms by observing solely the output labels, are a type of severe adversarial attacks against Deep Neural Networks (DNNs) that require minimal knowledge of attackers. Most existing DBA attacks rely on zeroth-order gradient estimation and require an excessive number (>20,000) of queries to converge. To better understand the attack, this paper presents an efficient DBA attack technique, namely QE-DBA, that greatly improves the query efficiency. We achieve this by introducing dimension reduction techniques and derivative-free optimization to the process of closest decision boundary search. In QE-DBA, we adopt the Gaussian process to model the distribution of decision boundary radius over a low-dimensional search space defined by perturbation generator functions. Bayesian Optimization is then leveraged to find the optimal direction. Experimental results on pre-trained ImageNet classifiers show that QE-DBA converges within 200 queries while the state-of-the-art DBA techniques using zeroth-order optimization need over 15,000 queries to achieve the same level of perturbation distortion.**

***Index Terms*—Adversarial Attack, Bayesian Optimization, Image Classification, Internet of Things**

I. Introduction

Recent advances in computation and learning have made deep neural networks (DNNs) an important enabler for the Internet of Things. However, DNNs have also shown vulnerabilities to *adversarial examples* - a type of maliciously perturbed examples that are almost identical to original samples in human perception but can cause models to make incorrect decisions ([1]). Such vulnerabilities can lead to severe and sometimes fatal consequences in many real-world DNN applications such as autonomous vehicles, financial services, and robotics. Therefore, it is critical to understand limitations of current learning algorithms and identify their vulnerabilities, which in turn helps to improve the robustness of learning.

According to the knowledge available to the attacker, adversarial attacks can be categorized into three primary types: *white-box attacks*, *score-based attacks*, and *decision-based attacks*. In white-box attacks *e.g.* [2] and [3], the attacker requires complete knowledge of the architecture and parameters of the target network. The latter two are black-box attacks in which the attacker only observes inputs and outputs without knowing internal architecture and parameters. In score-based attacks (SBA), the attacker can only access the soft-label output (real-valued probability scores) of the target model for a given input. In decision-based attacks (DBA), a.k.a. *hard-label attacks*, only the hard label of a given input is available.

Of the three attacks, DBA can lead to severe and ubiquitous threats to practical systems because of the minimal required knowledge on the victim model, and hence has attracted great interests recently. However, DBA is also the most challenging adversarial attack to design because of the relative insensitivity of model outputs to input perturbation - it is often difficult for the attacker to determine whether the change of perturbation is preferred or not when the target model's prediction does not change. To make the attack stealthy, a DBA attacker shall discover the decision boundary where a slight change of perturbation will cause the model to yield different prediction labels. In reality, cloud-based machine learning platforms often limit the query frequency to thwart malicious queries. For example, the Google cloud vision API only allows 1,800 requests per minute. Therefore, improving query efficiency is critical for successful DBA attacks in practical systems. While current research is mostly focused on improving the efficiency of zeroth-order gradient estimation ([4]–[7]), the overall query complexity of DBA attacks remains impractically high – it usually takes tens of thousands of queries to estimate the gradient at each iteration of optimization.

In this work, we propose an efficient decision-based black-box attack, namely QE-DBA, that searches adversarial examples via statistical information of the decision boundary radius. Specifically, to reduce the search space, we introduce the perturbation generation function to map the decision boundary radius to a low-dimensional subspace. Instead of searching the adversarial perturbation in the image-scale dimension directly, we model the distribution of the decision boundary radius on a reduced dimension using the Gaussian Process and search the most possible low dimensional input that yields the optimal perturbation using Gaussian optimization (BO). Experimental results show that our method reduces the query complexity by up to *one order of magnitude* as compared to the state of the art including RayS.

The rest of the paper is organized as follows: Section II reviews adversarial attacks and technical preliminaries. Section III describes our technical intuition and Section IV elaborates the design of QE-DBA. Experimental results are provided in Section V. We conclude the paper in Section VI.

II. Related work

A. Decision-based Adversial attacks

Decision-based Adversial attack (DBA) are one type of adversarial attacks against learning systems where the attacker has no knowledge about the model and can only observe inputs and their corresponding output labels by querying the model.

DBA attacks are detrimental to learning systems because of the minimal requirements on the knowledge of attackers. There have been several DBA techniques in the literature. In [8], a perturbed example is generated starting with a large perturbation sampled from a proposed distribution. It then iteratively reduces the distance of the perturbed example to the original input through a random walk along the decision boundary. [9] first proposed a formulation which turns DBA problem into the problem of finding the optimal direction that leads to the shortest l_2 distance to decision boundary and optimized the new problem via zeroth-order optimization methods. [5] and [4] furthermore improve [9]'s query efficiency via estimating the sign of gradient or optimizing the hyperparameters of optimizing procedural. Recently, [7] further reduce the query complexity of zeroth-order gradient estimation by projecting the input space into a low dimensional subspace. However, this work still relies on gradient estimation and does not support arbitrary projection methods that may violate the smoothness of the optimization function.

B. Bayesian Optimization

Bayesian optimization (BO) is a sequential optimization method particularly suitable for problems with low dimensions and expensive query budgets ([10]). It contains two main components - a *probabilistic surrogate model*, usually a Gaussian Process (GP), for approximating the objective function, and an *acquisition function* that assigns a value to each query that describes how optimal the query is.

Gaussian Process is a statistic surrogate that induces a posterior distribution over the objective functions ([11]). In particular, a Gaussian Process $\mathcal{GP}(\mu_0, \Sigma_0)$ can be described by a prior mean function μ_0 and positive-definite kernel or covariance function Σ_0. In this paper, we adapt the Matern $5/2$ Kernel ([12]) as the covariance function, which is defined as:

$$\Sigma(x, x') = (1 + \frac{\sqrt{5}r}{l} + \frac{5r^2}{3l^2})exp(-\frac{\sqrt{5}r}{l})$$

where $r = x - x'$ and l is the length-scale parameter ([13]).

Acquisition Function in Bayesian optimization is a function that evaluates the utility of model querying at each point, given the surrogate model, to find the optimal candidate query point for the next iteration ([14]). *Expected Improvement* (EI) and *Upper Confidence Bound* (UCB) are the two most popular acquisition functions that have been shown effective in real black-box optimization problems ([12]). In black-box adversarial attacks, most studies like [15] and [16] adopted EI as the acquisition function because of its better convergence performance ([12], [13]). In this paper, we also use EI as the acquisition function which is defined as:

$$EI_n(x) = \mathbb{E}_n[max(h(x) - h_n^*, 0)]$$

where h is the objective function and h_n^* is the best-observed value so far. $\mathbb{E}_n[\cdot] = \mathbb{E}_n[\cdot|D_{1:n-1}]$ denotes the expectation taken over the posterior distribution given evaluations of h at $x_1, \cdots, x_{n-1}$.

III. Technical intuition

In this section, we provide the technical intuition of our QE-DBA attack. We first take an overview of problem formulation and analyze the limitations of existing gradient-based methods. Then we describe the motivation for our design.

A. Formulation of Decision-based Attack

The classification model F takes images as inputs and outputs a $K-$dimensional probability vector which represents confidence scores over the K classes (without loss of generality, we will take images as examples in the rest of this paper). In decision-based setting, we can define a Boolean-valued objective function $h_b : [0, 1]^d \to \{-1, 1\}$ as following:

$$h_b(\gamma) = \begin{cases} sign\{\max_{i \neq y}[F_i(x+\gamma)] - F_y(x+\gamma)\} & \text{(Untargeted)} \\ sign\{F_k(x+\gamma) - \max_{i \neq k}[F_i(x+\gamma)]\} & \text{(Targeted)} \end{cases} \tag{1}$$

where $x \in \mathbb{R}^d$ is the targeted data sample, $y \in \{1...K\}$ is its true label and F_i means $i-th$ element of probability vector. $\gamma \in \mathbb{R}^d$ is the perturbation added to the input data. $k \in \{1...K\}$ represents the target label. Notes that the output of F is unavailable for decision-based attacker. So the objective function h_b can be considered as a black-box function:

$$h_b(\gamma) = \begin{cases} 1 & \text{Attack success} \\ -1 & \text{Attack failed} \end{cases} \tag{2}$$

Obviously, directly maximizing h_b is very difficult because h_b is neither continuous nor differentiable. To overcome this problem, [9] formulate the decision-based attack problem as:

$$\min_{\theta} g(\theta) \text{ where } g(\theta) = \arg\min_{\Delta > 0} \Big(h_b(x_0 + \Delta\theta) = 1\Big) \tag{3}$$

$g(\theta)$ represents the decision boundary radius from input x along the ray direction θ. Then the DBA attack problem can be converted to finding the ray direction with minimum decision boundary radius regarding the original example x. While most of existing works focus on how to solve the formula by estimating the gradient through zeroth-order optimization, only being able to access the decision in DBA makes solving (3) inefficient in terms of query complexity. Specifically, the decision boundary radius $g(\theta)$ is typically estimated by a binary search procedure, and approximation of the gradient of $g(\theta)$ via finite difference requires multiple rounds of computation of $g(\theta)$. RayS, on the other hand, adopts hierarchical searching to solve the formulation in a gradient-free fashion. However, straightforward searching will discard statistical information that can be utilized in optimization methods. Moreover, RayS conducts the searching process directly on the input space which introduces significant query complexity especially when the input dimension is large (e.g. color images).

B. Technical Intuition

In this work, we design an efficient DBA attack based on the statistical properties of the decision boundary and a reduced input search space. To this end, we define the perturbation generator as a function $S : \mathbb{R}^{d'} \to \mathbb{R}^d$ that takes low dimensional inputs $\delta' \in \mathbb{R}^{d'} (d' << d)$ and outputs an image-size perturbation $\delta \in \mathbb{R}^d$. Then we can formulate our objective function $g'(\delta')$ as:

$$\min_{\delta'} g'(\delta') \text{ where } g'(\delta') = \arg\min_{\Delta > 0} \left(h_b(x_0 + \Delta \frac{S(\delta')}{|S(\delta')|}) = 1 \right) \tag{4}$$

In this formulation, we define the search direction in (3) using normalized perturbation generated by S: $\theta = \frac{S(\delta')}{|S(\delta')|}$. The value of g' can be evaluated via multiple decision-based queries which we will discuss in section IV. Note that, although prior work [7] has also adapted a projection matrix to reduce the searching space in decision-based attack, they still aim to solve problem (3) by approximating gradient $g'(\theta)$. This requires the projection matrix to preserve smoothness of the objective function. On the other hand, our method adopts derivative-free optimization which allows us to support more complex or non-differentiable perturbation generation functions to gain better control over searching space selection. For example, the Perlin noise generator will reduce the input space from all possible images into images of Perlin noise.

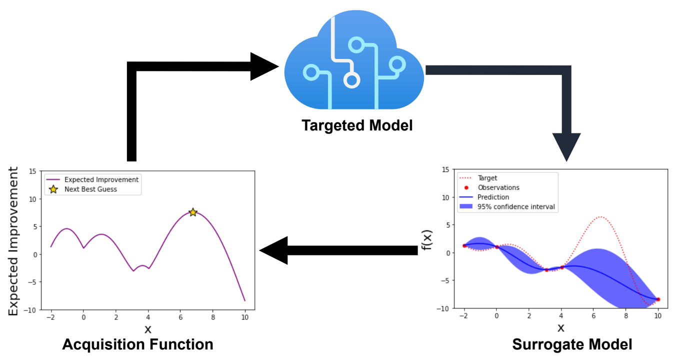

Fig. 1: Workflow of Bayesian Optimization in QE-DBA

With objective function $g'(\delta')$ available, the optimization problem (4) is solvable using Bayesian Optimization. Adopting the logic in Figure (1), the attacker will query the boundary radius $g'(\delta')$ on the searching direction θ generated by the low dimensional input δ'. The optimizer models the distribution of decision boundary radius over the input space and acquires the next most possible optimal input for querying until an adversarial example near enough is found. In particular, for each iteration t, based on observation set $\{\delta'_i, g'(\delta'_i)\}_{i=1}^{t-1}$, we use Gaussian process to model the radius distribution of all possible directions. Then we use Acquisition Function to select δ'_t with the highest probability to generate the lowest radius (smallest perturbation) according to the statistic distribution. Then we query the model to compute $g'(\delta'_t)$ and add the result $\{\delta'_t, g'(\delta'_t)\}$ to the observation set.

Compared to state-of-the-art methods that also adopt BO in SBA ([16], [17]) and DBA ([18]), where both of them formulate the attack as a constrained optimization problem to maximizes the probability score (implicitly in DBA) of the incorrect label, our optimization framework focuses on optimizing the boundary distance which can be evaluated via querying the decision-based model solely. Moreover, our algorithm does not rely on predefined distortion constraints like other BO-based attacks, where the attacker needs to define the required boundary distance beforehand to trade the success rate for perturbation quality.

IV. Decision-based Bayesian Optimization Attack

In this section, we describe an optimization framework for finding adversarial instances for a classification model F in detail. First we discuss how to compute $g'(\delta')$ up to certain accuracy using the Boolean-valued function h_b. Then we will solve the optimization problem via Bayesian Optimization and present our full algorithm.

Algorithm 1: Distance Evaluation Algorithm

```
input : Boolean-valued query function $h_b$ of target
        model, original image $x_0$, low dimensional
        input $\delta'$, increase step size $\eta$, stopping
        tolerance $\epsilon$, maximum distance $\Delta_{max}$
output: $g'(\delta')$
$\theta \leftarrow \frac{S(\delta')}{S(\delta')}$;        // Compute the searching direction
// Fine-grained search
if $h_b(x_0 + \eta\theta) = -1$ then
    $v_{low} \leftarrow x_0 + \eta\theta$, $v_{high} \leftarrow x_0 + 2\eta\theta$;
    while $h_b(v_{high}) = -1$ do
        $v_{low} \leftarrow v_{high}, v_{high} \leftarrow v_{high} + \eta\theta$;
        if $|v_{low}| \geq \Delta_{max}$ then
            return $g'(\delta') = \Delta_{max}$;
        end
    end
else
    $v_{low} \leftarrow 0$, $v_{high} \leftarrow x_0 + \eta\theta$;
end
// Binary search between $[v_{low}, v_{high}]$
while $|v_{high} - v_{low} > \epsilon|$ do
    $v_{mid} \leftarrow (v_{high} + v_{low})/2$;
    if $h_b(v_{mid}) = -1$ then
        $v_{high} \leftarrow v_{mid}$;
    else
        $v_{low} \leftarrow v_{mid}$;
    end
end
return $g'(\delta') = |v_{high}|$;
```

A. Distance Evaluation Algorithm

Algorithm 1 elaborates how to evaluate $g'(\delta')$ via queries on Boolean-value function h_b.

First, the attacker computes the search direction locally $\theta = \frac{S(\delta')}{|S(\delta')|}$. For a given low dimensional input δ', attacker first generates an image-size perturbation $S(\delta')$ via the perturbation generator S. Then normalize $S(\delta')$ into a unit vector $\frac{S(\delta')}{|S(\delta')|}$ to represent the search direction θ. It is easy to notice that for any given input δ', there is always a search direction θ that can be computed.

To evaluate the distance from input x_0 to the decision boundary along the direction θ, the attacker performs a fine-grain search and then a binary search. For simplicity, we assume the l_2 distance here, but the same procedure can also be applied to other distance measurements as long as vector operations are well defined in their respective spaces. In the fine-grained search phase, we cumulatively increase the search distance to query the points $\{x_0 + \eta\theta, x_0 + 2\eta\theta, \dots\}$ one by one until $h_b(x_0 + i\eta\theta) = 1$. Then we conduct a binary search between the interval $[x_0+(i-1)\eta\theta, x_0+i\eta\theta]$, within which the classification boundary is located. Note that, in practice, the fine-grained search may exceed the numerical bounds defined by the image (E.g. attacker cannot find an adversarial example along this direction). We can simply assign a maximum distance (e.g., the distance between all-black image and all-white image) for this searching direction to guide the optimizer away from the "hopeless" direction. Unlike the gradient-based method that needs an accurate radius estimation via small ϵ to evaluate the gradient, Bayesian Optimization only needs statistical knowledge about each direction. Therefore, we can use relatively larger ϵ to save query budgets.

B. Bayesian Optimization

The detailed procedure of QE-DBA is presented in Algorithm 2. At beginning, we sample T_0 random low dimensional inputs δ' from the input space and evaluate the distance $g'(\delta')$ using Algorithm 1. Then we iteratively update the posterior distribution of the GP using available data D and query new δ' obtained by maximizing the acquisition function over the current posterior distribution of GP until a valid adversarial example within the desired distortion is found or the maximum number of iteration is reached. Note that the query budget shall be larger than the number of iterations because we need multiple queries to evaluate the distance in Algorithm 1. The alternative stop condition of the optimization procedure is to set a maximum acceptable query budget.

V. Experiments

In this section, we carry out an experimental analysis of our QE-DBA attack. We first compare QE-DBA with other decision-based attack baselines on naturally trained models. Moreover, we also show the performance comparison against models with run-time adversarial example detection. Then, we examine how different types of perturbation generators affect attack success rate and perturbation quality. All experiments are carried out on a 2080 TI GPU.

Algorithm 2: Bayesian Optimization for DBA

```
input : Targeted input $x_0$, Guassian process model
        GP, Acquisition function $\mathcal{A}$, Initialization
        sample size $T_0$, Maximum sample size $T$,
        Distance evaluation function $g'(\cdot)$, stopping
        tolerance $\epsilon$, $D = \varnothing$.
output: Adversarial Examples $x'$
// Intialization
for $t = 0, 1, 2..., T_0 - 1$ do
    Generate input $\delta'_t$ randomly;
    $D \leftarrow D \cup (\delta'_t, g'(\delta'_t))$;
end
Update the GP on $D$;
// Optimization via GP and
   Acquisition function
while $t < T$ do
    $t \leftarrow t + 1$;
    $\delta'_t \leftarrow \arg\max_{\delta'} \mathcal{A}(\delta', D)$;
    if $|g'(\delta'_t)| > \epsilon$ then
        $D \leftarrow D \cup (\delta'_t, g'(\delta'_t))$ and update the GP;
    else
        $\theta = \frac{S(\delta'_t)}{|S(\delta'_t)|}$;
        return $x_0 + g'(\delta'_t)\theta$;
    end
end
// Return nearest adversarial example
$\theta = \frac{S(\delta'_*)}{|S(\delta'_*)|} \mid (\delta'_*, g(\delta'_*)) \in D$ such that $g(\delta'_*) \leq$
  $g(\delta') \quad \forall(\delta', g'(\delta')) \in D$;
return $x_0 + g'(\delta'_*)\theta$;
```

A. Experiments settings

Baselines: We first compare our algorithm with state-of-the-art decision-based attacks: Opt-Attack ([9]), HSJA ([5]), SignOPT ([4]), Bayes attack ([18]) and RayS ([19]). Note that we only compare the attack success rate with Bayes attack because they formulate the attack as an optimization problem with a constraint on perturbation distance. Therefore, Beyas attack only outputs adversarial examples with fixed perturbation distance.

Data and Models: We conduct experiments on two distinct datasets with different input size: ImageNet ([20]) and CIFAR-10 ([21]). In ImageNet, we use two pre-trained DCN models: ResNet-50 ([22]) and Inception V3 ([23]). ResNet-50 takes input images with dimensions $224 \times 224 \times 3$ while Inception V3 take images with dimensions $299 \times 299 \times 3$. In CIFAR-10, we pre-trained networks that achieved an accuracy of 84% and take images with dimensions $32 \times 32 \times 3$.

Metrics: To measure the efficiency, we use the average l_∞ distance between perturbed and original samples over a subset of test images. For each method, we restrict the maximum number of queries to 1000. As an alternative metric, we also evaluate the *attack success rate (ASR)*. An adversarial attack is considered a success if the distortion distance between

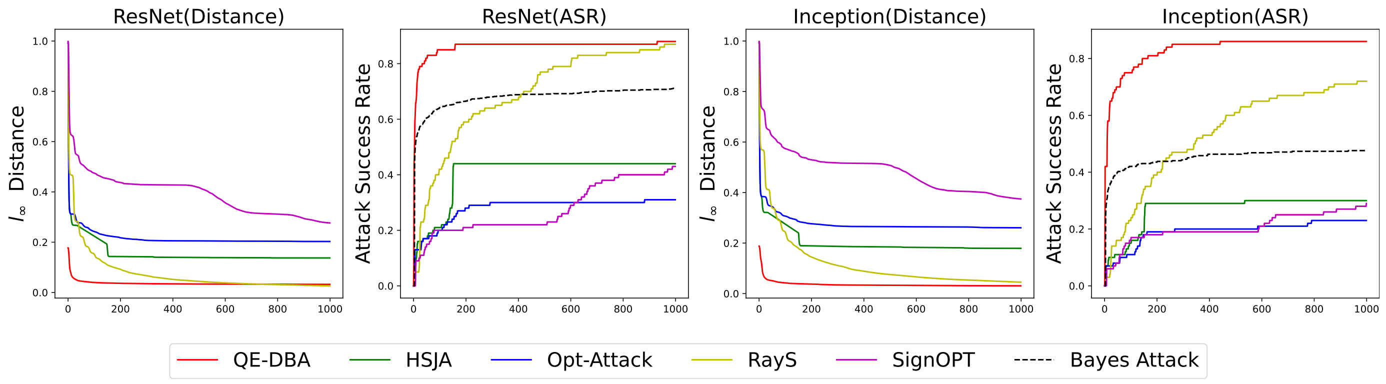

Fig. 2: Average distance versus query budgets on ResNet and Inception. 1st and 3rd cols: untargeted l_∞ distance. 2nd and 4th cols: untargeted Attack Success Rate

generated adversarial examples and original image does not exceed a given distance threshold.

Hyperparameters: In our experiments, we use Perlin noise function from [16] as perturbation generator which has low dimensional inputs $\{\lambda_x, \lambda_y, \phi_{\sin}, \Omega\}$. The perturbation value at point (x, y) with parameters can be formulated using following equation:

$$S_{per}(x,y) = \sin((\sum_{n=1}^{\Omega} p(x \cdot \frac{2^{n-1}}{\lambda_x}, y \cdot \frac{2^{n-1}}{\lambda_y})) \cdot 2\pi\phi_{\sin}) \quad (5)$$

where the $p(x, y)$ is the perlin noise value for coordinates (x, y). In Bayesian Optimization, we use uniformly randomized low dimensional inputs as initialization samples for optimization and the initialization sample size $T_0 = 5$. In the Distance Evaluation Algorithm, we set the increase step size $\eta = \frac{16}{255}$ and set the stopping tolerance $\epsilon = 0.01$.

B. Performance Comparison on ImageNet

In this subsection, we compare QE-DBA with other baselines on the ImageNet dataset which represent the case that the target image size is relatively high.

Figure 2 shows the average l_∞ distance and attack success rate against the query budgets. We compare the l_∞ distortion and attack success rate of our framework with baseline DBA attacks on classifiers trained with ResNet and Inception networks. We can see that QE-DBA consistently achieves a smaller distortion within 1000 queries than baseline methods. As a derivative-free method, QE-DBA can converge within 200 query budgets, while zeroth order optimization based techniques like OPT-Attack, HSJA, and SignOPT[1] need over 15,000 queries to achieve the same level of perturbation distortion ([5]). Although RayS adopts another derivative-free method, it is able to achieve similar perturbation distortion only after around 1000 queries. The obvious advantage of query efficiency of QE-DBA is mainly attributed to facts: 1) QE-DBA adopts the Bayesian Optimization to utilize the statistical information while RayS relies solely on a straightforward hierarchical searching; 2) QE-DBA reduces the searching space via perturbation generators, which results in a much higher convergence rate. In rows 2&4 we also compare our attack success rate with Beyas attack that also adopts BO into DBA setting. Although Beyas attack is also able to converge within very few queries because both of our methods have introduced dimension reduction to facilitate optimization efficiency in terms of query number. QE-DBA can reach a significantly higher attack success rate because we reformulate the problem to optimize the decision boundary radius directly which will lead to better perturbation quality.

[1]Note that the relatively weak performance of SignOPT is due to the fact that SignOPT is designed for l_2 norm attack while this experiment is under the l_∞ norm setting.

Notes that QE-DBA shows a much smaller perturbation distance in query 0 because we set the increase step size $\eta = \frac{16}{255}$ as hyperparameter to generate perturbation with l_∞ norm equal to $\frac{16}{255}$ during initialization. Unlike other works that initialize the algorithm using random perturbations, QE-DBA uses perturbation generator to reduce the input space to a relevant subspace with a higher possibility of finding adversarial examples.

C. Performance comparison Against Run-time Adversarial Example Detection

In addition to defenseless, we also carried out experiments on a classifier that has the run-time adversarial sample detection function ([24]). The function detects abnormal inputs by comparing a DNN model's prediction on the original input with that on squeezed inputs (by reducing the color bit depth of each pixel or spatial smoothing). As shown in Fig.3, QE-DBA achieves the highest overall attack success rate and best query efficiency as compared with the other four hard-label attack baselines.

D. Performance Comparison on CIFAR10

In addition to the dataset with high-resolution images, we also compare our work with RayS and HSJA on the CIFAR-10 dataset via both l_2 and l_∞ attacks. For a fair comparison with previous works, we use the same pre-trained networks that achieved an accuracy of 84% on CIFAR-10. As shown in Table I, within 500 query budgets, compare to gradient-based HSJA attack, QE-DBA consistently achieves a higher attack success rate while requiring fewer queries. As compared to

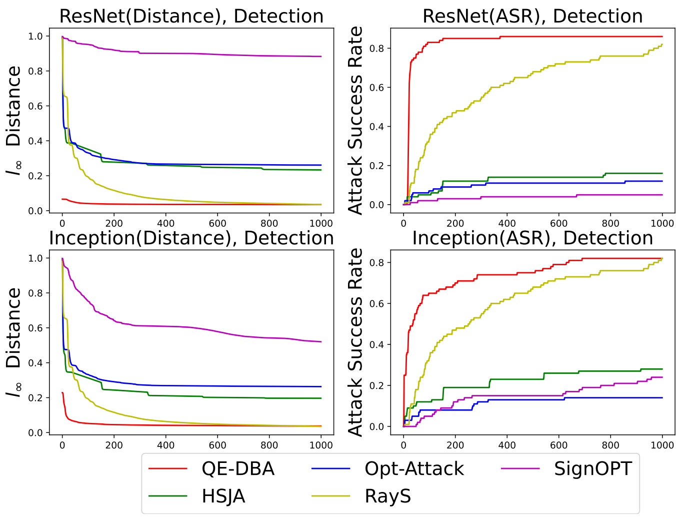

Fig. 3: Average distance versus query budgets on ResNet with adversarial example detection (row 1) and Inception with adversarial example detection (row 2). 1st col: l_∞ distance. 2nd col: ASR

	Avg. l_∞ Distance			ASR ($l_\infty \leq \frac{16}{255}$)		
Queries	50	250	500	50	250	500
QE-DBA	**0.112**	**0.060**	0.049	**43%**	**70%**	73%
RayS	0.228	0.068	**0.039**	39%	58%	**85%**
HSJA	0.298	0.227	0.220	8%	34%	38%
	Avg. l_2 Distance			ASR ($l_2 \leq 5$)		
Queries	50	250	500	50	250	500
QE-DBA	**5.19**	**2.62**	2.20	**59%**	**86%**	89%
RayS	11.38	3.68	**2.10**	18%	73%	**91%**
HSJA	10.10	8.18	7.29	25%	48%	50%

TABLE I: Performance comparison on CIFAR-10 dataset. Rows 1-5: l_∞ attack. Rows 6-10: l_2 attack

RayS, our method achieves a better attack success rate and perturbation quality while the query budget is very low ($<$ 250). RayS will reach a higher attack success rate and lower perturbation distance as the query budgets keep increasing. That's because the input searching space of CIFAR-10 ($32 \times 32 \times 3$) is relatively low. The dimension reduction in QE-DBA will not contribute much in terms of perturbation quality.

VI. Conclusion

In this paper we introduce a new decision-based attack QE-DBA which leverages Bayesian optimization to find adversarial perturbations with high query efficiency. With the optimized perturbation generation process, QE-DBA converges much faster than the state-of-the-art DBA techniques. As compared to existing decision-based attack methods, QE-DBA is able to converge within 200 queries while the state-of-the-art DBA techniques need over 15,000 queries to achieve the same level of perturbation distortion.

References

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[3] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[4] M. Cheng, S. Singh, P. Chen, P.-Y. Chen, S. Liu, and C.-J. Hsieh, "Sign-opt: A query-efficient hard-label adversarial attack," *arXiv preprint arXiv:1909.10773*, 2019.

[5] J. Chen, M. I. Jordan, and M. J. Wainwright, "Hopskipjumpattack: A query-efficient decision-based attack," in *2020 ieee symposium on security and privacy (sp)*, pp. 1277–1294, IEEE, 2020.

[6] H. Li, L. Li, X. Xu, X. Zhang, S. Yang, and B. Li, "Nonlinear projection based gradient estimation for query efficient blackbox attacks," 2021.

[7] J. Zhang, L. Li, H. Li, X. Zhang, S. Yang, and B. Li, "Progressive-scale boundary blackbox attack via projective gradient estimation," *arXiv preprint arXiv:2106.06056*, 2021.

[8] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," *arXiv preprint arXiv:1712.04248*, 2017.

[9] M. Cheng, T. Le, P.-Y. Chen, J. Yi, H. Zhang, and C.-J. Hsieh, "Query-efficient hard-label black-box attack: An optimization-based approach," *arXiv preprint arXiv:1807.04457*, 2018.

[10] J. Mockus, *Bayesian approach to global optimization: theory and applications*, vol. 37. Springer Science & Business Media, 2012.

[11] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*, pp. 63–71, Springer, 2003.

[12] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2015.

[13] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, pp. 2951–2959, 2012.

[14] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[15] S. N. Shukla, A. K. Sahu, D. Willmott, and J. Z. Kolter, "Black-box adversarial attacks with bayesian optimization," *arXiv preprint arXiv:1909.13857*, 2019.

[16] K. T. Co, L. Muñoz-González, S. de Maupeou, and E. C. Lupu, "Procedural noise adversarial examples for black-box attacks on deep convolutional networks," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 275–289, 2019.

[17] B. Ru, A. Cobb, A. Blaas, and Y. Gal, "Bayesopt adversarial attack," in *International Conference on Learning Representations*, 2019.

[18] S. N. Shukla, A. K. Sahu, D. Willmott, and Z. Kolter, "Simple and efficient hard label black-box adversarial attacks in low query budget regimes," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 1461–1469, 2021.

[19] J. Chen and Q. Gu, "Rays: A ray searching method for hard-label adversarial attack," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1739–1747, 2020.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[21] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[23] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[24] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," *arXiv preprint arXiv:1704.01155*, 2017.