Stealthy Query-Efficient Opaque Attack Against Interpretable Deep Learning

Eldor Abdukhamidov, Mohammed Abuhamad, *Senior Member, IEEE*, Simon S. Woo, *Member, IEEE*, Eric Chan-Tin, *Member, IEEE*, and Tamer Abuhmed, *Senior Member, IEEE*

***Abstract*—Deep neural network (DNN) models are susceptible to adversarial samples in white-box and opaque environments. Although previous studies have shown high attack success rates, coupling DNN models with interpretation models could offer a sense of security when a human expert is involved. However, in white-box environments, interpretable deep learning systems (IDLSes) have been shown to be vulnerable to malicious manipulations. As access to the components of IDLSes is limited in opaque settings, it becomes more challenging for the adversary to fool the system. In this work, we propose a *Qu*ery-efficient *Score*-based opaque attack against IDLSes, which requires no knowledge of the target model and its coupled interpretation model. By continuously refining the adversarial samples created based on feedback scores from the IDLS, our approach effectively reduces the number of model queries and navigates the search space to identify perturbations that can fool the system. We evaluate the attack's effectiveness on four convolutional neural network (CNN) models and two interpretation models, using both ImageNet and CIFAR datasets. Our results show that the proposed approach is query-efficient with a high attack success rate that can reach more than 95%, and an average transferability success rate of 69%. We have also demonstrated that our attack is resilient against various preprocessing defense techniques.**

***Index Terms*—Adversarial learning, opaque attack, deep learning, interpretability, transferability.**

I. Introduction

THE tremendous development and deployment of deep learning methods in practice have brought the attention of adversaries to exploit vulnerabilities within the application pipeline to compromise the results or lead the model to misbehave. Numerous studies have been conducted on the robustness and reliability of deep learning models and their behavior in adversarial settings, specifically adversarial examples [1]. Adversarial examples can serve various adversarial purposes, e.g., poisoning, evasion, model extraction, and inference.

Increasing the security, reliability, and understanding of the inner workings of deep neural network (DNN) models, several studies have shown that model interpretability has significant importance in both theory and practice in terms of resilience against adversarial attacks. Interpretable deep learning systems (IDLSes) can be examples of DNN models with interpretable knowledge representations. Furthermore, existing attacks [2], [3], [4] against DNN models are found to be ineffective as the interpretation can reveal adversarial manipulations, i.e., added perturbations to the example input. However, recent studies have shown that IDLSes are still susceptible to adversarial manipulations in white-box settings [5], [6], [7]. It is possible to generate adversarial examples that can mislead the target DNN model and deceive its coupled interpreter simultaneously. For example, an adversarial sample can be misclassified by the target DNN model and interpreted identically to its benign interpretation.

Attacks [5], [6], [7] are based on the white-box scenario, in which the attacker has complete knowledge of the target model and can achieve a high attack success rate (ASR) with high confidence. In practice and in most circumstances, the target model is unreachable. Therefore, this form of attack, i.e., white-box attack, has limited practicality. On the other hand, opaque attacks assume that the adversary can only query the model and access the output without extended knowledge about any of the system's components or the model's parameters. The attack is therefore more realistic in a opaque environment. The most common examples of this type of attack are transfer-based [8], [9], [10] and score-based attacks [11], [12]. Transfer-based methods involve the use of multiple image transformation techniques to enhance the transferability of adversarial examples. The score-based approach is model-agnostic and depends solely on the predicted scores of the model, such as class probabilities or logits. These attacks estimate the gradient numerically using the prediction of the model at a conceptual level. In opaque settings, attacking IDLSes is still an unexplored field, with many challenges to which this work contributes.

In this article, we conduct an in-depth investigation of the security of IDLSes in a opaque environment. Specifically, we

Received 1 December 2023; revised 10 July 2024 and 24 January 2025; accepted 19 February 2025. This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT) under Grant 2021R1A2C1011198, in part by the Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) under the ICT Creative Consilience Program under Grant IITP-2021-2020-0-01821, in part by AI Platform to Fully Adapt and Reflect Privacy-Policy Changes under Grant RS-2022-II220688, and in part by Artificial Intelligence Innovation Hub under Grant RS-2021- II212068. Associate Editor: Y. Le Traon. *(Corresponding author: Tamer Abuhmed.)*

Eldor Abdukhamidov and Tamer Abuhmed are with the Department of Computer Science and Engineering, Sungkyunkwan University, Suwon 03063, South Korea (e-mail: abdukhamidov@skku.edu; tamer@skku.edu).

Mohammed Abuhamad and Eric Chan-Tin are with the Department of Computer Science, Loyola University, Chicago, IL 60660 USA (e-mail: mabuhamad@luc.edu; chantin@cs.luc.edu).

Simon S. Woo is with the Department of Artificial Intelligence, Sungkyunkwan University, Suwon 03063, South Korea, and also with the Department of Applied Data Science, Sungkyunkwan University, Suwon 03063, South Korea (e-mail: swoo@g.skku.edu).

Digital Object Identifier 10.1109/TR.2025.3551717

propose a novel opaque attack that generates adversarial examples to mislead the target DNN models and their coupled interpreters. We use transfer-based strategies to take advantage of the transferability of adversarial samples across various DNN models. In addition, we use score-based techniques to efficiently guide the search process. This ensures that our attack is query-efficient and practical in real-world scenarios.

By evaluating our approach against four DNN models (i.e., Inception-V3, DenseNet-169, VGG-19, and ResNet-50) and two interpreters (i.e., CAM and Grad) on various datasets, we show the possibility and practicality of generating successful adversarial examples with accurate interpretations in a opaque environment. In all test cases, the attack had a success rate of more than 95% with a high degree of similarity of more than 80% in interpretation.

Contributions: We summarize our contributions as follows.

1) We propose a stealthy and query-efficient opaque attack against IDLSes. We evaluate the effectiveness of the attack from the perspective of four DNN models (i.e., Inception-V3 [13], ResNet-50 [14], Densenet-169 [15], and VGG-19 [16]) and two interpretation models (CAM [17] and Grad [18]). Our results show that the proposed attack achieves a high success rate of 95% to 100% in attacking target models and their interpreters on ImageNet and CIFAR datasets, with an average of only 150 queries.
2) We introduce a modified variant of genetic algorithm designed to enhance the practical applicability of the opaque attack in real-life scenarios. This algorithm optimizes the attack process, making it more adaptable and effective against various defense mechanisms and models.
3) We present the robustness of the proposed attack when using five defense mechanisms: i.e., bit depth compression, median smoothing, JPEG compression, random resizing/padding, and adversarial training. The results show that our approach achieved a success rate ranging from 81% to 100% and required an average of only 137 queries when confronting most of the defense methods.
4) We evaluate the transferability of the attack against different DNN models. The results show high transferability across various datasets and DNN models, indicating its effectiveness in generating adversarial examples that can deceive different models with an average success rate of 69% and an interpretability IoU score of 90%.

Organization: The rest of this article is organized as follows. Section II surveys related research studies; Section III describes the notations and terms used in this article and presents the proposed attack and its underlying mechanisms; Section IV provides the results of the attack effectiveness, robustness, and transferability against DNN and interpretation models; Section V explains the existing limitations and future work. Finally, Section VI concludes this article.

II. Related Work

This section provides an overview of the related research on attacks against DNN models. Specifically, it covers previous work on both white-box and opaque attacks that employ various techniques, including transfer-based attacks, interpretation-based attacks, and score-based attacks.

A. Transfer-Based Attacks

Previous research has shown that adversarial samples generated to attack specific DNN models can also be used to fool other DNN models. Such attacks, known as transfer-based attacks [10], [19], use adversarial inputs generated by white-box attacks against DNNs to attack other opaque models. Huang et al. [9] proposed a method that adds perturbations to the hidden layers of a model to enhance the transferability of adversarial samples. Furthermore, recent work [20] addresses the issue of weak transferability in opaque models by introducing a transfer-based sparse attack method, called adaptive momentum variance-based iterative gradient method with a class activation map. This method uses an adaptive momentum variance and a refining perturbation mechanism to enhance the transferability of adversarial examples while limiting the intensity of perturbation.

B. Interpretation-Based Attacks

Adversarial attacks based on interpretation generate samples that can deceive both the target DNN models and their interpreters simultaneously [5]. Abdukhamidov et al. [6] proposed white-box attacks called AdvEdge and AdvEdge$^+$ against DNN models and their coupled interpreters. These attacks demonstrate the vulnerability of DNN models that rely on interpretable features for decision-making and highlight the need to consider the interpretability of DNN models in addition to their accuracy and robustness. The proposed attack provides a valuable tool to assess the interpretability of DNN models and their susceptibility to adversarial attacks. Abdukhamidov et al. [21] proposed the single-class target-specific adversarial attack called SingleADV to craft a universal perturbation that leads the target model to misclassify a particular category as a different, intended category, while maintaining precise and highly relevant interpretations.

C. Score-Based Attacks

Heuristic methods, such as evolution strategies and genetic algorithms, have been used to devise adversarial attacks that can generate visually imperceptible samples to deceive DNN models [12]. GenAttack is a gradient-free optimization attack that can generate adversarial samples against opaque models with fewer queries. Another study proposed a query-efficient attack called MGAAttack [11], which uses transfer-based techniques to improve its efficacy. These attacks highlight the vulnerability of DNN models to adversarial attacks and the need to develop more robust defense mechanisms [22] to enhance their security. By studying these attacks, researchers can identify weaknesses in DNN models and devise better defenses against adversarial attacks.

III. Query-Efficient Score (QuScore): Methods

The section describes QuScore in opaque settings with a detailed explanation of the adopted methods.

A. Concepts and Notations

In this section, we introduce the notation, terms, and symbols used in this article.

1) Classifier: Image classification is the primary focus of this work, and we employ two types of DNN models: white-box and opaque models. In this article, $f(x) = y \in Y$ denotes a classifier in a opaque setting (target DNN model), and $f'(x) = y \in Y$ denotes a classifier in a white-box setting (source DNN model), where y refers to a single category from a set of categories Y.

2) Interpreter: We adopt an existing interpretation model g to produce an interpretation map m to display the importance of features for a sample x, which are found by a classifier f: $g(x; f) = m$. For our approach, post hoc interpretability is used [23], [24], [25], [26], where an interpretation model g obtains information about the sample input x and its classification decision by the classifier f to generate an attribution map. This type of interpretation requires another model to interpret the decision process of the current classification model.

3) Adversarial Attack: Pixel perturbation attack known as projected gradient descent (PGD) [2] is adopted by AdvEdge attack to generate perturbations. It generates an adversarial sample $\hat{x}$ to cause the source DNN model f' to misclassify $\hat{x}$ into another category: $f'(\hat{x}) \neq y$. PGD is implemented as follows:

$$\hat{x}^{(i+1)} = \prod_{\mathcal{B}_\varepsilon(x)} \left(\hat{x}^{(i)} - \alpha \cdot \operatorname{sign}(\nabla_{\hat{x}} \ell_{\text{adv}}(f'(\hat{x}^{(i)}))) \right)$$

where $\prod$ is the projection operator, α is the learning rate, ℓ_{adv} is the loss function (i.e., cross entropy), $\mathcal{B}_\varepsilon(x)$ is the norm ball limited with the specific range ε, and $\hat{x}^{(i)}$ is $\hat{x}$ at iteration i.

4) Threat Model: In this work, we consider a opaque setting, where the adversary has limited access to the target DNN classifier f (model output). Specifically, the adversary can query the target model and receive the output probabilities or scores for each class, but cannot access the model's internal parameters or architecture. This setting resembles a realistic scenario for the attack, simulating conditions where the adversary can only observe the model's predictions. We assume the following specifics within our threat model.

1) *Adversarial knowledge:* The adversary can query the target model and obtain the output probabilities or scores corresponding to each class. That is, the adversary has no access to the internal parameters, architecture, or training process of the target model. Moreover, the adversary may have some knowledge of the general type of data on which the target model is trained (e.g., images, text) but does not have access to the specific training dataset.
2) *Adversarial capabilities:* The adversary can train a source model that can be used to generate the initial population for the attack. The adversary can utilize public datasets to train the source model. In addition, the adversary can design and train interpretation models based on the source classifier and public datasets.

The threat model reflects realistic attack conditions, such as those encountered in practical applications where models are exposed through APIs and only the prediction scores are accessible to users.

B. Attack Formulation

Attacking IDLSes requires fooling both a DNN model and its associated interpretation model. To increase the stealthiness of the attack, we use AdvEdge [6] to generate the initial population of candidates for our attack. We note that AdvEdge is a white-box attack that takes advantage of edge information in the input sample to generate an adversarial sample to mislead a DNN classifier f' and its interpretation model g simultaneously. Our modified genetic algorithm uses these initial candidates to produce generations of mutated adversarial examples that can succeed in a opaque attack to make it more practical in real-life scenarios. The main objective of the attack is to generate an adversarial sample $\hat{x}$ by fulfilling the following conditions:

❶ $\hat{x}$ should fool the source DNN model f': $f'(\hat{x}) \neq y$;

❷ an interpretation map $\hat{m}$ of the adversarial sample $\hat{x}$ generated by an interpreter g should be similar to the interpretation map m of the benign sample x: $g(\hat{x}; f') = \hat{m}$ s.t. $\hat{m} \cong m$;

❸ the adversarial sample $\hat{x}$ and its benign version x should be visually imperceptible;

❹ The amount of noise is restricted to the edge of the image input. At a high level, the attack framework is formulated as follows:

$$\min_{\hat{x}} : \Delta(\hat{x}, x) \quad s.t. \begin{cases} f'(\hat{x}) \neq y, & \text{s.t.} \quad \|\hat{x} - x\|_\infty \in \{-\epsilon, \epsilon\} \\ g(\hat{x}; f') = \hat{m}, & s.t. \quad \hat{m} \cong m \\ \Delta(\hat{x}, x) \sim \text{edge}(x \cap m). \end{cases} \tag{1}$$

Equation (1) ensures that the prediction of the adversarial sample is not equal to the original category; an interpretation map of the adversarial sample is similar to the interpretation map of its benign version; the added perturbation lies within the edges of the sample that intersect with the interpretation map of the sample. The attack aims to minimize the overall adversarial loss (ℓ_{adv}) considering the classification loss $\ell_{\text{prd}}(f'(x)) = -\log(f'(x))$ and the interpretation loss $\ell_{\text{int}}(g(x; f', m) = \|g(x; f') - m\|_2^2$.

The overall adversarial loss is then defined as follows:

$$\ell_{\text{adv}} = \min_{\hat{x}} \ell_{\text{prd}}(f'(\hat{x})) + \lambda\, \ell_{\text{int}}(g(\hat{x}; f'), m) \tag{2}$$

where the hyperparameter λ balances ℓ_{prd} and ℓ_{int}. The final adversarial framework can be described as follows:

$$\hat{x}^{(i+1)} = \prod_{\mathcal{B}_\varepsilon(x)} \left(\hat{x}^{(i)} - N_w\, \alpha. \operatorname{sign}(\nabla_{\hat{x}} \ell_{\text{adv}}(\hat{x}^{(i)})) \right) \tag{3}$$

where $\prod$ is the production operator, $\mathcal{B}_\varepsilon$ is a norm ball, α is the learning rate, x is the sample input, and $\hat{x}^{(i)}$ is the adversarial sample at the ith iteration. N_w is the edge operator function that is used to optimize the location and magnitude of the added perturbation:

$$d = \sqrt{d_h^2 + d_v^2}$$

$$N_w = d \cap m$$

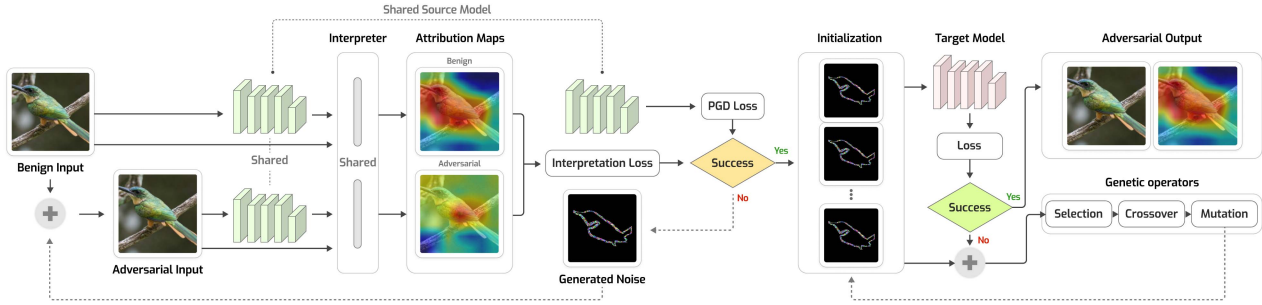


Fig. 1. Framework of QuScore: After generating successful adversarial examples using AdvEdge built upon the PGD attack on the source model, these examples are introduced as the initial population candidates for genetic operation-based refinement. Each candidate is evaluated against the target model (i.e., opaque model) to assess its effectiveness. If a candidate is ineffective, two distinct samples—one highly impactful and the other less so—are chosen for genetic crossover and mutation. This process generates new samples subjected to strict evaluation against the target model. The iterative process continues, adapting and refining the perturbations, until an evolved sample is produced that successfully deceives the target IDLS.

where d is an image that contains edges of the sample x extracted by $d = \sqrt{d_h^2 + d_v^2}$. d_h and d_v contain the sample's horizontal and vertical edge information. The attack uses the intersection of the edges of a sample image and its interpretation map to extract important regions.

As in (3), AdvEdge adopts the PGD framework [2] to generate perturbation, and it is controlled by (2) to fool the target DNN model and mislead the coupled interpreter. This approach is used in a white-box scenario where the model's internal structure is known. However, to adapt this strategy for opaque environments where such detailed knowledge of the target DNN model is unavailable, we focus on creating an efficient and adaptive method that retains the precise positioning of perturbations while exploring an adequate adversarial search space without direct insight into the target DNN model. To this end, we leverage the transferability property of attacks. This property allows adversarial examples crafted on one model to be practical on another, even without explicit knowledge of the second model's architecture. We propose a genetic algorithm-based attack to generate perturbations that are strategically positioned and effective in the opaque scenario. The attack is described in Section III-C.

Algorithm 1: QuScore in Opaque Settings.

Data: Source DNN f', interpreter g, input x, original category y, perturbation threshold ϵ, mutation rate mr, crossover rate cr, population size n, generation G, target DNN f
Result: Adversarial sample $\hat{x}$

1 x' = advedge_attack(f', g, x, n)
2 pop = init_population(x, x', ϵ)
3 **for** $g \leftarrow 1$ **to** G **do**
4 p_1, p_2 = random_select(pop)
5 v_1, v_2 = get_fitness(f, x, p_1, p_2)
6 $alpha$, $omega$ = sort_by_fitness(p_1, p_2, v_1, v_2)
7 $child$ = crossover(cr, $omega$, $alpha$)
8 $child$ = mutation(mr, $child$)
9 **if** $f(child) \neq y$ **then**
10 return $child$
11 **end**
12 pop = update_population(pop, $child$)
13 **end**

C. QuScore Optimization

Traditional genetic algorithms are ineffective for our objectives as they primarily target classifiers. To address this, we introduce our modified version of the genetic algorithm, designed for gradient-free optimization. Our algorithm considers model interpretation and significantly advances the effectiveness of genetic progression in navigating complex optimization challenges (see Fig. 1). First, we generate adversarial samples against a "*source*" DNN model in a white-box setting and use them as the initial population for genetic operators. Genetic operators update the initial population to produce new generations of these adversarial examples to deceive a opaque "*target*" DNN model and mislead its coupled interpreter. Using AdvEdge in the initial seed generation, the strategy of our algorithm involves maintaining original perturbation positions while adding minimal perturbations with the aim of creating adversarial inputs that can fool the opaque model and its coupled interpreter. This is done through mutations and crossovers to increase the chances of generating effective adversarial examples in a opaque setting.

Algorithm 1 shows the detailed information of the attack. Genetic operators of our attack are as follows: *initialization* (line 1 and 2), *selection* (line 4–6), *crossover* (line 7), *mutation* (line 8), and *population update* (line 12).

Initialization: Seeding the initial population is the first phase of the genetic operators. Although the process is executed only once during the attack process, the initial population is crucial to the convergence of the technique. The population with an optimal solution helps the technique converge quickly. In our case, we generate adversarial samples using the AdvEdge attack, which is explained in Section III-B and provide them as the initial population $\Psi : \{\psi_1, \psi_2, \ldots, \psi_m\}$, where m is the size of the population.

Fitness function: This is also known as the evaluation phase. It is used to assess the quality of individuals in a population and help evolve toward the optimal population. In other words, it evaluates how close the given sample is to meet the attack requirements. In our attack, we evaluate each individual in the population by applying a loss function (i.e., cross entropy) as the fitness function: $\text{argmax}_{\hat{x}} L(\hat{x}, y)$ s.t. $\Delta(\hat{x}, x) \leq \epsilon$. Loss

values reflect the fitness scores of each sample in the population. The desired adversarial samples have higher fitness scores, while others have lower fitness scores.

Selection: This step helps a new generation inherit genetic information by selecting samples. In traditional genetic algorithms, the proportionate fitness selection technique [27] is widely used, in which samples in the population with high fitness scores have higher chances of propagating their features to the next generation. The technique helps generate better adversarial samples to fool a DNN model, but those samples can have higher chances of being detectable when an interpreter is applied. In our case, our main objective is to generate undetectable adversarial samples by an interpreter. Based on the objective, we randomly selected two samples from the population, one with a higher fitness score and the second with a lower fitness score. We refer to them as "alpha" (larger fitness scores) and "omega." By doing that, we try to keep the perturbation in the newly generated sample's area that is considered important by the target DNN model and its interpreter. After the process, the selected samples are passed to the crossover phase.

Crossover: The phase is also called recombination. The function combines genetic information from the selected parents to produce new offspring. In traditional genetic algorithms, two individuals with high fitness scores are selected to pass on their genetic information to a new offspring. In our case, we have an alpha and an omega after the selection process. We generate a new offspring (adversarial sample) by transferring the genetic data of the alpha and omega with the predefined crossover rate cr: $\psi_{\text{child}} = \psi_{\text{alpha}} * S_{\text{cr}} + \psi_{\text{omega}} * (1 - S_{\text{cr}})$, where S_{cr} is a matrix with values of 1 and 0. ψ_{alpha} and ψ_{omega} are the samples that we selected in the previous step. S_{cr} is generated based on the crossover rate cr as follows:

$$S_{cr} = \begin{cases} 1, & \text{rand}(0,1) < \text{cr} \\ 0, & \text{otherwise} \end{cases}$$

where $\text{rand}(0,1)$ generates uniformly distributed numbers between 0 and 1, cr is the probability of crossover (i.e., the crossover rate). In the experiment, we use the default value for the crossover rate, which is 0.7.

Mutation: The process diversifies the population and helps reach the points outside the regions of the local optima [28]. The attack requires enough diversity in the population, which the mutation function can introduce. Mutation can be carried out via binary encoding: $\psi_{\text{child}} = -\psi_{\text{child}} * S_{mr} + \psi_{\text{child}} * (1 - S_{\text{mr}})$, where S_{mr} is a binary matrix that is generated based on the mutation rate mr:

$$S_{\text{mr}} = \begin{cases} 1, & \text{rand}(0,1) < \text{mr} \\ 0, & \text{otherwise} \end{cases}$$

where mr is the probability of mutation (i.e., the mutation rate). We use the default value for the mutation rate in the experiment, 1e-4. Fig. 2 illustrates the process.

Population update: For continuous evolution, the population should be updated by keeping the alphas and replacing the omegas with new generations. In this step, the selected omega is replaced with the mutated offspring.

The size of the population is an important parameter that directly affects the ability to search for an optimal solution in the search space. As our attack must satisfy more requirements than traditional attacks, the adversarial search space becomes smaller. In that case, having a large population size is not sufficient, which can lead to increased time complexity and make the search more complex by having more generations converge. Taking this into consideration, we use 5 as the population size for our experiment.

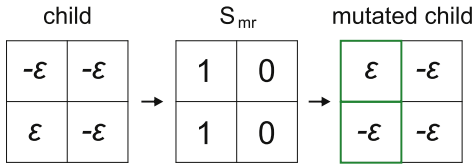


Fig. 2. Example illustration of the mutation process. S_{mr} is the binary matrix generated based on mutation probability.

In summary, AdvEdge generates adversarial samples against a source DNN model f' and its coupled interpreter g in a white-box environment. Then, we provide those generated samples as the seed for the initial population. The algorithm evaluates the fitness scores of the population by sending them to the target DNN model f (opaque). In our case, the population size is five. Therefore, we use five queries to evaluate the initial population. When evaluating the initial population, if one of the individuals meets the attack requirements as the desired adversarial sample, the algorithm stops further steps. Otherwise, it randomly chooses two individuals (an alpha and an omega) from the population and calculates their fitness scores if any child is generated from a previous iteration (this process is applied if this is not the first iteration). After that, it generates a new offspring from alpha and omega by conducting a crossover process. A newly created offspring is mutated in the mutation phase. Finally, the omega is replaced by that offspring, and the alpha is kept. The attack repeats the steps until it succeeds or reaches the query threshold.

D. Effectiveness of QuScore

The effectiveness of QuScore is based on several concepts as follows.

Leveraging Transferability: QuScore takes advantage of the transferability property of adversarial examples. Using AdvEdge to generate an initial population for MGA against a source model provides a strong starting point for the attack with perturbations likely to transfer to the target model, offering a significant advantage over random initialization.

Perturbation Placement: The attack generates perturbations concentrated in semantically important areas of the image, as determined by the source model and interpreter.

Adaptive Search in Opaque Settings: Adopting MGA with our approach allows for perturbation optimization with limited information from the target model, which allows efficient navigation of the search space in opaque settings.

Balance of Exploration and Exploitation: The selection phase of QuScore chooses both high and low fitness samples, combined with tuned mutation and crossover operations, allows the algorithm to exploit promising solutions and explore new areas of the search space.

Dual Optimization Objective: QuScore simultaneously optimizes for fooling the classifier and maintaining accurate interpretation. These factors enables QuScore to generate high-quality adversarial examples that are effective against opaque models.

IV. Experiments and Results

This section provides the settings and metrics used for our experiments and the results of the proposed attack. For the reproducibility of our experiments, our code, data, and models are available online.[1]

A. Experimental Settings and Evaluation Metrics

1) Datasets: Our experiment uses ImageNet, CIFAR-10, and CIFAR-100 datasets, which cover 1.2 million images for 1000 categories, 60 000 images for ten categories and 60 000 images for 100 categories, respectively. We randomly select an image from each category of the validation set in ImageNet and testing sets in CIFAR-10 and 100 (overall 3000 images, 1000 images per dataset), which are correctly classified by the selected DNN model f with a classification confidence score higher than 60%. In the experiment, we set ϵ to 8 for the datasets on the scale of [0, 225], which is similar to the settings of the AdvEdge attack [6]. By experimenting with the selected images, we ensure that the proposed attack is valid across all categories of the selected DNN models.

2) Classifiers: Our experiment includes different well-known DNN models. They include Inception-V3, DenseNet-169, VGG-19, and ResNet-50, which demonstrated high performance with 76.6%, 77.9%, 71.3% and 77.2% top-1 accuracy on ImageNet, respectively. We use DenseNet-169 and ResNet-50 for two purposes in our evaluation. In our study, we chose to use pretrained versions of these four models because of their proficiency in learning robust feature representations from our used datasets. They are used as source DNN models (white-box models) to implement the transfer-based attack part, and they are also used as target DNN models (opaque models) to be attacked. Note that we do not use those two models to attack themselves. Specifically, we do not use ResNet-50 (in white-box setting) to attack ResNet-50 (in opaque setting), and the same is true for DenseNet-169 to create a realistic opaque attack scenario. For hyperparameters, as we have a limited adversarial search space, we set the maximum query to a larger number, which is 50 000. In the transfer-based approach, the step size α and the number of iterations are set to 1/255 and 300, which are the same as the AdvEdge attack settings [6].

3) Interpreters: CAM [17] and Grad [18] interpreters are adopted as the representatives of interpretation models. They represent different types of interpretation models, as they use different characteristics of DNN models. CAM uses the feature maps of convolutional layers in a DNN model to generate interpretation maps: $m_c = \sum_i w_{i,c} a_i(j,k)$, where $a_i(j,k)$ is the activation of the ith channel at the spatial location (j,k) and $w_{i,c}$ is the weight of the ith input and the cth output in the linear layer of a DNN model. Grad calculates the gradients of a prediction of a DNN model based on a sample input: $m = |\frac{\partial f_y(x)}{\partial x}|$. We set λ in (2) at 0.007 and 0.204 for Grad and CAM, respectively [6]. We use their open-source implementations for the experiment.

4) Evaluation Metrics: We apply different evaluation metrics as we use different DNN classifiers and interpreters. The following metrics are used to evaluate the attack.

1) *ASR:* Calculates the ratio of successful attack cases to total attack cases: $\{\#\text{successful\_test\_cases}\} \div \{\#\text{total\_test\_cases}\}$. A successful test case is one that causes the target model to misclassify an image, which is the ultimate goal of the attack. A higher success rate indicates that the attack algorithm is more effective in generating successful adversarial examples.
2) *Average queries:* The efficiency of the attack algorithm is evaluated using the metric of the average number of queries required to generate successful adversarial examples in a opaque setting. This metric is important as it reflects the amount of information the attacker needs to obtain from the target model. A lower average number of queries indicates that the attack algorithm is more efficient in generating adversarial examples with limited access to the target model.
3) *Noise rate:* The noise rate metric is used to evaluate the quality of adversarial examples generated by the attack algorithm. It measures the amount of noise that needs to be added to the original image to create a successful adversarial example. A lower noise rate indicates that the attack algorithm is more effective in generating high-quality adversarial examples with smaller amounts of noise. The amount of disturbance is calculated using the structural similarity index (SSIM) [29]. SSIM measures the similarity score, and we find the nonsimilarity portion using that score (i.e., noise_rate $= 1 -$ SSIM).

To evaluate the effectiveness of the attack against interpreters, we employ the following metrics.

1) *Qualitative comparison:* The metric measures the ability of the attack algorithm to generate adversarial examples that are difficult to distinguish from benign cases based on the visual inspection of images and their interpretations.
2) *Intersection-over-union (IoU) Test:* We use the metric to measure the similarity of interpretation maps. In the metric, interpretation maps are converted into binary-valued maps based on a threshold to compare adversarial maps with benign ones. We use different numbers as threshold values (from 0.1 to 1.0), resulting in an overall 9 threshold values for each interpretation map. Then, we measure the IoU scores using those threshold values per interpretation map and calculate their average. The IoU metric is evaluated only on successful adversarial examples. This ensures that we measure the similarity of interpretation maps specifically for adversarial examples that have successfully fooled the target model. The IoU score is calculated as follows: $\text{IoU} = \text{Area of Overlap}/\text{Area of Union}$ By averaging the IoU scores across all thresholds, we obtain a comprehensive measure of similarity.

We use these metrics to calculate the effectiveness of the attack against DNN classifiers with and without defenses.

[1] [Online]. Available: https://github.com/InfoLab-SKKU/QuScore

TABLE I
SUCCESS RATE, AVERAGE QUERIES, AND AVERAGE NOISE OF QUSCORE AND THE HYBRID ATTACK AGAINST DIFFERENT CLASSIFIERS AND INTERPRETERS TESTING ON 1000 IMAGES FOR EACH DATASET (TOTAL 3000 IMAGES)

Interpreter	Source Model	Target Model	QuScore				Hybrid Attack [30]			
			ASR	Avg. Queries	Median Query	Avg. Noise Rate	ASR	Avg. Queries	Median Query	Avg. Noise Rate
ImageNet										
CAM	ResNet	InceptionV3	0.95	438.24	5	0.21 ± 0.06	0.95	890.48	465	0.64 ± 0.05
		DenseNet	0.99	209.76	5	0.20 ± 0.06	0.97	437.52	465	0.64 ± 0.05
		VGG	1.00	179.80	5	0.20 ± 0.06	0.92	369.60	465	0.64 ± 0.05
	DenseNet	InceptionV3	0.95	363.31	5	0.21 ± 0.06	0.93	744.62	465	0.64 ± 0.05
		ResNet	1.00	188.53	5	0.20 ± 0.06	0.95	389.06	465	0.64 ± 0.05
		VGG	1.00	158.33	5	0.20 ± 0.06	0.92	336.66	465	0.64 ± 0.05
GS	ResNet	InceptionV3	0.95	479.93	5	0.21 ± 0.06	0.95	890.48	465	0.64 ± 0.05
		DenseNet	1.00	231.62	5	0.21 ± 0.06	0.97	437.52	465	0.64 ± 0.05
		VGG	1.00	180.04	5	0.20 ± 0.06	0.92	369.60	465	0.64 ± 0.05
	DenseNet	InceptionV3	0.95	372.12	5	0.21 ± 0.06	0.93	744.62	465	0.64 ± 0.05
		ResNet	1.00	189.08	5	0.20 ± 0.06	0.95	389.06	465	0.64 ± 0.05
		VGG	1.00	161.25	5	0.20 ± 0.06	0.92	336.66	465	0.64 ± 0.05
CIFAR-100										
CAM	ResNet	InceptionV3	1.00	145.00	5	0.04 ± 0.03	0.88	301.00	183	0.30 ± 0.04
		DenseNet	1.00	70.36	5	0.04 ± 0.03	0.89	159.72	183	0.30 ± 0.04
		VGG	1.00	60.31	5	0.04 ± 0.03	0.90	132.62	183	0.30 ± 0.04
	DenseNet	InceptionV3	1.00	120.87	5	0.04 ± 0.03	0.87	259.74	183	0.30 ± 0.04
		ResNet	1.00	61.24	5	0.04 ± 0.03	0.90	139.48	183	0.30 ± 0.04
		VGG	1.00	53.11	5	0.04 ± 0.03	0.89	125.22	183	0.30 ± 0.04
GS	ResNet	InceptionV3	1.00	154.98	5	0.04 ± 0.03	0.88	301.00	183	0.30 ± 0.04
		DenseNet	1.00	77.69	5	0.04 ± 0.03	0.89	159.72	183	0.30 ± 0.04
		VGG	1.00	60.39	5	0.04 ± 0.03	0.90	132.62	183	0.30 ± 0.04
	DenseNet	InceptionV3	1.00	122.82	5	0.04 ± 0.03	0.87	259.74	183	0.30 ± 0.04
		ResNet	1.00	65.42	5	0.04 ± 0.03	0.90	139.48	183	0.30 ± 0.04
		VGG	1.00	56.09	5	0.04 ± 0.03	0.89	125.22	183	0.30 ± 0.04
CIFAR-10										
CAM	ResNet	InceptionV3	1.00	153.03	5	0.04 ± 0.03	0.84	252.55	146	0.30 ± 0.04
		DenseNet	1.00	78.45	5	0.04 ± 0.03	0.84	140.68	146	0.30 ± 0.04
		VGG	1.00	75.18	5	0.04 ± 0.03	0.85	130.77	146	0.30 ± 0.04
	DenseNet	InceptionV3	1.00	138.66	5	0.04 ± 0.03	0.83	231.99	146	0.30 ± 0.04
		ResNet	1.00	80.43	5	0.04 ± 0.03	0.85	144.65	146	0.30 ± 0.04
		VGG	1.00	67.23	5	0.04 ± 0.03	0.85	122.85	146	0.30 ± 0.04
GS	ResNet	InceptionV3	1.00	169.78	5	0.04 ± 0.03	0.84	252.55	146	0.30 ± 0.04
		DenseNet	1.00	86.28	5	0.04 ± 0.03	0.84	140.68	146	0.30 ± 0.04
		VGG	1.00	69.73	5	0.04 ± 0.03	0.85	130.77	146	0.30 ± 0.04
	DenseNet	InceptionV3	1.00	140.18	5	0.04 ± 0.03	0.83	231.99	146	0.30 ± 0.04
		ResNet	1.00	76.82	5	0.04 ± 0.03	0.85	144.65	146	0.30 ± 0.04
		VGG	1.00	72.16	5	0.04 ± 0.03	0.85	122.85	146	0.30 ± 0.04

Results of hybrid attack are repeated for different interpreters, as the attack does not consider attacking interpreter.

B. Attack Effectiveness Against DNNs

In this section, we discuss the effectiveness of our proposed attack on a set of four popular DNNs as target models, namely, Inception-V3, ResNet, DenseNet, and VGG. Our attack is implemented and tested on two interpreters with two source models as follows: (1) CAM with ResNet, (2) CAM with DenseNet, (3) Grad with ResNet, and (4) Grad with DenseNet. We compare our results with the existing attack, i.e., "*Hybrid Attack*" [30]. We used the source code provided by the authors of this article and the same setting as ours for a fair comparison.

While our approach follows a similar high-level protocol to [30], utilizing a white-box attack followed by opaque refinement, it introduces several novel elements that significantly advance the state of the art in attacking IDLSes. Unlike [30], which focuses solely on fooling classifiers, our method tackles the more challenging dual objective of misleading both the classifier and its coupled interpreter. We achieve this by utilizing AdvEdge for initial population generation, taking into account both classification and interpretation from the beginning. Our modified genetic algorithm, designed with a specific selection process and genetic operators, enables a more thorough exploration of the adversarial space while preserving perturbations in areas of semantic significance. In addition, our method shows improved adaptability to different opaque situations and potentially increased query efficiency.

Table I reports the results of our experiments on the scenarios mentioned above. The table uses a different color in the existing attack part to show that the results are the same in both interpreter parts. This is because the attack method does not change for different interpreters, as the attack does not use an interpreter in the attack process.

1) CAM Interpreter With ResNet: For the ImageNet dataset, the proposed opaque attack shows a high success rate of 0.95 to 1.00 for InceptionV3, DenseNet, and VGG target models with an average number of queries between 179.80 and 438.24 and an average noise rate of 0.20 ± 0.06. In contrast, the Hybrid Attack demonstrates similar success rates but requires significantly more queries (ranging from 369.60 to 890.48) and demonstrates a higher average noise rate of 0.64 ± 0.05.

For the CIFAR-100 and CIFAR-10 datasets, the proposed attack maintains a 100% success rate across all target models with fewer queries needed (i.e., ranging from 60.31 to 145.00 for CIFAR-100 and 75.18 to 153.03 for CIFAR-10) and a lower average noise rate of 0.04 ± 0.03. The existing attack, however, shows slightly reduced success rates (0.88 to 0.90) and again requires more queries, suggesting that the proposed attack is more efficient and stealthy compared to it.

2) CAM interpreter with DenseNet: The proposed attack with a CAM interpreter and DenseNet as the source model shows success rates of 0.95 to 1.00 on ImageNet, with a lower average number of queries (158.33 to 363.31) and noise (0.20 ± 0.06)

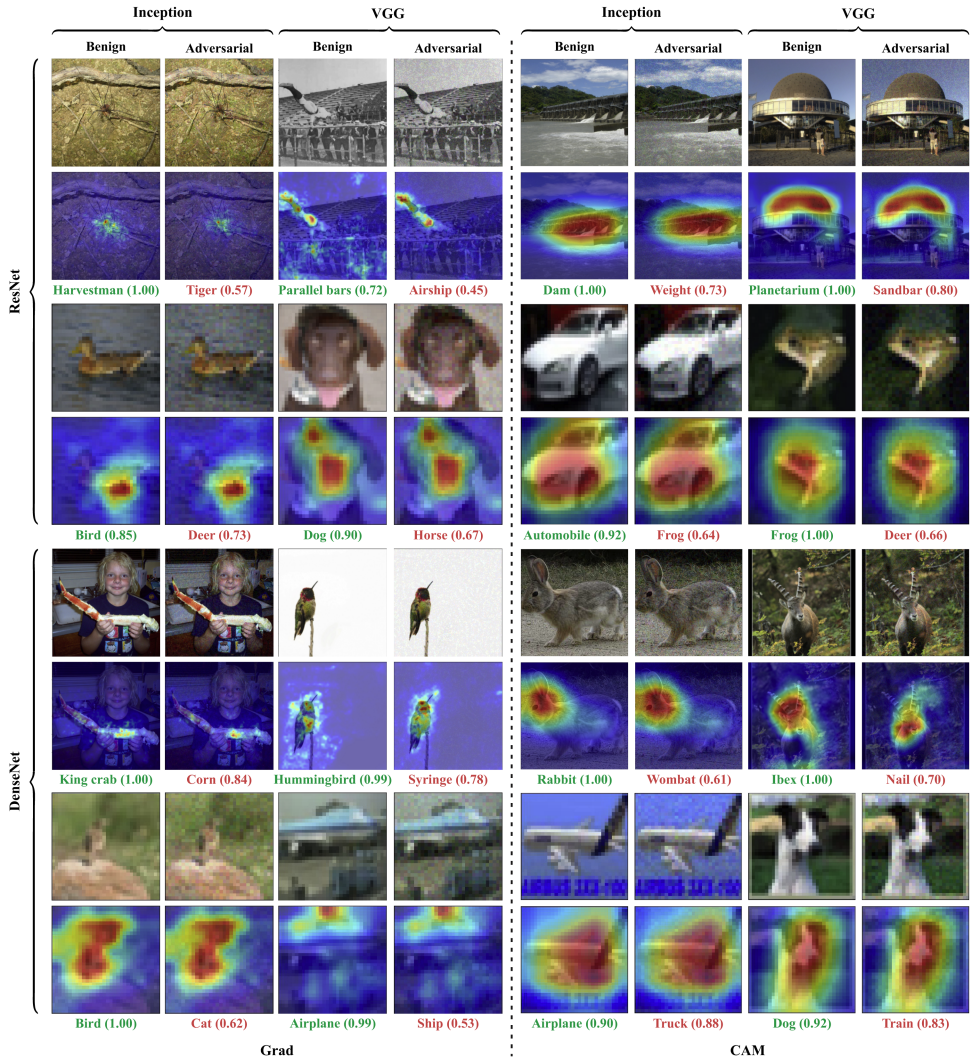


Fig. 3. Attribution maps of benign and adversarial samples using QuScore with respect to Grad and CAM on ResNet, DenseNet as source models, and Inception, VGG as target models. Samples were randomly selected from ImageNet and CIFAR datasets.

compared to the Hybrid Attack. On CIFAR-100 and CIFAR-10, it consistently reaches a success rate of 1.00 with even fewer queries and noise.

3) Grad interpreter with ResNet and DenseNet: Using the Grad interpreter, the proposed attack achieves success rates of 1.00 across CIFAR datasets and 0.95 to 1.00 on ImageNet when targeting the ResNet and DenseNet models. The attack requires fewer queries, with an average ranging from 76.82 to 297.86, and maintains a low noise level of 0.04 ± 0.03 on CIFAR and 0.20 ± 0.06 on ImageNet. These results are better than those of the Hybrid Attack, which requires more queries and higher noise rates.

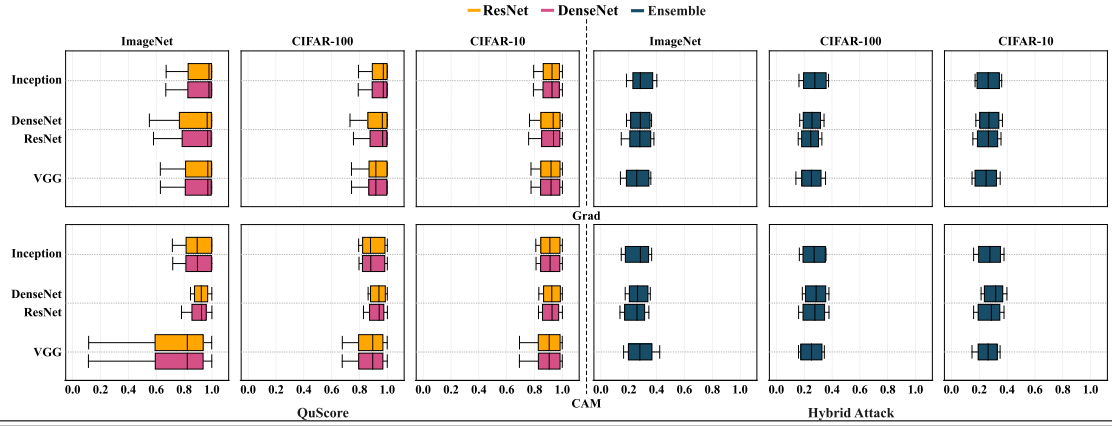
In all cases, the proposed attack achieves high success rates with fewer queries and less noise, indicating a more efficient attack compared to the Hybrid Attack. This is especially significant in the context of more complex datasets, such as ImageNet, where the effectiveness of an attack is crucial for its stealth and success. The consistency in the results of the proposed attack, with a median of 5.00 queries across datasets and interpreters, further indicates its reliability.

Observation 1—Effectiveness against DNNs: The results show that the proposed attack was highly effective against all classifiers and interpreters, achieving success rates typically above 95% and requiring relatively few queries, with an average ranging from 158.33 to 438.24 while maintaining a low average noise rate around 0.20.

C. Attack Effectiveness Against Interpreters

In this part, we explore the effectiveness of our attack in comparing the benign and adversarial interpretations based on the qualitative comparison and the IoU test.

1) Qualitative Comparison: In this comparison, we differentiate whether the attribution map of our adversarial sample is similar to the attribution map of benign input. Fig. 3 illustrates a



Source Model	Target Model	ImageNet			CIFAR-100			CIFAR-10		
		Our Attack (Grad)	Our Attack (CAM)	Hybrid Attack	Our Attack (Grad)	Our Attack (CAM)	Hybrid Attack	Our Attack (Grad)	Our Attack (CAM)	Hybrid Attack
ResNet	InceptionV3	0.85	0.79	0.25	0.80	0.75	0.30	0.85	0.80	0.35
	DenseNet	0.83	0.79	0.20	0.80	0.75	0.25	0.85	0.80	0.30
	VGG	0.77	0.75	0.15	0.75	0.70	0.20	0.80	0.75	0.25
DenseNet	InceptionV3	0.82	0.76	0.25	0.80	0.75	0.30	0.85	0.80	0.35
	ResNet	0.80	0.75	0.20	0.80	0.75	0.25	0.85	0.80	0.30
	VGG	0.78	0.78	0.15	0.75	0.70	0.20	0.80	0.75	0.25

Fig. 4. IoU scores of interpretation maps generated by QuScore and Hybrid Attack [30] using Grad (top row) and CAM (bottom row) as interpreters, and ResNet and DenseNet as source models for ImageNet and CIFAR datasets. The *y*-axis represents the target models. The ensemble model is the combination of ResNet and DenseNet models in the existing attack. The table shows the comparison of IoU results between Our Attack and the Hybrid Attack on various datasets and cross models.

set of examples of ImageNet and CIFAR datasets for both benign and adversarial samples with their attribution maps. Based on selected examples, it can be seen that it is difficult to distinguish between benign and adversarial attribution maps on CAM interpreter. However, as mentioned before, the Grad interpreter is very specific in highlighting the area on the attribution map, which makes it difficult to deceive. Even though it is challenging to mislead the Grad interpreter, our attack succeeds in generating adversarial examples that have very similar attribution maps to attribution maps of benign inputs. We can conclude that our adversarial attribution map is as reliable as the benign attribution map.

2) IoU Test: We evaluate the similarity between benign and adversarial attribution maps using the IoU score. The IoU score is calculated by finding the intersection of the maps, and as it approaches 1, the overlap between the interpretation maps becomes complete, leading to indistinguishability between them. The performance of QuScore and the existing attack on different DNN models with Grad and CAM interpreters is summarized in Fig. 4. In the right part of the figure, the results of the Hybrid Attack are provided. As shown, the boxplots on the right show that the existing attack has low IoU scores, indicating that the attack can be easily detected by the interpreters on different models and datasets. This suggests that the attack can be detectable and ineffective against the interpretation models tested. Based on the result of our approach, the findings showed that the Grad interpreter is more effective than the CAM interpreter in conducting attacks on all datasets, with an average IoU score closer to 1. This is shown in the figure, where the median IoU score is greater than 0.9 in all datasets. For the CAM interpreter, our attack works exceptionally well on ResNet and DenseNet, with well-balanced IoU scores and median scores above 0.9, while stability on VGG and Inception-V3 varies.

Observation 2—Effectiveness against interpreters: The proposed attack succeeds in generating adversarial interpretation maps that are indistinguishable from corresponding benign interpretation maps. The attack shows a significant opaque attack capability with median IoU scores greater than 0.8.

D. Attack Effectiveness Against Defensive IDLSes

This section focuses on evaluating the performance of our attack in the presence of various defense techniques. Specifically, we investigate the impact of five commonly employed preprocessing defense strategies, namely, random resizing and padding (R&P) [31], bit depth reduction [32], median smoothing [33], JPEG compression [32], and adversarial training [34] on the efficacy of our attack. It is important to note that the first four defense techniques do not modify the underlying structure of the model; rather, they alter the input sample. The last technique requires the model to be trained on adversarial examples to increase its robustness. Fig. 5 displays the examples of the generated adversarial samples of the attack with/without defense mechanisms.

To experiment, we selected Inception-V3 and VGG as our target models, as they possess different architectures compared to our source models (ResNet and DenseNet). We present the results of our attack against the aforementioned defenses in Table II, with all experiments performed using the default values of the defense techniques. Fig. 6 showcases the IoU scores of the interpretation maps generated by our attack against five defense

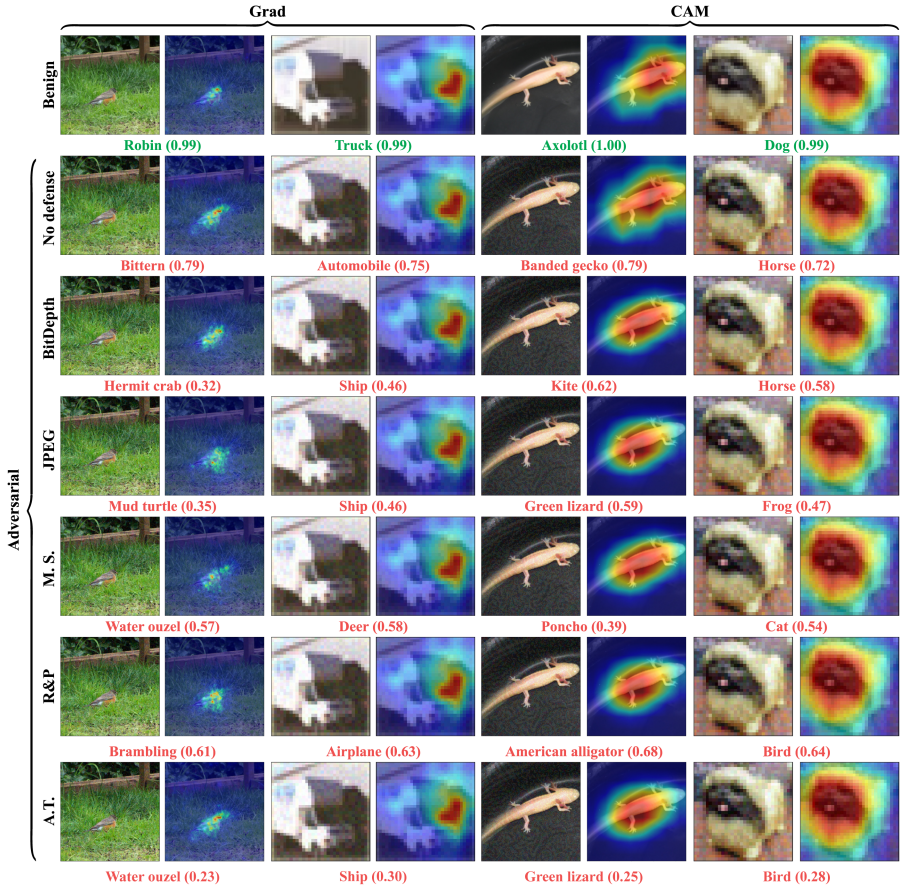


Fig. 5. Adversarial samples with attribution maps generated using QuScore with/without defense techniques using Grad, CAM on ResNet, DenseNet as source, Inception, and VGG as target models. The examples were selected at random from the ImageNet and CIFAR datasets. In the figure, M.S. and A.T. stand for median smoothing and adversarial training, respectively.

TABLE II
SUCCESS RATE, AVERAGE QUERIES, AND AVERAGE NOISE OF QUSCORE AGAINST IDLSES WHEN USING VARIOUS DEFENSES

Interpreter	Source Model	Target Model	R&P			BitDepth			Median Smoothing			JPEG			Adversarial Training		
			Success Rate	Avg. Queries	Avg. Noise Rate	Success Rate	Avg. Queries	Avg.Noise Rate	Success Rate	Avg. Queries	Avg. Noise Rate	Success Rate	Avg. Queries	Avg. Noise Rate	Success Rate	Avg. Queries	Avg. Noise Rate
ImageNet																	
CAM	ResNet	InceptionV3	0.82	188.49	0.22±0.05	0.96	514.87	0.21±0.06	0.96	286.79	0.21±0.06	0.81	427.31	0.22±0.06	0.37	967.15	0.22±0.06
		VGG	0.95	78.85	0.21±0.06	1.00	140.65	0.19±0.06	1.00	118.33	0.21±0.06	0.95	139.02	0.21±0.06	0.40	644.78	0.21±0.06
	DenseNet	InceptionV3	0.87	104.45	0.21±0.06	0.96	441.57	0.21±0.06	0.97	255.73	0.21±0.06	0.86	334.05	0.21±0.06	0.42	993.42	0.21±0.06
		VGG	0.92	77.06	0.20±0.06	0.99	167.78	0.19±0.06	1.00	89.55	0.20±0.06	0.95	163.73	0.21±0.06	0.43	759.39	0.21±0.06
GS	ResNet	InceptionV3	0.82	248.9	0.22±0.06	0.96	509.51	0.20±0.06	0.97	315.49	0.21±0.06	0.82	454.78	0.22±0.06	0.39	987.33	0.22±0.06
		VGG	0.92	152.1	0.21±0.05	0.99	114.25	0.20±0.06	1.00	109.28	0.21±0.06	0.99	133.07	0.20±0.06	0.44	617.18	0.20±0.06
	DenseNet	InceptionV3	0.86	100.53	0.21±0.06	0.96	451.88	0.21±0.06	0.97	248.4	0.21±0.06	0.86	323.8	0.21±0.06	0.41	998.75	0.21±0.06
		VGG	0.92	87.83	0.20±0.06	0.99	142.9	0.20±0.06	1.00	96.49	0.20±0.06	0.99	161.93	0.20±0.06	0.45	751.04	0.20±0.06
CIFAR-100																	
CAM	ResNet	InceptionV3	0.99	86.41	0.04±0.03	0.99	116.05	0.04±0.03	0.98	131.48	0.04±0.03	0.98	135.90	0.04±0.03	0.41	630.31	0.04±0.03
		VGG	0.99	36.15	0.04±0.03	0.99	64.48	0.04±0.03	0.98	54.25	0.04±0.03	0.98	63.73	0.04±0.03	0.46	295.58	0.04±0.03
	DenseNet	InceptionV3	0.98	47.89	0.04±0.03	0.99	102.44	0.04±0.03	0.98	117.24	0.04±0.03	0.97	113.15	0.04±0.03	0.44	524.79	0.04±0.03
		VGG	0.99	35.33	0.04±0.03	0.99	76.92	0.04±0.03	0.98	41.05	0.04±0.03	0.98	75.06	0.04±0.03	0.42	348.13	0.04±0.03
GS	ResNet	InceptionV3	0.99	114.11	0.04±0.03	0.99	133.59	0.04±0.03	0.99	144.64	0.04±0.03	0.97	108.50	0.04±0.03	0.44	503.23	0.04±0.03
		VGG	0.99	69.73	0.04±0.03	0.99	52.38	0.04±0.03	0.99	50.10	0.04±0.03	0.98	61.01	0.04±0.03	0.46	282.97	0.04±0.03
	DenseNet	InceptionV3	0.99	46.09	0.04±0.03	0.99	107.17	0.04±0.03	0.99	113.88	0.04±0.03	0.97	128.45	0.04±0.03	0.41	595.76	0.04±0.03
		VGG	0.99	40.27	0.04±0.03	0.99	65.51	0.04±0.03	0.99	44.24	0.04±0.03	0.98	74.24	0.04±0.03	0.45	344.33	0.04±0.03
CIFAR-10																	
CAM	ResNet	InceptionV3	0.97	119.21	0.04±0.03	0.96	125.61	0.04±0.03	0.97	131.37	0.04±0.03	0.96	170.24	0.04±0.03	0.43	789.58	0.04±0.03
		VGG	0.97	49.87	0.04±0.03	0.97	88.95	0.04±0.03	0.97	74.83	0.04±0.03	0.98	87.92	0.04±0.03	0.43	407.78	0.04±0.03
	DenseNet	InceptionV3	0.96	66.06	0.04±0.03	0.97	139.26	0.04±0.03	0.96	141.73	0.04±0.03	0.97	131.26	0.04±0.03	0.46	608.79	0.04±0.03
		VGG	0.98	48.73	0.04±0.03	0.96	106.11	0.04±0.03	0.97	56.63	0.04±0.03	0.97	103.55	0.04±0.03	0.41	480.27	0.04±0.03
GS	ResNet	InceptionV3	0.98	157.41	0.04±0.03	0.97	122.22	0.04±0.03	0.96	139.52	0.04±0.03	0.97	187.61	0.04±0.03	0.46	870.14	0.04±0.03
		VGG	0.98	96.19	0.04±0.03	0.96	72.25	0.04±0.03	0.97	69.11	0.04±0.03	0.97	84.16	0.04±0.03	0.44	390.34	0.04±0.03
	DenseNet	InceptionV3	0.97	63.58	0.04±0.03	0.97	125.78	0.04±0.03	0.97	137.09	0.04±0.03	0.96	144.78	0.04±0.03	0.45	671.49	0.04±0.03
		VGG	0.98	55.55	0.04±0.03	0.96	90.37	0.04±0.03	0.98	61.02	0.04±0.03	0.97	102.41	0.04±0.03	0.46	474.98	0.04±0.03

The results are based on 3000 images using ImageNet and CIFAR Datasets. The median of queries against all defenses is 5.

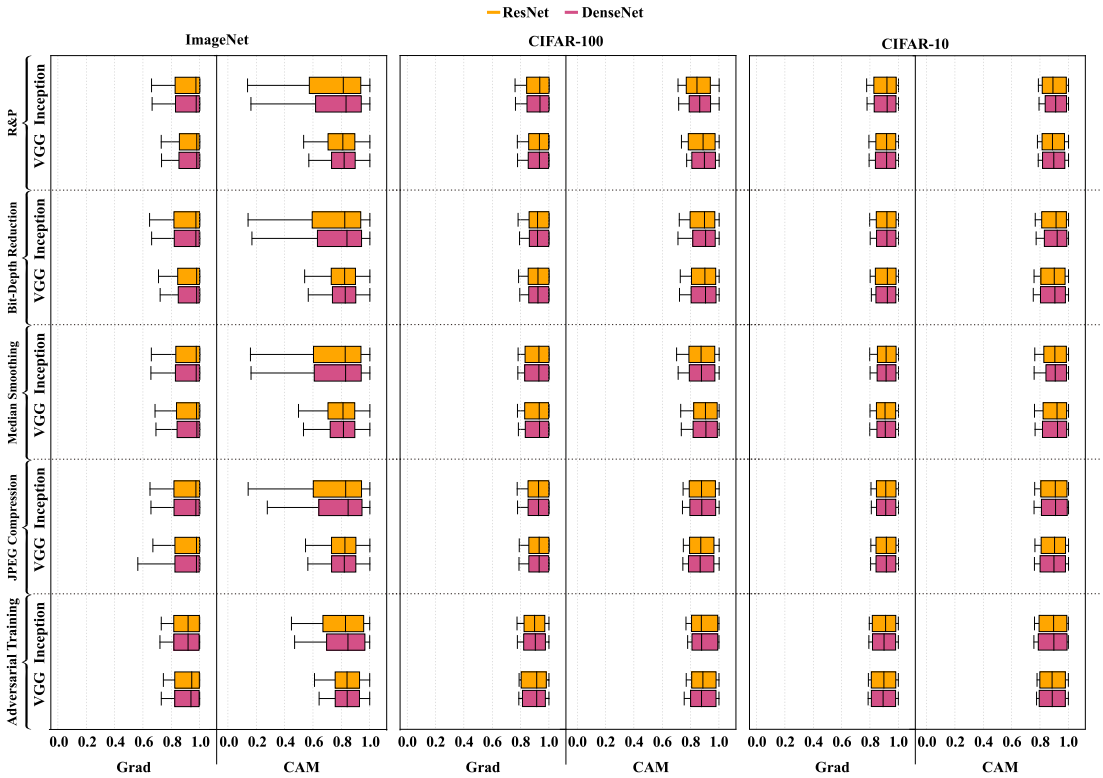


Fig. 6. IoU scores of interpretation maps generated by QuScore against five different defenses using Grad and CAM interpreters with Inception-V3 and VGG as target models and ResNet and DenseNet as source models on ImageNet and CIFAR datasets.

techniques, using Grad and CAM interpreters, and ResNet and DenseNet as source models, InceptionV3 and VGG as target models, in the ImageNet, CIFAR-100, and CIFAR-10 datasets, respectively.

1) R&P: Comparing the results of the datasets, we can see that in general, the success rates of the attack are lower on the ImageNet dataset compared to the CIFAR-100 and CIFAR-10 datasets. For example, the success rates on the ImageNet dataset range from 0.82 to 0.95, while on the CIFAR datasets, the success rates range from 0.96 to 0.99. Furthermore, we can observe that the average number of queries required to successfully execute the attack is higher on the ImageNet dataset (ranging from 77.06 to 248.9) than on the CIFAR datasets (ranging from 35.33 to 157.41). The lower success rates and the higher number of queries on the ImageNet dataset than CIFAR datasets can be attributed to a larger size, greater image diversity of the dataset and its complexity, and larger search space to generate effective adversarial perturbations.

In terms of IoU score, we observe that on the Grad interpreter, our attack performs significantly well, with consistently high IoU scores across both target models. When CAM interpreter is used, we observe that our attack's performance on Inception-V3 with ImageNet dataset is somewhat unstable, whereas the results obtained on the CIFAR dataset are more reliable. The difference can be attributed to each interpreter's interpretability and robustness characteristics.

2) Bit-Depth Reduction: When this defense is used, it takes more queries for the attack to produce effective adversarial samples. This suggests that the defense technique is more robust than the other defenses. However, the success rate of the attack is high, ranging from 0.96 to 1.00 and the average queries required to succeed in the attack range from 52.38 to 514.87, with the lowest average queries required for the ResNet model with the Grad interpreter against VGG and the highest average queries required for the ResNet model with the CAM interpreter against Inception-V3. The attack with this defense showed a similar trend as R&P in performance based on the IoU score. It performed well in the Grad interpreter, consistently achieving high scores in all target models. And the attack shows instability in the CAM interpreter with Inception-V3 and the ImageNet dataset.

3) Median Smoothing: Regarding the success rate, the attack has a high success rate for all datasets ranging from 0.96 to 1.00. Regarding the average queries, the attack requires more queries for the ImageNet dataset than for the CIFAR-100 and CIFAR-10 datasets. For ImageNet, the average queries range from 89.55 to 315.49, while for CIFAR-100 and CIFAR-10 datasets, the average queries range from 41.05 to 144.64 and 56.63 to 141.73, respectively. As for the median queries, the attack requires only five queries for all datasets and target models. Compared to previous defense methods, the technique required more queries on both datasets. This indicates that it effectively preprocesses adversarial samples. Furthermore, when this defense is used, the attack demonstrates IoU performance comparable to previous defenses. Although it requires more queries, the attack still achieves similar IoU scores.

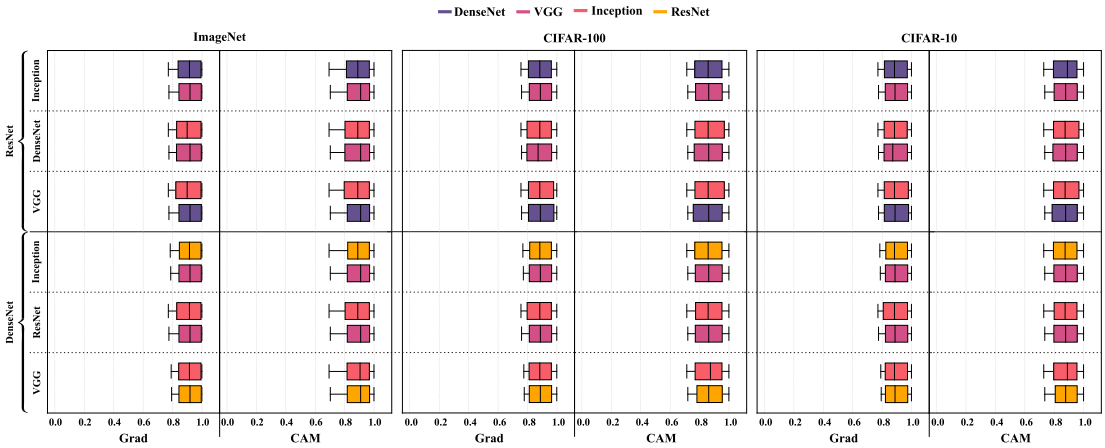


Fig. 7. IoU scores of interpretation maps generated by the transferred models. The adversarial samples, originally intended to compromise given models, are transferred to alternative models using the ImageNet and CIFAR datasets.

4) JPEG Compression: The ASR varies between 0.81 and 0.99, with higher success rates achieved on the CIFAR dataset than on the ImageNet dataset. The average queries required for successful attacks ranged from 61.01 to 454.78, with the median queries being five for all combinations, indicating that the attack can successfully break the defenses with a small number of queries. Note that most defense techniques are designed to reduce the noise in adversarial examples. However, the average noise rate remains unchanged, around 0.21 ± 0.06 and 0.04 ± 0.03. As depicted in Fig. 5, the added perturbation in the samples is difficult to detect visually, even with the implementation of defenses.

5) Adversarial Training: Adversarial training has proven to be a robust defense mechanism against adversarial attacks. To illustrate its effectiveness, we report the performance of adversarially trained models on ImageNet when evaluated using AutoAttack [35]. The VGG19 model achieves a clean accuracy of 60.1% and an AutoAttack robustness of 20.5%. The Inception-V3 model demonstrates better performance, with a clean accuracy of 67.3% and AutoAttack robustness of 30.2%. Table II shows the results of those models against our attack. The success rate of the attack against models with adversarial training varies notably, indicating its effectiveness as a defense. Success rates range from 0.37 to 0.46, significantly lower than other defenses. This reduction highlights the robustness provided by adversarial training. The attack requires more queries, with averages from 295.58 to 998.75, suggesting the attack faces more challenges against this defense. Despite this, the average noise rate remains fairly consistent, around 0.20 ± 0.06 and 0.04 ± 0.03, similar to other defenses. This implies that while adversarial training is effective in reducing success rates, it does not necessarily lead to a noticeable change in the noise in adversarial examples.

Observation 3—Effectiveness against Defensive IDLSes: The proposed attack generally maintains resilience against various defenses, demonstrating its ability to overcome enhanced security measures in most scenarios.

TABLE III
SUCCESS RATE OF ADVERSARIAL SAMPLES IN TERMS OF ATTACK TRANSFERABILITY

Interpreter	Source Model	Target Model	Transferred Model	ASR		
				ImageNet	CIFAR-100	CIFAR-10
CAM	ResNet	InceptionV3	DenseNet	0.75	0.72	0.71
			VGG	0.81	0.85	0.75
		DenseNet	InceptionV3	0.63	0.69	0.62
			VGG	0.76	0.71	0.66
		VGG	InceptionV3	0.65	0.64	0.61
			DenseNet	0.67	0.66	0.62
	DenseNet	InceptionV3	ResNet	0.74	0.73	0.68
			VGG	0.81	0.85	0.81
		ResNet	InceptionV3	0.69	0.75	0.69
			VGG	0.71	0.68	0.59
		VGG	InceptionV3	0.62	0.58	0.54
			ResNet	0.69	0.63	0.58
GS	ResNet	InceptionV3	DenseNet	0.83	0.74	0.64
			VGG	0.79	0.76	0.71
		DenseNet	InceptionV3	0.71	0.71	0.60
			VGG	0.73	0.73	0.64
		VGG	InceptionV3	0.65	0.57	0.56
			DenseNet	0.69	0.63	0.55
	DenseNet	InceptionV3	ResNet	0.75	0.79	0.71
			VGG	0.83	0.82	0.82
		ResNet	InceptionV3	0.65	0.63	0.48
			VGG	0.72	0.69	0.63
		VGG	InceptionV3	0.62	0.64	0.57
			ResNet	0.65	0.69	0.59

Adversarial samples generated against specific DNN models are transferred to alternative DNN models on ImageNet, CIFAR-100, and CIFAR-10 Datasets.

E. Attack Transferability

Table III provides the success rate of adversarial samples generated against specific DNN models as target models using source models, which are then transferred to alternative DNN models on ImageNet, CIFAR-100, and CIFAR-10 datasets. The results showed that the proposed attack method exhibits high transferability across different datasets with a high success rate. Specifically, the ASR in transferability achieved over 50% across all datasets, indicating the effectiveness of the proposed method in generating adversarial examples that can fool different models trained on different datasets. Fig. 7 shows the IoU scores of the interpretation maps of adversarial samples regarding transferability. The scores indicate the similarity between the interpretation maps of adversarial samples and their benign

TABLE IV
AVERAGE CLASSIFICATION CONFIDENCE SCORES OF THE PROPOSED ATTACK AGAINST DIFFERENT CLASSIFIERS AND INTERPRETERS TESTING ON 1000 IMAGES FOR EACH DATASET (TOTAL 3000 IMAGES)

Source Model	Target Model	ImageNet		CIFAR-100		CIFAR-10	
		CAM	Grad	CAM	Grad	CAM	Grad
ResNet	InceptionV3	0.46	0.45	0.53	0.51	0.55	0.51
	DenseNet	0.52	0.49	0.56	0.54	0.58	0.56
	VGG	0.41	0.42	0.49	0.50	0.51	0.52
DenseNet	InceptionV3	0.42	0.43	0.51	0.52	0.53	0.54
	ResNet	0.55	0.54	0.57	0.55	0.58	0.57
	VGG	0.44	0.43	0.48	0.49	0.50	0.51

TABLE V
COMPARISON OF QUSCORE WITH BASES ON 1000 IMAGES

Attacks	Source Model	Target Model	ASR	Query	Query limit	Median Query	IoU
BASES	ResNet50 + DenseNet169	InceptionV3	0.89	**2.81 ± 2.74**	500	**2.00**	0.33
		VGG	0.91	**2.11 ± 2.48**	500	**2.00**	0.35
Ours		InceptionV3	**0.94**	47.96 ± 98.45	500	5.00	**0.89**
		VGG	**0.93**	25.61 ± 54.75	500	5.00	**0.84**
BASES	**20 models**	InceptionV3	0.99	1.24 ± 2.45	-	2.00	0.35
		VGG	1.00	1.21 ± 2.62	-	1.00	0.38

IoU scores are based on the CAM interpreter. The top rows present a benchmark setup for comparison with QuScore, while the last row represents the original setup of BASES.

ones. The results indicate that the attack transferability across all datasets and models is high. Specifically, the IoU scores of the interpretation maps of adversarial samples are similar to those of their benign counterparts. This indicates that adversarial samples are highly transferable across DNN models.

Observation 4—Attack Transferability: The proposed attack shows high transferability across various DNN models while maintaining high similarity in interpretation maps using different datasets.

V. DISCUSSION

A. Additional Experiments and Comparisons

1) Attack Comparison with BASES [36]: To provide a comprehensive evaluation of our proposed attack method, we compared it with the recent BASES attack method. For a fair comparison, we used the same 1000 images for both attacks and maintained the same number of surrogate models. The experiments were conducted using ResNet50 and DenseNet169 as source models, targeting InceptionV3 and VGG models. We set a query limit of 500, as the BASES attack does not require many queries [36]. The results are summarized in Table V.

The result show that QuScore achieved a higher success rate of 0.94 and 0.93 against InceptionV3 and VGG, respectively, compared to 0.89 and 0.91 for BASES. While BASES demonstrated a lower average query count (2.81 ± 2.74 for InceptionV3 and 2.11 ± 2.48 for VGG) than QuScore (47.96 ± 98.45 and 25.61 ± 54.75, respectively), QuScore significantly outperformed in IoU scores, achieving 0.89 and 0.84 compared to 0.33 and 0.35 for BASES. The median query count for QuScore was 5.00 for both target models, slightly higher than the 2.00 median for BASES.

We note that the BASES [36] heavily relies on the number of surrogate models. In the original BASES paper [36], the authors used 20 surrogate models to achieve high ASR. For further evaluation, we also conducted experiments with BASES using 20 surrogate models on the same images, targeting InceptionV3 and VGG. Under these conditions, BASES achieved untargeted ASR of 99% and 100%, respectively. However, in the comparison with BASES, we limited the number of surrogate models to 2 to highlight the robustness of methods under constrained settings. QuScore achieved higher IoU scores and comparable or better success rates, emphasizing its efficiency and effectiveness even with restricted resources such as the number of surrogate models and query limits.

TABLE VI
PERFORMANCE OF QUSCORE AGAINST ROBUSTBENCH MODELS ON 1000 IMAGES.

Source Model	Target Model	ASR	Query	Query limit	Median Query	IoU
DenseNet169	ResNet50 [38]	0.75	685.27 ± 412.55	1000	5.00	0.85
	WideResNet50 [39]	0.93	565.55 ± 490.12	1000	5.00	0.88

IoU scores are based on CAM.

TABLE VII
RESULTS OF QUSCORE WITH $\epsilon = \frac{2}{255}$ (1000 IMAGES)

Attacks	Source Model	Target Model	ASR	Query	Query limit	Median Query	IoU
Ours	ResNet50	InceptionV3	0.95	632.94	5000	5.00	0.82

2) Attack Against Robustbench Models [37]: Table VI provides the performance of QuScore against ResNet50 and WideResNet50, models sourced from the Robustbench table, using 1000 images. To calculate the IoU score, we used CAM interpreter in the experiment. Based on Robustbench, ResNet50 [38] has a clean accuracy of 62.56% and robust accuracy of 29.22% while WideResNet50 [39] achieves a higher clean accuracy of 68.76% and robust accuracy of 40.60% on the ImageNet dataset. Against these models, QuScore achieves ASR of 75% and 93%, respectively, with low median query counts of 5. The average query counts are 685.27 for ResNet50 and 565.55 for WideResNet50, highlighting QuScore 's efficiency even under a query limit of 1000. The 1000 query limit was chosen as these models are adversarially trained to defend against attacks, making them harder to bypass and requiring careful evaluation within a reasonable query budget. The high IoU values of 0.85 for ResNet50 and 0.88 for WideResNet50 indicate that the adversarial examples closely resemble the original inputs, maintaining perceptual similarity.

3) Experiments with Lower Perturbation ϵ: To further evaluate the robustness and effectiveness of our proposed attack method under more strict conditions, we conducted additional experiments using $\epsilon = 4/255$. The experiment utilized ResNet50 as the source model and InceptionV3 as the target model.

The results in Table VII indicate that with a lower ϵ value of 4/255, our method achieved an ASR of 0.95. The average number of queries required was 632.94, with a query limit set at 5000. The median query count was 5.00, and the IoU score was 0.82. These results demonstrate that our method remains highly effective even under stricter conditions, maintaining a high

success rate and producing high-quality adversarial examples with a relatively low median query count.

B. Limitations

Although our evaluation in Section IV demonstrates the effectiveness of QuScore in a range of classifiers and interpretation models, it is important to recognize its limitations and potential countermeasures. We discuss these factors in the following.

During the initial development of QuScore, the attack was designed to target DNN models solely on the basis of the success rate, disregarding the confidence level of the model's misclassification. While the attack achieved a high success rate in fooling the target models (as shown in Table I), this is done with low misclassification confidence, as shown in Table IV. This is common in opaque settings, as the objective function focuses on shifting the decision of adversarial samples to other classes, which naturally results in low misclassification confidence scores. The issue can be addressed by changing the objective function to target higher misclassification confidence. This means optimizing the attack to maximize the confidence of the adversarial class, which can lead to higher misclassification confidence scores.

Another limitation of QuScore is that the number of queries required to perform the attack increases with the complexity of the target DNN model. As a result, there are cases where QuScore fails to fool more complex architectures. For example, Inception-V3 requires more queries and has a lower success rate than the other models. One possible solution to address the limitation can be optimizing the transfer-based aspect of the attack and adjusting key hyperparameters such as the perturbation threshold and maximum query count. In addition, the use of more complex source models may help mitigate the limitation. Although our attack requires slightly more queries than existing approaches [11], [12] (e.g., 98 and 130 queries for VGG and Inception-V3, respectively), the difference is not significant considering the constraints on the adversarial search space of our attack.

C. Potential Countermeasures and Future Work

Based on the illustration in Fig. 3, adversarial samples generated by QuScore provide high-quality interpretations that are indistinguishable from their benign interpretations. Another countermeasure is to train a DNN model with an adversarial dataset to increase its robustness (see Table II). It becomes more difficult to fool a robust DNN model in a opaque setting when adversarial training is used [40], [41]. In this case, an attacker needs to increase the perturbation rate to fool the adversarially trained DNN model, however, as the perturbation rate is high, it could reveal the attacker's presence. Adopting several defense techniques [42] at the same time as an ensemble-based method (i.e., preprocessing techniques) for the decision-making process could be another countermeasure against the attack. The main reason for this is that defense techniques utilize various features of a sample to remove added perturbation. Also, optimally generating an adversarial sample considering all the features of a sample becomes computationally expensive.

Future Work: While our proposed attack demonstrated its efficacy against various DNN models, future research can explore other DNN/transformer-based models with different architectures. Another potential direction is to aim for high ASR with high misclassification scores. Furthermore, future work could investigate other types of interpreters, such as RTS [23], SHAP [43], and MASK [24].

VI. Conclusion

In this work, we propose the opaque version of the AdvEdge attack, which can deceive both DNN models and their interpreters in a opaque setting. Our attack is based on transfer-based and score-based methods and is both gradient-free and query-efficient. Our experimental results demonstrate that our method achieves a high ASR and generates adversarial interpretation maps that are highly similar to benign interpretations. Moreover, our attack is effective against various defense techniques, even when they are involved in the process. We achieve a high ASR and transferability while still misleading the target interpreters in most cases, highlighting the efficacy and robustness of our proposed attack. These findings emphasize the need to develop more robust defense mechanisms to enhance the security of DNN models against adversarial attacks.

References

[1] Y. Lin, H. Zhao, X. Ma, Y. Tu, and M. Wang, "Adversarial attacks in modulation recognition with convolutional neural networks," *IEEE Trans. Rel.*, vol. 70, no. 1, pp. 389–401, Mar. 2021.

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

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

[4] C. Xiao, J. Zhu, B. Li, W. He, M. Liu, and D. Song, "Spatially transformed adversarial examples," in *Proc. 6th Int. Conf. Learn. Representations*, 2018.

[5] X. Zhang, N. Wang, H. Shen, S. Ji, X. Luo, and T. Wang, "Interpretable deep learning under fire," in *Proc. 29th USENIX Secur. Symp.*, 2020, pp. 1659–1676.

[6] E. Abdukhamidov, M. Abuhamad, F. Juraev, E. Chan-Tin, and T. AbuHmed, "Advedge: Optimizing adversarial perturbations against interpretable deep learning," in *Proc. Int. Conf. Comput. Data Social Netw.*, 2021, pp. 93–105.

[7] E. Abdukhamidov, M. Abuhamad, S. S. Woo, E. Chan-Tin, and T. Abuhmed, "Hardening interpretable deep learning systems: Investigating adversarial threats and defenses," *IEEE Trans. Dependable Secure Comput.*, vol. 21, no. 4, pp. 3963–3976, Jul./Aug. 2024.

[8] Y. Dong, T. Pang, H. Su, and J. Zhu, "Evading defenses to transferable adversarial examples by translation-invariant attacks," in *Proc. CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4312–4321.

[9] Q. Huang, I. Katsman, H. He, Z. Gu, S. Belongie, and S.-N. Lim, "Enhancing adversarial example transferability with an intermediate level attack," in *Proc. CVF Int. Conf. Comput. Vis.*, 2019, pp. 4733–4742.

[10] C. Szegedy et al., "Intriguing properties of neural networks," in *Proc. 2nd Int. Conf. Learn. Representations*, 2014.

[11] L. Wang, K. Yang, W. Wang, R. Wang, and A. Ye, "Mgaattack: Toward more query-efficient black-box attack by microbial genetic algorithm," in *Proc. 28th ACM Int. Conf. Multimedia*, 2020, pp. 2229–2236.

[12] M. Alzantot, Y. Sharma, S. Chakraborty, H. Zhang, C.-J. Hsieh, and M.B. Srivastava, "Genattack: Practical black-box attacks with gradient-free optimization," in *Proc. Genet. Evol. Comput. Conf.*, 2019, pp. 1111–1119.

[13] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2818–2826.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[15] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4700–4708.
[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[17] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2921–2929.
[18] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *Proc. Int. Conf. Learn. Representations*, 2014.
[19] P. Qi, T. Jiang, L. Wang, X. Yuan, and Z. Li, "Detection tolerant black-box adversarial attack against automatic modulation classification with deep learning," *IEEE Trans. Rel.*, vol. 71, no. 2, pp. 674–686, Jun. 2022.
[20] C. Li, W. Yao, H. Wang, and T. Jiang, "Adaptive momentum variance for attention-guided sparse adversarial attacks," *Pattern Recognit.*, vol. 133, 2023, Art. no. 108979. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320322004599
[21] E. Abdukhamidov, M. Abuhamad, G. K. Thiruvathukal, H. Kim, and T. Abuhmed, "SingleADV: Single-class target-specific attack against interpretable deep learning systems," *IEEE Trans. Inf. Forensics Secur.*, vol. 19, pp. 5985–5998, May 2024.
[22] G. Xu et al., "ASQ-FastBM3D: An adaptive denoising framework for defending adversarial attacks in machine learning enabled systems," *IEEE Trans. Rel.*, vol. 72, no. 1, pp. 317–328, Mar. 2023.
[23] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6970–6979.
[24] R. C. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3429–3437.
[25] A. Karpathy, J. Johnson, and L. Fei-Fei, "Visualizing and understanding recurrent networks," 2015, *arXiv:1506.02078*.
[26] W. J. Murdoch, P. J. Liu, and B. Yu, "Beyond word importance: Contextual decomposition to extract interactions from LSTMs," in *Proc. Int. Conf. Learn. Representations*, 2018.
[27] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. London, U.K.: Oxford Univ. Press, 1996.
[28] D. Corus, J. He, T. Jansen, P. S. Oliveto, D. Sudholt, and C. Zarges, "On easiest functions for mutation operators in bio-inspired optimisation," *Algorithmica*, vol. 78, no. 2, pp. 714–740, 2017.
[29] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[30] F. Suya, J. Chi, D. Evans, and Y. Tian, "Hybrid batch attacks: Finding black-box adversarial examples with limited queries," in *Proc. 29th USENIX Secur. Symp.*, 2020, pp. 1327–1344.
[31] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," in *Proc. Int. Conf. Learn. Representations*, 2018.
[32] C. Guo, M. Rana, M. Cissé, and L. van der Maaten, "Countering adversarial images using input transformations," in *Proc. 6th Int. Conf. Learn. Representations*, 2018.
[33] G. W. Ding, L. Wang, and X. Jin, "AdverTorch v0.1: An adversarial robustness toolbox based on Pytorch," 2019, *arXiv:1902.07623*.
[34] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *Stat*, vol. 1050, 2017, Art. no. 9.
[35] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 2206–2216.
[36] Z. Cai, C. Song, S. Krishnamurthy, A. Roy-Chowdhury, and S. Asif, "Blackbox attacks via surrogate ensemble search," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 5348–5362.
[37] F. Croce et al., "Robustbench: A standardized adversarial robustness benchmark," 2020, *arXiv:2010.09670*.
[38] L. Engstrom, A. Ilyas, S. Santurkar, D. Tsipras, B. Tran, and A. Madry, "Adversarial robustness as a prior for learned representations," 2019, *arXiv:1906.00945*.
[39] E.-C. Chen and C.-R. Lee, "Data filtering for efficient adversarial training," *Pattern Recognit.*, vol. 151, 2024, Art. no. 110394.
[40] A. Shafahi et al., "Adversarial training for free!," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 3358–3369.
[41] L. Hsiung, Y.-Y. Tsai, P.-Y. Chen, and T.-Y. Ho, "Towards compositional adversarial robustness: Generalizing adversarial training to composite semantic perturbations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 24658–24667.
[42] W. Wei et al., "Cross-layer strategic ensemble defense against adversarial examples," in *Proc. 2020 Int. Conf. Comput. Netw. Commun.*, 2020, pp. 456–460.
[43] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should i trust you?" Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1135–1144.