Single**ADV**: Single-Class Target-Specific Attack Against Interpretable Deep Learning Systems

Eldor Abdukhamidov, Mohammed Abuhamad, *Senior Member, IEEE*, George K. Thiruvathukal, *Senior Member, IEEE*, Hyoungshick Kim, and Tamer Abuhmed, *Member, IEEE*

***Abstract*— Establishing trust and helping experts debug and understand the inner workings of deep learning models, interpretation methods are increasingly coupled with these models, building interpretable deep learning systems. However, adversarial attacks pose a significant threat to public trust by making interpretations of deep learning models confusing and difficult to understand. In this paper, we present a novel Single-class target-specific ADVersarial attack called SingleADV. The goal of SingleADV is to generate a universal perturbation that deceives the target model into confusing a specific category of objects with a target category while ensuring highly relevant and accurate interpretations. The universal perturbation is stochastically and iteratively optimized by minimizing the adversarial loss that is designed to consider both the classifier and interpreter costs in targeted and non-targeted categories. In this optimization framework, ruled by the first- and second-moment estimations, the desired loss surface promotes high confidence and interpretation scores of adversarial samples. By avoiding unintended misclassification of samples from other categories, SingleADV enables more effective targeted attacks on interpretable deep learning systems in both white-box and black-box scenarios. To evaluate the effectiveness of SingleADV, we conduct experiments using four different model architectures (ResNet-50, VGG-16, DenseNet-169, and Inception-V3) coupled with three interpretation models (CAM, Grad, and MASK). Through extensive empirical evaluation, we demonstrate that SingleADV effectively deceives target deep learning models and their associated interpreters under various conditions and settings. Our results show that the performance of SingleADV is effective, with an average attack success rate of 74% and prediction confidence exceeding 77% on successful adversarial samples. Furthermore, we discuss several countermeasures against SingleADV, including a transfer-based learning approach and existing preprocessing defenses.**

***Index Terms*— Convolutional neural networks, interpretation models, adversarial attack, adversarial perturbation, images.**

I. Introduction

DEEP learning (DL) has made significant contributions and advancements across various domains, including computer vision [1], [2], [3], natural language processing, and numerous security-sensitive applications [4], [5]. The impressive performance of deep learning models on large datasets has gained significant attention from the research community. However, a fundamental challenge lies in comprehending the underlying factors that drive the outcomes of DL models, primarily due to their complex architectures. Consequently, converting the behavior of DL models into a more comprehensible format for end-users has become crucial. To address this issue, numerous interpretation techniques [1], [5], [6], [7] have been developed to make DL models more understandable. These models (*i.e.,* interpreters) provide insights into how DL models make decisions, which can help users trust and use these models more effectively. These insights include ❶ *feature importance* (*i.e.,* identifying parts of the input that are most significant for the model's predictions; ❷ *heatmaps and visualizations* (*i.e.,* offering visual representations of the model's focus areas, particularly in image processing tasks).

The combination of interpretability and prediction models, known as Interpretable Deep Learning Systems (IDLSes), is essential to understand the behavior of models and ensure trustworthiness in detecting adversarial input. IDLSes have become increasingly popular as they offer both predictions and interpretations. However, recent studies have shown that it is possible to create Adversarial Examples (AEs) that can deceive both the target model and its coupled interpreters [8], [9], [10], [11].

We introduce Single**ADV**, a single-class target-specific adversarial attack method designed to generate targeted perturbations that deceive IDLSes by causing misclassifications of an entire class of objects (referred to as the "source class") into a specific "target class." The perturbations are designed to maintain interpretations similar to those of benign inputs, making it difficult for IDLSes to detect the attack. In a targeted attack threat model, Single**ADV** generates stealthy perturbations that effectively deceive IDLSes, therefore hindering human involvement in analyzing the interpretation of

Manuscript received 7 June 2023; revised 3 January 2024 and 24 May 2024; accepted 26 May 2024. Date of publication 30 May 2024; date of current version 11 June 2024. This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea Government [Ministry of Science and Communications Technology (MSIT)] under Grant 2021R1A2C1011198, in part by the Institute for Information and Communications Technology Planning and 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 Artificial intelligence (AI) Platform to Fully Adapt and Reflect Privacy-Policy Changes under Grant 2022-0-00688, and in part by Convergence Security Core Talent Training Business under Grant 2022-0-01199. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Luisa Verdoliva. *(Corresponding author: Tamer Abuhmed.)*

Eldor Abdukhamidov, Hyoungshick Kim, and Tamer Abuhmed are with the Department of Computer Science and Engineering, Sungkyunkwan University, Suwon 16419, South Korea (e-mail: abdukhamidov@skku.edu; hyoung@skku.edu; tamer@skku.edu).

Mohammed Abuhamad and George K. Thiruvathukal are with the Department of Computer Science, Loyola University at Chicago, Chicago, IL 60611 USA (e-mail: mabuhamad@luc.edu; gthiruvathukal@luc.edu).

Digital Object Identifier 10.1109/TIFS.2024.3407652

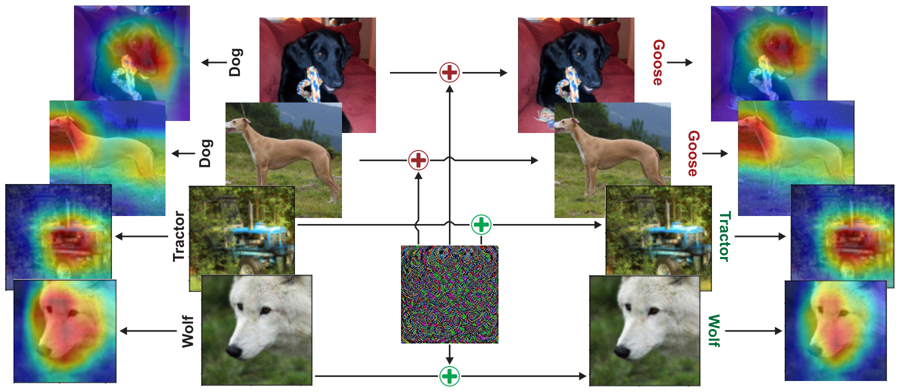

Fig. 1. Adversarial examples that are generated using the ResNet-50 model with the CAM interpreter. The original image is a dog, but the adversarial example is misclassified as a goose. The other class images (tractor and wolf) are not affected. The attacker achieves this by adding a single perturbation to the original image. The perturbations are carefully chosen to have a minimal impact on the interpretation of the image, yet still cause the model to make inaccurate classification.

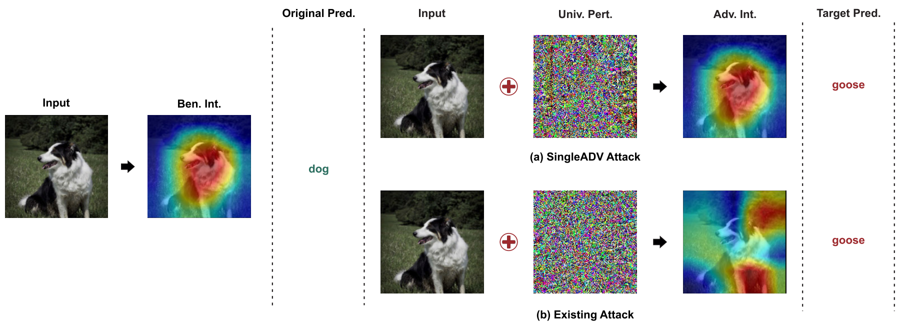

Fig. 2. Single**ADV** approach vs. the existing attack [10] against ResNet-50 with CAM interpreter. Single**ADV** preserves benign attribution maps. Pred., Univ. Pert. and Adv. Int. stand for prediction, universal perturbation, and adversarial interpretation.

adversarial inputs, as shown in Figure 1. Figure 2 demonstrates that the employed interpreter can detect previous attacks, even in white-box scenarios. For example, Figure 1 shows that existing universal perturbation attacks [10] can be detected due to inconsistencies between adversarial interpretations and the object of the image, which can be recognized by an "observer." On the other hand, our approach generates adversarial interpretations that are indistinguishable from benign ones, suggesting no manipulation of the input data.

The main objective of Single**ADV** is to increase the success rate of adversarial attacks. This involves fooling the classifier and misleading its interpreter, while reducing the likelihood of adversarial detection. Single**ADV** achieves this by generating perturbations in a fine-grained manner. This ensures that the perturbations significantly impact the target category while minimizing their influence on other categories. Instead of employing the traditional targeted attack approach, where the model predicts the same label for all images, this work focuses on a single class in an adversarial scenario to reduce suspicion of an attack. We call this attack "*single-class attack*." Although we apply this technique to an image classification task in this paper, it can be used in various security-sensitive applications, such as malware detection and facial recognition systems. Additionally, the generated adversarial samples cause the interpreter to produce false positives, resulting in interpretations similar to benign inputs. This aspect makes it difficult to identify the involvement of the adversary. This work investigates the effects of generating universal perturbations to launch a single-class attack in both white-box and black-box scenarios. Furthermore, it explores potential countermeasures and defenses against such attacks.

A. Our Contribution

We present Single**ADV**, a single-class attack method designed to generate target-specific perturbations for inputs to fool the target deep learning models and deceive their coupled interpreters. Our method enables targeted attacks that are specific to a particular category. We evaluate the effectiveness of Single**ADV** on both the prediction and interpretation models using the ImageNet dataset [1]. Our contributions can be summarized as follows:

- We propose a novel adversarial attack called Single**ADV**, which leverages interpretation-derived techniques to perform targeted and category-specific fooling of DL models and their associated interpreters. Unlike traditional approaches focusing on individual input samples, our method extends the scope of adversarial effects to encompass an entire object category, thus limiting the impact within the chosen category.
- We evaluate Single**ADV** in terms of the success rate for fooling the prediction model, the Intersection-over-Union (IoU) score for deceiving the coupled interpreter, and the leakage rate to measure the attack's impact on non-targeted classes. We demonstrate the effectiveness of our approach, *e.g.,* an average fooling ratio of 0.74 and a corresponding adversarial confidence level of 0.78.

• We conducted experiments using a knowledge distillation (teacher-student) approach to assess the practicality and effectiveness of Single**ADV** in a black-box setting. Our experiments confirm that Single**ADV** can be applied effectively in the black-box scenario, highlighting its practicality.
• We analyze the effectiveness of existing general defense techniques against Single**ADV**. Furthermore, we propose a novel adversarial training method that utilizes interpretation-derived information to enhance the robustness of DL models against adversarial attacks.

B. Organization

The rest of the paper is organized as follows: Section II highlights the relevant literature; Section III provides the problem formulation and the main algorithm; Section IV and Section V provide the experiments and results in white-box and black-box settings, respectively; Section VI proposes potential countermeasures; Section VII discusses the limitations; and Section VIII offers the conclusion.

II. Related Work

This section highlights previous studies related to our work, specifically in the domains of adversarial attacks and interpretation-guided attacks. Machine learning models face two primary threats: evasion attacks [4], which involve manipulating the model's behavior through data manipulation, and poisoning attacks [4], which weaken the target model by infecting the training data. This work focuses on the first type of threat in which we manipulate data to make a DL model misbehave while ensuring the preservation of correct interpretation. Unlike existing approaches, we specifically consider targeted attacks where the perturbations affect a specific class without influencing other classes. Our work is one of the pioneering studies exploring targeted attacks against DL models using interpretability via a universal perturbation in both white-box and black-box settings. In the following, we highlight the related studies that are relevant to our work. Moosavi-Dezfooli et al. [12] highlighted the existence of Universal Adversarial Perturbations (UAPs) capable of misleading deep neural networks, yet these perturbations often appear as noise-like patterns to the human eye. Akhtar et al. [10] explored extended universal perturbations to exploit the explainability of the model. Recent advances have further refined these concepts: Stochastic Gradient Aggregation addresses gradient vanishing and local optima problems in UAP generation [17]; Feature-Gathering UAP leverages *neural collapse* phenomena for more effective attacks [18]; ensemble attack methodologies have evolved to overcome bias issues in CNNs, enhancing UAP transferability [16]; the HardBeat method introduces a novel approach to generate adversarial patches with limited information, showing significant effectiveness [19]; and the Sigma-UAP method improves the imperceptibility of perturbations by manipulating image frequency regions [20]. Our work aims to build a similar attack, focusing on misleading interpretability alongside classification to increase the robustness and stealthiness of attacks, making them more resilient against defenses and interpretative methods.

A. Attacking Interpretability

Recent studies show that some interpretation models are detached from DL models, and they can be impacted by manipulations without affecting the DL model performance [5], [15], [22]. Other studies have demonstrated the validity and practicality of simultaneously attacking the deep learning models and their corresponding interpreters [9]. Abdukhamidov et al. [23] introduce an optimized attack to fool IDLSes with limited perturbation using the edge information of the image in white-box settings. Oskouie and Farnia [21] has shown the existence of a Universal Perturbation for Interpretation (UPI) for standard image datasets, which can alter the gradient-based feature map of neural networks in a wide range of test samples. This UPI is computed using a gradient-based optimization method and principal component analysis, effectively altering the underlying model's gradient-based interpretation on different samples. This work focuses on attacking IDLSes by generating universal perturbations with a broader impact across diverse samples in a target category regardless of the interpreters employed. The properties of Single**ADV** and previous studies are summarized in Table I.

III. Methods

This section discusses several key concepts related to our work. We first introduce the problem formulation and then describe the main algorithm for Single**ADV**.

A. Fundamental Concepts

1) DL Model: As our paper mainly focuses on the classification task, let $f(x) = y \in Y$ denote a classifier that assigns an input x to a category y from a set of categories Y.

2) Interpreter: Let $g(x; f) = m$ denote an interpreter g that generates an attribution (*i.e., interpretation*) map m that reflects the importance of features in a sample x based on the output of the classifier f, (*i.e.,* the value of the i-th element in m reflects the importance of the i-th element in x). Based on the methods used to obtain interpretations of a model, interpretation models can be divided into two types:

❶ **Pre-hoc Interpretability:** It is achieved by constructing self-explanatory models that integrate interpretability directly into their structures. In other words, this type of interpretability focuses on building DL models that can explain their behavior explicitly in terms of the inference process [7], [24]. The category includes decision tree, rule-based model, attention model, *etc.*

❷ **Post-hoc Interpretability:** Post-hoc Interpretability is based on the complexity-regulated DL model interpretation or adopting post-training methods [25], [26]. This type of interpretation requires another model to provide explanations for the current model.

Our proposed attack specifically targets post-hoc interpretability, which involves the use of an interpreter that receives information from the target DL model (*e.g.,* gradients) and generates an interpretation of how the target model classifies an input sample. One reason for choosing this type of interpreter is that it does not require any modifications to

TABLE I
COMPARISON OF RELATED WORKS BASED ON SEVERAL ASPECTS

Research Studies	Adversarial Attack	Perturbation Generation	Universal Perturbation	Targeting Single Category	Interpretation-based Attack	Adaptive to Interpreters	White-box Attack	Black-box Attack
Moosavi et al. [12]	✓	✓	✓	✗	✗	✗	✓	✗
Khrulkov et al. [13]	✓	✓	✓	✗	✗	✗	✓	✗
Hayes et al. [14]	✓	✓	✓	✗	✗	✗	✓	✗
Heo et al. [15]	✓	✗	✗	✗	✓	✗	✓	✗
Zhang et al. [9]	✓	✓	✗	✗	✓	✓	✓	✗
Abdukhamidov et al. [8]	✓	✓	✗	✗	✓	✓	✓	✗
Akhtar et al. [10]	✓	✓	✓	✓	✗	✗	✓	✗
Weng et al. [16]	✓	✓	✓	✗	✗	✗	✓	✓
Liu et al. [17]	✓	✓	✓	✗	✗	✗	✓	✓
Ye et al. [18]	✓	✓	✓	✗	✗	✗	✓	✓
Guanhong et al. [19]	✓	✓	✓	✗	✗	✗	✗	✓
Qin et al. [20]	✓	✓	✓	✗	✗	✗	✓	✗
Oskouie et al. [21]	✓	✓	✓	✗	✓	✗	✓	✗
Ours (SingleADV)	✓	✓	✓	✓	✓	✓	✓	✓

the architecture of the prediction model, thereby preserving its high prediction accuracy.

3) Threat Model: This work considers both white-box and black-box attack scenarios for target-specific (*i.e.,* targeted) attacks against a given deep learning model and its coupled interpreter. In the white-box setting, the adversary has complete access to the victim classifiers f and their interpreters g, and aims to target specific category with minimal effect on other categories. The white-box threat model has been adopted by previous studies (*e.g.,* targeted [10] and non-targeted [8] attacks. Although white-box scenarios provide valuable insight into the strengths and weaknesses of the system, they are often impractical in real-world applications. Therefore, we also consider a black-box environment in which the adversary has limited knowledge about the victim classifier, and can only query the black-box model to receive its output.

B. Target Interpretation Model

The interpreters chosen for our experiment are representative of state-of-the-art interpretation techniques. For example, Grad [27] shares the same formulations as DeepLift [28], SmoothGrad [29], *etc.*, while CAM [6] belongs to the same family of representation-guided interpreters (*e.g.,* Grad-CAM [30]). We also use the MASK interpreter [31] from the perturbation-guided interpreters. Thus, the attack is applicable to other related interpreters.

1) CAM: In **C**lass **A**ctivation **M**ap (CAM) [6], interpretation maps are generated using feature maps taken from the target DL classifier's intermediate layers. The significance of the areas of the samples is generated by reflecting the weights of the fully-connected layer on the features maps of the convolutional layer. Assume $a_i(j,k)$ as the activation of a channel i in the last convolutional layer at a spatial position (j,k), and $\sum_{j,k} a_i(j,k)$ denote the outcome of global average pooling. So, the softmax function receives: $\psi_y(x) = \sum_{j,k}\sum_i w_{i,y}\ a_i(j,k)$, and the attribution map m_y is as follows: $m_y(j,k) = \sum_i w_{i,y}\ a_i(j,k)$, where $w_{i,y}$ is the weight associated with the output class y for i-th channel. We construct interpretation maps by collecting and combining feature vectors from f up to its last CNN layer.

2) Grad: To generate the importance of features of a given sample, the gradient of a DL classifier's outcome in terms of the given sample is computed by the interpreter. To be specific, based on the DL model f and its prediction y to a certain sample x, the interpreter generates an interpretation map or also called attribution map m as $m_y = \left|\frac{\partial f_y(x)}{\partial x}\right|$. Since the ReLU activation function is used in the target DL models, and all CNN-based models, the computed result of the Hessian matrix becomes all-zero. To find an optimal adversarial sample $\hat{x}$, the gradient of the ReLU $r(z)$ function is approximated as:

$$h(z) \triangleq \begin{cases} (z+\sqrt{z^2+\tau})' = 1 + \dfrac{z}{\sqrt{z^2+\tau}} & \text{for} \quad z < 0 \\ (\sqrt{z^2+\tau})' = \dfrac{z}{\sqrt{z^2+\tau}} & \text{for} \quad z \geq 0 \end{cases}$$

where $h(z)$ approximates the gradient of ReLU $r(z)$ with a small constant parameter τ (*e.g.,* $\tau = 1e-4$) [9].

3) MASK: This interpreter [31] generates interpretation maps by detecting changes in the prediction of a DL model while adding a minimal amount of noise to the sample. Specifically, the interpreter creates a mask m that is a binary matrix of the same size as the sample x. In the matrix, 0 represents the area of the sample x where the feature is kept without noise. The value 1 in the matrix means that the area is replaced with Gaussian noise. The main objective of the interpreter is to find the smallest *mask* that disturbs a model's performance:

$$\min_{mask} : f_y(\phi(x;_{mask})) + \lambda\ \|1 - mask\|_1 \quad s.t. \quad 0 \leq_{mask} \leq 1, \tag{1}$$

where $\phi(x; mask)$ is the operator that generates perturbation to decrease the probability of the current prediction category y and the second term $\lambda\ \|1 - mask\|$ helps the mask to be scattered. Since the MASK interpreter is an optimization function, applying the attack (as another optimization) is directly infeasible. We reformulate the attack as a bi-level optimization task (similar to the framework in [8] and [9]).

C. Problem Formulation

Let S be a distribution over the dataset and $s \in \mathbb{R}^d$ indicate a sample from a distribution S. For trained model $f(s) \rightarrow y$, where y is the correct class, the main objective of adversarial attacks is to generate a perturbation $p \in \mathbb{R}^d$ that satisfies the following constraint:

$$f(s+p) \rightarrow y_t, where\ y \neq y_t,\ \|p\|_{\ell_p} \leq \eta \tag{2}$$

In Equation (2), confining y_t to a chosen category and ℓ_p-norm to a predefined value of η produces a targeted attack.

Generating universal perturbation in adversarial attacks expands the domain of p, which we denote as $D(p)$. Given that $|D(p)| \geq 1$, where $|\ .\ |$ denotes the cardinality of a set, we maximize the objective of Equation (2) as follows:

$$\max_{p} \mathbf{P}_{D(p)}(f(s+p) \to y_t), \text{ where:}$$
$$\mathbf{P}_{D(p)}(f(s+p) \to y_t) \geq \gamma, \|p\|_{\ell_p} \leq \eta, \text{and } |D(p)| \geq 1 \quad (3)$$

In Equation (3), $\mathbf{P}$ refers to the probability that, for all samples within the domain defined by $D(p)$ applying the perturbation p to an input sample s results in the model f predicting the target class y_t. In other words, it aims to find a perturbation p that, with high probability (controlled by the adversarial confidence threshold γ in the range of [0, 1]), can fool the model f into predicting the target class for all relevant input samples. The constraint $|D(p)| \geq 1$ ensures that the perturbation p is effective in a wide range of samples in the domain set, not just for one sample, which is crucial to ensure the universality of the perturbation across the dataset. We use $\eta = 7$ and $\gamma = 0.7$ in our experiments. Experimentally, these settings provide the best trade-off between effectiveness and computational cost.

1) Rationale for Thresholds: In the context of adversarial attacks, ensuring the success of the misclassification and the model's high prediction confidence are crucial. This is achieved by introducing a threshold for other classes in our minimization problem to ensures that the model's prediction confidence of the adversarial sample remains high enough to avoid two potential issues: the rejection of the predicted outcome [32], [33] and out-of-world scenarios [34], [35]. Many classifiers reject predictions if the confidence score for the predicted class is below a specific threshold to avoid incorrect predictions and ensure the model only produces relatively confident outputs. The attack might fail in this case because the model could reject these predictions rather than misclassify them into the targeted class. In the case of the Out-of-World Scenario, the model's prediction also does not align with any known class, often due to extremely low confidence across all classes, and outputs a special "unknown" or "out-of-distribution" class, effectively mitigating the adversarial attack. Therefore, the threshold ensures effective adversarial perturbations by maintaining high misclassification confidence, preventing the rejection of such samples.

In Single**ADV** case, we consider an interpretation model to generate universal perturbation that keeps the adversarial interpretation result similar to the interpretation of the original samples. Hence, we have the other following constraints:

❶ Ensuring model misclassification to a predefined category: $f(s+p) \to y_t$, $y \neq y_t$, where y_t is the target category.
❷ Restricting the sample domain to the selected category: $D(p) = \{s \mid s \sim S_{selected}\}$, where $S_{selected}$ is the sample distribution from the category considered for the attack.
❸ Restricting the effect of the perturbation on other categories' domain: $\mathbf{P}_{\hat{D}(p)}(f(\hat{s}+p) \to y_t) < \gamma$, where $\hat{D}(p)$ represents the domain and $\hat{s}$ represents the samples from non-selected categories.
❹ Triggering an interpreter g to generate target attribution maps: $g(s+p;\ f) \xrightarrow{similar} m_t$, such that $g(s;\ f) \to m_t$.

We focus on the single-class attack in our research for the following reasons. ❶ Targeting a specific class allows fine-tuning adversarial perturbations to exploit unique vulnerabilities, leading to a higher success rate and more controlled optimization. ❷ We aim to create UAPs for a single class, which can enhance the manipulation of deep features associated with that class. This helps obtain specific yet effective perturbations that achieve the adversarial objectives with minimal effect on other categories while maintaining similar interpretations. ❸ Adversarial samples can appear less suspicious when targeting a single class, creating an additional layer of stealth and posing challenges in detecting such samples.

D. Computing the Perturbation

The algorithm described in Algorithm 1 generates perturbations that satisfy the constraints mentioned in Subsection III-C. The objective of the algorithm is to calculate a universal perturbation through a series of steps that aim to minimize the cost of the attack. The goal is to increase the confidence of adversarial samples for the selected category in both the target classifier and the interpreter while minimizing the difference between benign and adversarial attribution maps.

The desired cost surface for high confidence and low interpretation loss is based on stochastic computation and is ruled by first- and second-moment estimations. The first and second moments check if the computed perturbation prevents other categories from crossing their decision boundaries during the generation process. The computed perturbation should not interfere with the prediction of non-source classes. ℓ_∞ norm is applied to bounce the perturbation norm. We explain each line of the algorithm in detail.

The *typical* attack is based on the white-box scenario, as it requires the target model's parameters. Samples of the selected category and other categories, S and $\hat{S}$, are accumulated from $D(p)$ and $\hat{D}(p)$, respectively. Given S and $\hat{S}$, model f, interpreter g, η for the ℓ_p-norm of the perturbation, target category y_t, batch size b, adversarial confidence γ (confidence level as a target category), pre-set first- and second-moment hyperparameters, β_1 and β_2, the algorithm starts the initialization process by setting $p_0, \upsilon_0, \omega_0 \in \mathbb{R}^d$ as zero vectors.

The algorithm first randomly samples the selected category and the other categories in sets S_x and S_o, and the cardinality of each set is half of the batch size b (*line 3*). Then the attribution maps M and $\hat{M}$ are calculated as $g(S_x;\ f)$ and $g(S_o;\ f)$, respectively (*line 4*).

In *line 5*, all sets are perturbed by subtracting the currently estimated perturbation p_i from each of them (the operation is displayed as $\ominus$). The subtraction optimizes the perturbation effectively by aligning with the negative gradient direction, ensuring efficient exploration of the adversarial space while maintaining norm constraints, which makes it suitable in gradient-based optimization and preventing data saturation issues. The perturbed samples are then clipped to a valid range (*e.g.,* [0,255] in 8-bit pixel values) using the C function.

In *line 7*, we calculate the ratio between the expected norms (it is referred to as $\mathbb{E}$ in the algorithm) of the gradients of

Algorithm 1 Single**ADV** Attack's Main Algorithm

```
Data: Target model $f$, interpreter $g$, selected category
      samples $S$, non-selected categories samples $\hat{S}$
      s.t. $S_i$ or $\hat{S}_i \in \mathbb{R}^d$, target category $y_t$,
      perturbation norm $\eta$, balance factor $\lambda$, batch
      size $b$, and adversarial confidence $\gamma$,
      $\beta_1 = 0.9$ and $\beta_2 = 0.999$.
Result: Universal Adversarial Perturbation $p \in \mathbb{R}^d$
1  Initialization: $i = 0$, and setting $p_0, \upsilon_0, \omega_0 \in \mathbb{R}^d$ ;
2  while adversarial confidence $< \gamma$ do
3      $S_x \overset{\text{rand}}{\sim} S,\ S_o \overset{\text{rand}}{\sim} \hat{S}$ s.t. $|S_x| = |S_o| = \frac{b}{2}$;
4      $M \leftarrow g(S_x; f)$ and $\hat{M} \leftarrow g(S_o; f)$ ;
5      $X \leftarrow \mathrm{C}(S_x \ominus p_i),\ \hat{X} \leftarrow \mathrm{C}(S_o \ominus p_i)$;
6      $i \leftarrow i + 1$;
7      $\vartheta \leftarrow \frac{\mathbb{E}_{x_i \in X, m_i \in M}\left(\|\nabla_{x_i}(l_{prd}(f(x_i, y_t)) + \lambda\ l_{int}(g(x_i; f), m_i))\|_2\right)}{\mathbb{E}_{\hat{x}_i \in \hat{X}, \hat{m}_i \in \hat{M}}\left(\|\nabla_{\hat{x}_i}(l_{prd}(f(\hat{x}_i, y)) + \lambda\ l_{int}(g(\hat{x}_i; f), \hat{m}_i))\|_2\right)}$;
8      $\psi_i \leftarrow \frac{1}{2}\Big(\mathbb{E}_{x_i \in X, m_i \in M}\big[\nabla_{x_i}(l_{prd}(f(x_i, y_t)) +$
                $\lambda\ l_{int}(g(x_i; f), m_i))\big] +$
           $\vartheta\ \mathbb{E}_{\hat{x}_i \in \hat{X}, \hat{m}_i \in \hat{M}}\big[\nabla_{\hat{x}_i}(l_{prd}(f(\hat{x}_i, y)) +$
                $\lambda\ l_{int}(g(\hat{x}_i; f), \hat{m}_i))\big]\Big)$;
9      $\upsilon_i \leftarrow \beta_1 \upsilon_{i-1} + (1 - \beta_1)\ \psi_i$;
10     $\omega_i \leftarrow \beta_2 \omega_{i-1} + (1 - \beta_2)(\psi_i \odot \psi_i)$;
11     $\bar{p} \leftarrow \frac{\sqrt{1-\beta_2^i}}{1-\beta_1^i}.\ diag\big(diag(\sqrt{\omega_i})^{-1} \upsilon_i\big)$;
12     $p_i \leftarrow p_{i-1} + \frac{\bar{p}}{\|\bar{p}\|_\infty}$;
13     $p_i \leftarrow sign(p_i) \odot \min(|p_i|, \eta)$;
14 end
```

selected category and other categories using the selected and non-selected category samples, and the computed attribution maps of the interpreter. In the algorithm, l_{prd} is the classification loss that shows the difference between the prediction of the model and the intended category, while l_{int} is the interpretation loss to calculate the difference between the current and target attribution maps: $l_{int}(g(x; f), m)) = \|(g(x; f) - m\|_2^2$. Notice that we compute the gradients of the selected category and other categories differently w.r.t. prediction labels (*i.e.,* selected samples $x \rightarrow y_t$ and other samples $\hat{x} \rightarrow y$ where y is the right label). $\nabla_{x_i}(l_{prd}(f(x_i, y_t))$ (ignoring the negative sign) provides the direction to deceive the model into predicting y_t for x_i when the sample is from the source category, while $\nabla_{\hat{x}_i}(l_{prd}(f(\hat{x}_i, y))$ increases the confidence of a correct prediction when the sample is from other categories. Since we consider the interpretation loss in the context of the chosen direction for optimization, we use λ to scale the interpretation loss l_{int} and balance the two factors (*i.e.,* l_{prd} and l_{int}) within the overall optimization. The value of the hyperparameter depends on the interpreter. In our experiments, we explored different values of λ to account for different interpreters (*e.g.,* 0.02, 0.65, 400 for Grad, CAM, and MASK, respectively). The norms of calculated gradients might vary significantly, and we use the calculated scaling factor ϑ to account for such variation. Using ϑ, we calculate ψ_i as the weighted average of the expected gradients for both the source (selected) and non-source (non-selected) categories (*line 8*). Next, the first and raw second moments, *i.e.,* υ_i and ω_i, of the computed gradients are calculated using exponential moving averages with decay factors of β_1 and β_2 (*line 9* and *line 10*). The $\odot$ represents the Hadamard product for effective stochastic optimization on the cost surface. On *line 11*, we perform a bias-corrected estimation, since the second moment (*i.e.,* moving average) is known to be heavily biased in the early stages of optimization. The derivation of the expression in *line 11* is explained in [10]. The perturbation is computed by the ratio between the moment estimates ($\frac{\upsilon_i}{\sqrt{\omega_i}}$, where ω_i represents the second moment), and as the notations are vectors, we convert vectors into diagonal matrices or diagonal matrices into vectors via $diag(.)$ operation. Finally, we update the perturbation by restricting the $\bar{p}$ with ℓ_∞-norm to keep the desired direction (*line 12*). The norm of the computed perturbation is restricted by ℓ_∞-ball projection at the end of each iteration to minimize the perturbation perceptibility by performing the Hadamard ($\odot$) product between the element-wise sign ($sign(.)$) and the minimum values for perturbation (*line 13*).

The objective of Single**ADV** is not only to deceive the classifier by making it misclassify the designated source class samples while correctly predicting non-source classes but also to preserve the original interpretation for all samples of the source class. From an adversarial standpoint, this approach minimizes the suspicion of the attack by manipulating a single class rather than making the classifier predict the same label for all images and providing different attribution maps.

IV. Single**ADV** in White-Box Settings

A. Experimental Settings

1) Dataset: We use the ImageNet dataset [1] for our experiments. The training set consists of 1,300 samples per category from the ImageNet dataset. We use the training set within the attack framework to calculate the perturbation. The testing portion of the dataset, which includes 50 samples per category (both source and non-source categories), is used to evaluate the generated perturbation. To ensure accurate gradient directions, we randomly select samples that are correctly classified with ≥60% confidence from both the targeted and non-targeted categories. In this paper, we designate certain object classes as the targeted category while considering other classes as the non-targeted category. For instance, categories such as **panda**, **dog**, and **cup** are used as targeted categories (see Table II).

2) DL Models: In the experiments, four pretrained versions of state-of-the-art DL models are used for our targeted attack, which are **ResNet-50** [3] (74.90% top-1 accuracy and 92.10% top-5 accuracy on ImageNet dataset), **VGG-16** [2] (71.30% top-1 accuracy and 90.10% top-5 accuracy on ImageNet dataset), **DenseNet-169** [36] (76.20% top-1 accuracy and 93.15% top-5 accuracy on ImageNet dataset), and **Inception-V3** [37] (77.90 % top-1 accuracy and 93.70% top-5 accuracy on ImageNet dataset). The models are chosen in terms of performance and network architecture to help measure the effectiveness of our attack.

3) Interpretation Models: **CAM** [6], **Grad** [27] and **MASK** [31] interpreters are utilized as representative of interpretation models. Their original open-source implementations are used for our experiment to build interpreters of all DL models.

4) Attack Evaluation: The effectiveness of the attack is evaluated by conducting several experiments and calculating some evaluation metrics in terms of attack success rate in fooling the classifier while maintaining a convincing interpretation. The evaluation aims to find answers to the following questions: ❶ *Is our technique effective in attacking DL models by targeting a single category?* ❷ *Is our technique effectively misleading the interpreters by generating a similar interpretation to the benign sample?*

5) Evaluation Metrics: To assess the effectiveness of the proposed attack against the target IDLSes, we utilize various metrics. These metrics are employed to evaluate the attack's impact on both the classifiers and interpreters. The following metrics are used in our evaluation:

Against Classifiers:

- **Fooling Ratio:** This metric [12] measures the proportion of images that undergo the universal perturbation and have their labels changed. It indicates the attack's success in causing misclassifications by the target model, specifically into the target category. This metric provides a quantitative measure of the attack success against DL models.
- **Misclassification Confidence:** This metric [9] measures the probability (*i.e., confidence*) of an adversarial sample assigned to the target category (*i.e.,* we measure the average confidence scores of adversarial samples successfully misclassified).
- **Classification Confidence:** This metric [9] evaluates the impact of universal perturbations on non-target categories. It measures the confidence scores of these categories when the perturbation is applied, revealing the extent to which the model's performance is affected while maintaining its effectiveness against the targeted category.
- **Leakage Rate:** This metric [12] measures the impact of universal perturbations on non-source categories. It is calculated as the ratio of misclassified non-source category images to the total number of non-source category images used for testing. A lower leakage rate indicates that universal perturbations are more specific to the target category, while a higher leakage rate suggests a more general impact.
- **Noise Rate:** We use the inverse of the average Structural Similarity Index Measure (SSIM) [38] between the original and perturbed images to measure the visual disturbance. A higher score signifies greater visual distortion, while a lower score indicates a more imperceptible perturbation, which is preferable to maintain the visual quality after the attack.

Against Interpreters:

- **Qualitative Comparison:** This method [9] is used to verify whether the results of interpretation are perceptually similar. Manual inspection is conducted to compare interpretation maps to their corresponding benign versions.
- **IoU Test** (**I**ntersection-**o**ver-**U**nion): This metric [39] calculates the intersection areas between the adversarial and the benign interpretation maps. To conduct the IoU test, we implement a binarization process. Specifically, we apply ten thresholds within the range [0, 1], converting the continuous values of benign and adversarial interpretation maps into binary representations. The similarity of these binary maps is then assessed at each threshold, and an average similarity score is calculated across all thresholds. We calculate scores as: $\text{IoU}(m, m_\circ) = |O(m) \bigcap O(m_\circ)| \div |O(m) \bigcup O(m_\circ)|$, where m denotes the binary attribution map of samples with the universal perturbation, and $m_\circ$ represents the binary attribution map of samples without perturbation. Here, $O(.)$ is the binarization function applied at each threshold.

B. Attack Effectiveness Against DL Models

We first assess the attack's effectiveness in deceiving the classifiers. The results are summarized using the *fooling ratio*, *misclassification confidence*, and *leakage rate* with the perturbation norm $\eta = 7$ and adversarial confidence $\gamma = 0.7$ in Table II. The reported results (*i.e.,* fooling ratio or attack success rate, misclassification confidence, and leakage rate) are based on the test samples that are not seen by the selected model and the attack algorithm. The source and target categories were selected randomly. We also included the results of the *Targeted Fooling Attack* [10] for comparison. We used similar settings for a fair comparison. Since the attack [10] does not require any interpreter to generate adversarial samples, we only need to test it against DNN models once. The success of the attack is demonstrated by fooling ResNet-50, VGG-16, DenseNet-169, and Inception-V3 trained on the ImageNet dataset [1].

Observing the results for four classifiers (*i.e.,* ResNet-50, VGG-16, DenseNet-169, and Inception-V3), the attack generates a universal perturbation for different architectures with high fooling rates. More specifically, VGG-16 and ResNet-50 were deceived with the universal perturbation of Single**ADV** with a success rate of more than 70% for all target categories regardless of the interpreter. This means that the addition of the universal perturbation to any raw image in our test samples can deceive the target DL models more than seven times out of ten. The attack also achieved better results with a higher than 60% fooling ratio in all interpreters when Densenet-169 and Inception-V3 were used as the target DL models. Among the target models, Inception-V3 was attacked with a relatively lower fooling ratio while VGG-16 achieved a higher fooling ratio compared to the other models. Based on the misclassification confidence results, the attack fools target DL models with confidence scores higher than 70% regardless of the interpreters employed. Higher scores can be seen when the attack is implemented against ResNet-50 and DenseNet-169, while the Inception-V3 model provides lower scores in all interpreters. The main reason for the lower confidence scores of Inception-V3 could be due to its architectural design and global averaging before the output layer, which reduces the computational cost and decreases the effect of perturbation on the output. Compared to existing attack [10], our attack algorithm demonstrates better misclassification confidence performance on the VGG-16, ResNet-50, and DenseNet-169 models for Targets 1 and 2. Taking into account the multiple objectives of simultaneously targeting

TABLE II

FOOLING RATIO, MISCLASSIFICATION CONFIDENCE, AND LEAKAGE RATE AGAINST SEVERAL MODELS USING IMAGENET.THE RESULTS SHOW SOURCE CATEGORY $\xrightarrow{\text{TO}}$ TARGET CATEGORY FOR TARGET 1: PANDA → **CAT**, TARGET 2: DOG → **GOOSE**, TARGET 3: CUP → **WOLF**.THE BEST RESULTS WITH RESPECT TO THE LAST ROW (TARGETED FOOLING ATTACK [10] WITH NO INTERPRETATION CONSIDERED) ARE PROVIDED IN BOLD

Interpreter	Model	Target 1 Fooling Ratio	Target 1 Misclassification Confidence	Target 1 Leakage Rate	Target 2 Fooling Ratio	Target 2 Misclassification Confidence	Target 2 Leakage Rate	Target 3 Fooling Ratio	Target 3 Misclassification Confidence	Target 3 Leakage Rate	Average Fooling Ratio	Average Misclassification Confidence	Average Leakage Rate
CAM	VGG-16	**0.85**	0.82	**0.30**	**0.87**	0.85	0.35	0.71	0.80	**0.32**	0.81 ± 0.07	**0.82**	**0.32**
	ResNet-50	**0.86**	**0.86**	**0.36**	0.79	0.82	0.34	**0.75**	**0.79**	**0.35**	**0.80 ± 0.05**	**0.82**	**0.35**
	DenseNet-169	**0.78**	0.81	0.34	0.75	0.80	0.33	0.69	0.77	0.36	0.74 ± 0.05	**0.79**	**0.34**
	Inception-V3	0.68	**0.79**	**0.30**	0.62	**0.78**	**0.31**	0.60	0.75	**0.29**	0.63 ± 0.04	**0.77**	**0.30**
Grad	VGG-16	**0.87**	0.71	0.34	**0.85**	0.75	0.36	0.75	0.77	**0.31**	**0.82 ± 0.03**	0.74	**0.34**
	ResNet-50	0.83	**0.88**	0.38	**0.81**	**0.85**	0.37	0.72	0.76	**0.35**	**0.79 ± 0.02**	**0.83**	0.37
	DenseNet-169	**0.80**	**0.86**	**0.35**	0.73	**0.82**	**0.33**	0.70	0.76	**0.35**	0.74 ± 0.02	**0.81**	**0.34**
	Inception-V3	0.72	**0.80**	**0.31**	0.68	**0.79**	**0.33**	0.65	0.72	**0.30**	0.68 ± 0.01	**0.77**	**0.31**
MASK	VGG-16	0.81	0.79	0.36	0.76	0.77	**0.34**	0.72	0.70	**0.33**	0.76 ± 0.04	0.75	**0.34**
	ResNet-50	0.80	**0.82**	0.37	0.73	**0.84**	0.36	0.70	0.72	**0.34**	0.74 ± 0.04	0.79	0.36
	DenseNet-169	0.77	**0.81**	**0.35**	0.71	**0.83**	**0.33**	0.68	0.75	**0.34**	0.72 ± 0.04	**0.80**	**0.34**
	Inception-V3	0.69	**0.76**	**0.30**	0.64	**0.79**	**0.34**	0.63	0.70	**0.31**	0.65 ± 0.03	**0.75**	**0.32**
N/A [10]	VGG-16	0.85	0.84	0.31	0.82	0.87	0.34	0.76	0.81	0.36	0.81 ± 0.05	0.84	0.34
	ResNet-50	0.85	0.78	0.36	0.80	0.83	0.32	0.73	0.79	0.37	0.79 ± 0.06	0.80	0.35
	DenseNet-169	0.78	0.77	0.37	0.78	0.77	0.36	0.73	0.78	0.38	0.76 ± 0.03	0.77	0.37
	Inception-V3	0.75	0.67	0.39	0.73	0.70	0.37	0.68	0.78	0.40	0.72 ± 0.04	0.72	0.39

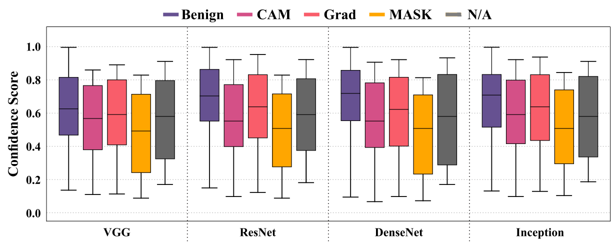

Fig. 3. Classification confidence of adversarial samples (non-source categories) based on various models and interpreters. The N/A (no interpreter) shows the results of the existing attack [10]. The universal perturbation is for Dog → **Goose**.

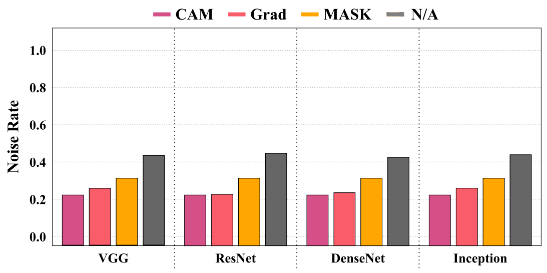

Fig. 4. Noise rate of adversarial samples of Targets 1, 2, and 3 based on various models and interpreters. The N/A (no interpreter) shows the results of the existing attack [10].

DL models and interpreters, our approach exploits specific vulnerabilities within these architectures.

According to the leakage rate analysis, Single**ADV** appears to perform well. The computed leakage rate indicates that the algorithm's universal perturbation has a limited impact on non-source classes, with an average of 33% leakage rate across all interpreters while the existing attack [10] has an average leakage rate of 36%. This suggests that the perturbation is more specific to the targeted source class and has minimal impact on other non-source classes, which is a desirable characteristic. A low leakage rate implies that the algorithm's universal perturbation has a more targeted effect and is less likely to cause errors in the classification of non-source classes.

The results of the classification confidence metric are presented in Figure 3, which depicts the impact of a universal perturbation on the confidence scores assigned by the DL models for non-source categories when various interpreters are used. To provide a basis for comparison, the scores obtained without any universal perturbation and the scores of the existing attack [10] are also included. The figure shows that the universal perturbations generated using the CAM and Grad interpreters have a smaller impact on non-source categories than that of the MASK interpreter. Nevertheless, the results indicate that the perturbation has a minimal impact on the scores of non-target categories.

To determine the noise rate of UAPs, the perturbation is applied to each image in the dataset, and the inverse of SSIM (1 - SSIM) is evaluated between the perturbed and original versions. Figure 4 displays the results of the noise rate of Single**ADV** and [10]. The results indicate that our attack yields low levels of noise, demonstrating its ability to create effective and impactful adversarial examples.

C. Attack Effectiveness Against the Interpreters

First, we use qualitative comparison to verify whether the attribution maps produced from adversarial samples are perceptually similar to their benign samples. We checked all adversarial attribution maps and found that observing all the cases for all targeted categories, Single**ADV** attack generates universal perturbations that produce attribution maps on adversarial domains similar to or indistinguishable from attribution maps in the corresponding benign domain. Figure 5 displays a set of samples along with their attribution maps based on the **CAM**, **Grad**, and **MASK** interpreters. The samples are randomly selected from the set of outputs of our attack. In the figure, the first three columns display benign samples, their attribution maps, and their prediction categories. The last three columns present universal perturbations generated based on a DL model and an adopted interpreter, adversarial attribution maps produced by adding the universal perturbations to benign samples, and target prediction categories. As shown in the figure, the results support high similarity in terms of interpretations. By observing the attribution maps produced for adversarial samples in both targeted and non-targeted categories, Single**ADV** produces perturbations that only affect

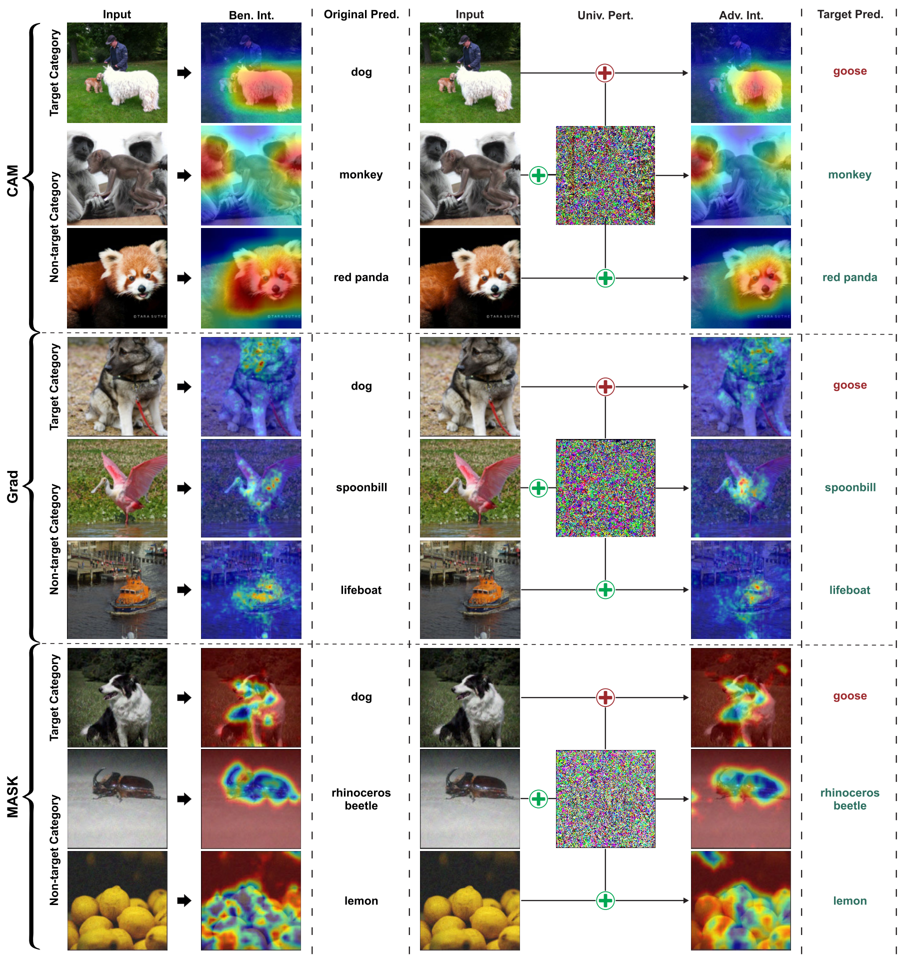

Fig. 5. Attribution maps of benign and adversarial samples based on VGG-16, ResNet-50 with CAM, Grad and MASK interpreters. In this sample, our target category is Dog → **Goose**. (Ben.Int. stands for Benign Interpretation, Adv.Int. stands for Adversarial Interpretation and Univ.Pert. stands for Universal Perturbation).

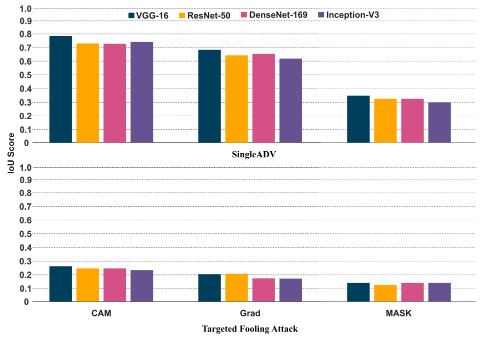

Fig. 6. IoU scores of Single**ADV** (Upper) and [10] (Lower) for attribution maps of adversarial samples. The results are based on CAM, Grad, and MASK interpreters for VGG-16, ResNet-50, DenseNet-169, and Inception-V3.

the target category in terms of prediction while maintaining accurate interpretations across all categories.

Additionally, the attribution maps of benign and adversarial samples are compared using the IoU score metric. The IoU score is calculated by binarizing the attribution maps using a threshold value. Any value above the threshold is assigned a value of 1, while any value below the threshold is set to 0. We apply different threshold values (*i.e.,* 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9) to measure IoU scores and report the average of those IoU scores in Figure 6. The region of interest is generally considered positive if it has an IoU score of 0.5 or higher compared to the ground truth. Therefore, we assume that adversarial interpretation maps with an IoU ≥ 0.5 based on their benign interpretation maps are credible. The IoU scores for all evaluated models, namely VGG-16, ResNet-50, DenseNet-169, and Inception-V3, exceed 60% for Single**ADV** when using CAM and Grad interpreters, as shown in Figure 6. However, our attack shows lower IoU scores for all models when using the MASK interpreter. This reduction in IoU scores with the MASK interpreter can be attributed to differences in the shape and size of the regions identified as important for benign and adversarial attribution maps. Further discussion regarding the factors contributing to lower IoU scores with the MASK interpreter is in Section VII. Compared to the IoU scores of [10], which are below 30%, our attack maintains a higher degree of stealth, preserving the similarity of the attribution maps to those of benign samples, which is crucial to creating convincing adversarial examples.

V. SINGLE**ADV** IN BLACK-BOX SETTINGS

Since white-box attacks have limited practicality in real-life cases, we also employ our proposed method in black-box

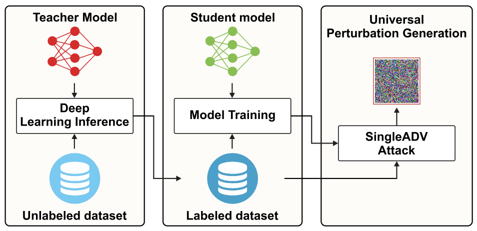

Fig. 7. Teacher-student strategy of Single**ADV** in a black-box environment, where the unlabeled dataset is pseudo-labeled using the teacher model (a black-box model). Single**ADV** uses the student model to learn universal perturbation.

settings. We focus primarily on specific architectures and interpretation models to establish a foundational understanding and serve as a proof of concept for our method. However, we note that our focus on a select group of models may not fully demonstrate the effectiveness of Single**ADV** with a wider range of models and interpretation tools.

A. Methodology

To apply Single**ADV** in black-box settings, we adopt a teacher-student learning strategy, a method considered effective in attacking black-box DL models [40], [41], [42]. In a black-box scenario, the adversary has limited knowledge about the target DL model (*teacher model*), the training process, and the data used for training, and can only interact with it by querying the model and receiving the output. To overcome this limitation, the adversary uses a *student model*, which is trained to approximate the decision boundaries of the teacher model. This process begins by preparing a substitute dataset, relevant to the task of the teacher model (but not requiring its labels), and querying the teacher model to obtain pseudo-labels for samples in the substitute dataset. These pseudo-labeled data are then used to train the student model, ensuring that the architecture of the student is sufficiently complex to mimic the teacher's behavior [42]. After training, the student model serves as a base for attacking the teacher model, *i.e.,* helps to generate adversarial examples that can exploit vulnerabilities in the teacher model when targeting specific categories.

Figure 7 illustrates the teacher-student strategy. The perturbation that triggers misclassification in the student model for a selected category is then tested against the teacher model to validate whether it can cause a similar misclassification. In our experiments, the total number of queries to the target model is determined by the size of the unlabeled dataset used in training the 'student model.' Each dataset sample equates to one query, and we perform one more query to test the effectiveness of the universal perturbation generated by the student model. Hence, the total query count equals the size of the unlabeled dataset plus one, providing a straightforward measure of our method's computational efficiency in real-world scenarios.

1) Dataset: In the experiment, we randomly select a deep learning app from Google Play to use its deep learning model as a target black-box model. We use an app called Bei Ke that is used to identify scenes. By tracking the APIs of the deep learning frameworks in the app using tools called Soot [43] and FlowDroid [44], we observe the working and invoking processes of the DL framework. The tools help us invoke the DL model of the app to send a sample and receive its response. We utilize the ImageNet dataset as an unlabeled dataset, which we find most relevant to the app task. We query the teacher model (*i.e.,* the DL model within the app) to label the ImageNet dataset, which we find to be the most relevant to the app task. This process involves only the training dataset we used in previous experiments, *i.e.,* 1,300 training images per class. This newly labeled dataset is used to train the student model. After training is complete, 30 random samples from the test set are selected to test our approach against one universal perturbation of the target category.

2) Student Model: In the teacher-student learning approach, a student model with a complex architecture is the correct option to imitate a teacher model well. The more complex the student model, the higher the success rate of the attack [42]. Therefore, we adopt the VGG model architectures (VGG-11 and VGG-16) as student models. In training student models, we use the SGD optimizer [45] with a learning rate of 0.001 to optimize cross-entropy loss for 70 epochs.

3) Interpreter: We employ the CAM interpreter based on its performance shown in Figure 6. However, the target black-box model (teacher model) does not provide interpretability, and we cannot add the CAM as we do not have access. Therefore, we employ only the fooling ratio metric for the teacher model while calculating the fooling ratio and the similarity of attribution maps for the student model. To show the attack's effectiveness in generating similar interpretations with the adversarial samples as the ones for the benign samples, we perform a controlled experiment where we do not have access to the teacher model, but generate interpretation maps from the student model. We adopt ResNet-50 as the teacher model and VGG-16 as the student model.

For the reproducibility of our experiments, our code, data, and models are available at (*https://github.com/InfoLab-SKKU/SingleClassADV*).

B. Experimental Results

Figure 8 depicts the results of the experiment. Since the app does not provide interpretability, we test the app in terms of the fooling ratio, while the IoU test results are calculated from the attribution maps of the student models. The attack achieves fooling ratios of more than 60% and 70% with interpretation maps of more than 60% and 70% similarity against benign ones, using Student 1 (VGG-11) and Student 2 (VGG-16), respectively. The same perturbation reaches about 30% and 50% in the fooling ratio when used against the teacher model. Figure 9 shows the results of a controlled experiment to generate interpretation maps on the teacher model to check if they share similarities with benign interpretation maps. The results show that the attack achieved approximately a 60% fooling ratio of the teacher model (ResNet-50) and over 60% IoU score using CAM, which is more or less the same as the student model (VGG-16). The results of both experiments show the efficacy of Single**ADV** against black-box models.

VI. Countermeasures

In the following and based on our observations, we discuss several potential countermeasures against Single**ADV**.

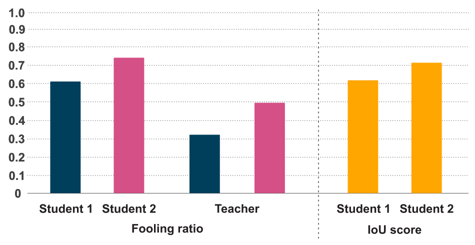

Fig. 8. The fooling ratio and IoU scores results are based on the universal perturbation generated by Single**ADV** following the teacher-student learning approach. A student refers to a white-box DL model (Student 1: VGG-11, Student 2: VGG-16), and a teacher refers to a black-box DL model used by an app. Source and target categories are selected randomly.

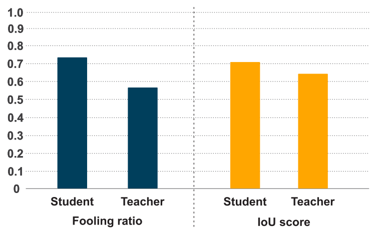

Fig. 9. The fooling ratio and IoU scores results are based on the universal perturbation generated by Single**ADV** following the teacher-student learning approach. A student refers to a white-box DL model (VGG-16), and a teacher refers to a black-box DL model (ResNet-50). IoU results are based on the attribution maps generated CAM interpreter.

A. Preprocessing Methods

Preprocessing approaches involve specific modifications to eliminate adversarial noise from samples before passing them to the DL model. The objective of preprocessing defenses is to enhance the robustness of DL models against adversarial samples, ensuring that the models can classify adversarial samples correctly with a small reduction in performance on benign images. It is important to note that relying on a single preprocessing defense technique may not be sufficient, as the attack can be adapted to bypass that particular defense. Therefore, employing multiple preprocessing techniques can help remove the perturbation added to the samples. By focusing on different features of the samples, these defense techniques make it more challenging to generate robust adversarial samples that can bypass these defenses.

To evaluate the effectiveness of defense techniques, including our proposed defense method, we apply them to adversarial samples and visualize their impact in Figure 10. These examples provide insights into how the defense models affect the adversarial samples and highlight the effectiveness of the defense techniques in mitigating the impact of Single**ADV**.

Table III provides the fooling ratio of Single**ADV** when a pair of defense methods is applied to preprocess the adversarial samples generated by Single**ADV**. In the experiment, three defense techniques were included, namely bit depth reduction [46], median smoothing [47], and random resizing and padding (R&P) [48]. Each defense technique was applied with its default hyperparameters. The results demonstrate that employing two defense techniques together decreases the effectiveness of the attack. It is worth noting that the performance of the defense methods can be further improved by adjusting the settings and hyperparameters of the defense techniques. We note that optimizing the parameters makes it possible to improve the performance of defense methods in mitigating the impact of Single**ADV**.

TABLE III

FOOLING RATIO OF ADVERSARIAL SAMPLES WHEN TWO DEFENSE TECHNIQUES ARE APPLIED. THE RESULTS ARE BASED ON 30 SAMPLES OF THE TARGETED CATEGORY (TARGET 2: DOG → **GOOSE**)

Pair of Defenses	Fooling Ratio
Bit-Depth Reduction - Median Smoothing	0.13
Median Smoothing - R&P	0.07
R&P - Bit-Depth Reduction	0.10

B. Ensemble Interpreter

The term "ensemble interpreter" refers to using a set of interpreters to provide a comprehensive view of a DL model. By employing multiple interpreters, each with its unique characteristics, a more comprehensive understanding of the DL model can be achieved [23]. Therefore adopting several interpreters can help detect if a sample is benign or adversarial. Future research direction can be in examining whether Single**ADV** can be adapted to counter IDLSes with ensemble interpreters. For example, the attack can be optimized to minimize the interpretation loss across multiple interpreters used by an IDLS. However, it is worth mentioning that the generation process of adversarial samples in Single**ADV** can be computationally expensive against IDLSes with ensemble interpreters. Therefore, the number of interpreters employed in an IDLS can play a significant role in defending against Single**ADV**.

C. Interpretation-based Adversarial Training (IbAT)

The objective of this task is to develop robust classifiers that can effectively handle universal perturbations. To this end, we adopt a similar approach to the one in [49] and leverage concepts from robust optimization. Our strategy involves formulating the problem of universal adversarial training as a min-max optimization problem to construct highly resilient models to universal perturbations. By treating the generation of universal perturbations as an optimization task, we aim to find the optimal perturbation that maximally affects the model's robustness while minimizing its impact on the model's performance on benign inputs. To solve this optimization problem, we utilize alternating stochastic gradient methods.

Algorithm 2 uses a single perturbation refined throughout all iterations.

We only update the weights w and perturbations p once per training step.

In the algorithm, the universal perturbation is cropped by identifying significant areas. This is achieved by converting the interpretation mask $m \in M$ into binary form $m_o = O_t(m)$ using threshold $t = 0.3$ so that a value of 0 indicates an irrelevant area and 1 indicates a relevant area. The mask is then multiplied with the universal perturbation p using the operator $\odot$ for element-wise multiplication.

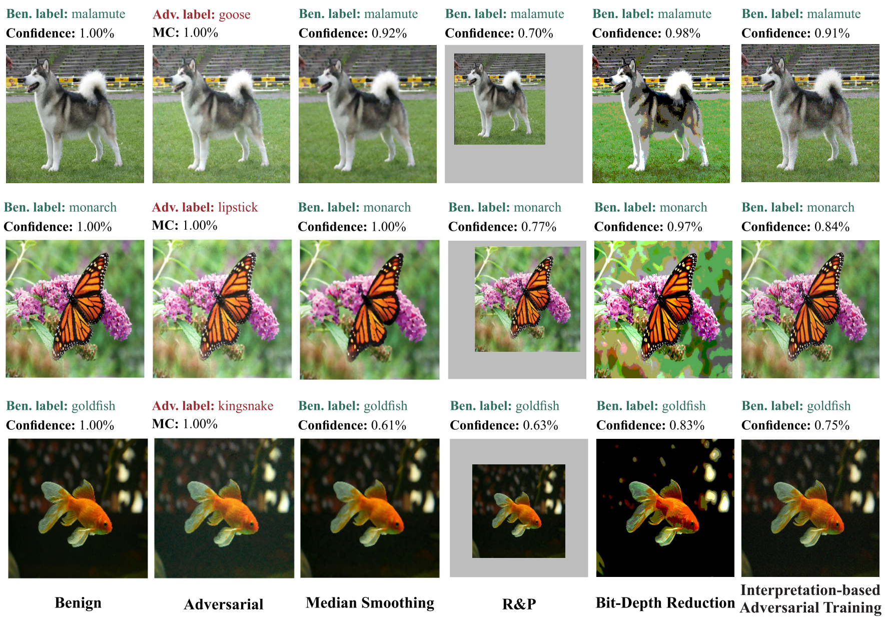

Fig. 10. Examples demonstrating the influence of defense mechanisms on adversarial samples. Ben. label and Adv. label represent benign and adversarial labels, respectively. MC stands for misclassification confidence.

Algorithm 2 Interpretation-Based Adversarial Training Against Universal Perturbations

```
Data: Samples $X$, perturbation threshold $\epsilon$, learning
      rate $\alpha$, momentum $\mu$, perturbation $p$,
      interpretation masks $M$
1  for epoch $i = 1 \dots N$ do
2      for minibatch $b \subset X$ and $I \subset M$ do
3          $M_o = O_t(I)$  # Binarization using threshold $t$
4          # weight update based on expected norms $\mathbb{E}$ of
             the gradient $\nabla$
5          $g_w \leftarrow \mu g_w - \mathbb{E}_{x \in B\ m \in M_o}[\nabla_w l(w, x + m \odot p)]$;
6          $w \leftarrow w + \alpha g_w$;
7          # perturbation update
8          $p \leftarrow$
             $p + \epsilon\ sign\big(\mathbb{E}_{x \in B\ m \in M_o}\big[\nabla_p L(w, x + m \odot p)\big]\big)$;
9          Projecting $p$ onto the $p$-ball.
10     end
11 end
```

TABLE IV
FOOLING RATIOS AND LEAKAGE RATES OF SINGLE**ADV** USING CIFAR-10 AND IMAGENET DATASETS BEFORE AND AFTER INTERPRETATION-BASED ADVERSARIAL TRAINING. THE RESULTS WERE OBTAINED USING A SAMPLE SIZE OF 200 PER CATEGORY (2,000 TOTAL) FOR CIFAR-10 AND 50 PER CATEGORY (50,000 TOTAL) FOR IMAGENET

#	Source	Target	Fooling Ratio (Before)	Fooling Ratio (After)	Leakage Rate (Before)	Leakage Rate (After)
			CIFAR-10			
1	Bird	Airplane	0.77	0.15	0.39	0.10
2	Deer	Frog	0.84	0.16	0.40	0.12
3	Frog	Cat	0.87	0.17	0.41	0.14
4	Ship	Cat	0.83	0.16	0.38	0.13
5	Truck	Horse	0.78	0.15	0.39	0.10
6	Airplane	Deer	0.80	0.16	0.37	0.12
7	Horse	Dog	0.84	0.16	0.40	0.14
8	Dog	Frog	0.69	0.13	0.38	0.12
			ImageNet			
1	Panda	Cat	0.78	0.14	0.34	0.11
2	Dog	Goose	0.75	0.15	0.33	0.11
3	Cup	Wolf	0.69	0.14	0.36	0.12

This approach enables the model to learn and become more robust to the perturbation within the interpretation mask. For the CIFAR-10 experiment, we use a perturbation threshold ϵ of 0.031, a batch size of 128, Momentum SGD with an initial learning rate of 0.1 that drops until 0.001 and trained for 500 epochs for the ResNet-20 model. The selection of the CIFAR-10 dataset and the ResNet-20 model for adversarial training and testing was driven by practical considerations such as performance, computational efficiency, standardization, and generalization properties. For the ImageNet experiment, we use pre-calculated universal perturbations (*i.e.,* using the DenseNet-169 model with CAM interpreter from the experiments presented in Table II), as a computationally efficient alternative. To enable adversarial training, we use a fine-tuning approach to learn relevant features in the presence of perturbations efficiently. We use a batch size of 32, Momentum SGD with an initial learning rate of 0.1 that drops until 0.001, a perturbation threshold of 0.04 and 100 epochs.

In our experiments, we randomly select the source and target categories for the CIFAR-10 dataset. However, when using the ImageNet dataset, we utilize previously calculated universal perturbations for the targets described in Table II. To evaluate the effectiveness of IbAT, we conduct experiments and measure the fooling ratio and leakage rate before and after applying adversarial training. Table IV shows that the fooling ratio and leakage rate decrease significantly after applying IbAT. This indicates that the customized adversarial training approach improves the classifier's robustness against the attack on both source and non-source categories. IbAT offers a unique approach by enhancing the robustness of models against dual adversarial attacks aimed to fool the models and their interpretations, a benefit not typically addressed in traditional adversarial training methods.

VII. Limitations

Table II, Figure 5 and Figure 8 show that the proposed technique is efficient and effective. However, Figure 6 highlights a limitation in the similarity between the adversarial interpretation maps generated by the MASK interpreter and their benign counterparts. This can be attributed to the construction of the MASK interpreter as an optimization procedure, where even small amounts of noise can lead to variations in the interpretation maps. Despite this limitation, the interpretation maps generated by the MASK interpreter still possess meaningful information, which can deceive the human-involvement process (e.g., expert-checking), as depicted in Figure 5.

Another limitation of the proposed attack is the potential impact of the universal perturbation on the classification accuracy of non-source classes. The attack may interfere with the class predictions of different categories, resulting in misclassifications. While the paper focused on the efficiency of the attack regarding the source and target categories, the impact on non-source classes was not extensively considered. One possible solution to address this limitation is to increase the classification confidence of non-source classes for their actual categories while ensuring that they are not misclassified. This can be a potential direction for future research in mitigating the impact of the attack on non-source classes.

VIII. Conclusion

In this paper, we present Single**ADV**, a targeted adversarial attack that specifically aims to deceive deep learning models in a single object category while minimizing the impact on other categories in both white-box and black-box settings. The proposed attack exploits interpretations to generate universal perturbations that produce adversarial examples with similar interpretations to benign inputs, making them more difficult to detect. Through extensive experiments, we evaluated the effectiveness of Single**ADV** on four popular deep learning models (*i.e.,* VGG-16, ResNet-50, DenseNet-169, and Inception-V3) and three interpretation models (*i.e.,* CAM, Grad, and MASK), demonstrating the vulnerability of interpretable deep learning systems to our proposed attack. We explored existing defense techniques to mitigate the impact of such adversarial attacks. Our experiments show that interpretation-based adversarial training can enhance the robustness of the models against interpretation-guided attacks.

A. Future Work

Possible future research directions include investigating the countermeasures and limitations discussed in the paper and exploring the potential applications of the proposed attack and defenses in other domains. Future studies can also focus on studying the effects of the attack given different model architectures and types of interpretation models.

References

[1] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[4] Y. He, G. Meng, K. Chen, X. Hu, and J. He, "Towards security threats of deep learning systems: A survey," *IEEE Trans. Softw. Eng.*, vol. 48, no. 5, pp. 1743–1770, May 2022.

[5] P.-J. Kindermans et al., "The (un) reliability of saliency methods," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Cham, Switzerland: Springer, 2019, pp. 267–280.

[6] 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. (CVPR)*, Jun. 2016, pp. 2921–2929.

[7] Q. Zhang, Y. N. Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8827–8836.

[8] E. Abdukhamidov, M. Abuhamad, F. Juraev, E. Chan-Tin, and T. AbuHmed, "AdvEdge: Optimizing adversarial perturbations against interpretable deep learning," in *Computational Data and Social Networks*. Montreal, QC, Canada: Springer, 2021, pp. 93–105.

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

[10] N. Akhtar, M. A. A. K. Jalwana, M. Bennamoun, and A. Mian, "Attack to fool and explain deep networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 5980–5995, Oct. 2022.

[11] E. Abdukhamidov, M. Abuhamad, S. S. Woo, E. Chan-Tin, and T. Abuhmed, "Interpretations cannot be trusted: Stealthy and effective adversarial perturbations against interpretable deep learning," 2022, *arXiv:2211.15926*.

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

[13] I. Oseledets and V. Khrulkov, "Art of singular vectors and universal adversarial perturbations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8562–8570.

[14] J. Hayes and G. Danezis, "Learning universal adversarial perturbations with generative models," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, May 2018, pp. 43–49.

[15] J. Heo, S. Joo, and T. Moon, "Fooling neural network interpretations via adversarial model manipulation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.

[16] J. Weng, Z. Luo, Z. Zhong, D. Lin, and S. Li, "Exploring non-target knowledge for improving ensemble universal adversarial attacks," in *Proc. AAAI Conf. Artif. Intell.*, 2023, vol. 37, no. 3, pp. 2768–2775.

[17] X. Liu, Y. Zhong, Y. Zhang, L. Qin, and W. Deng, "Enhancing generalization of universal adversarial perturbation through gradient aggregation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2023, pp. 4435–4444.

[18] Z. Ye, X. Cheng, and X. Huang, "FG-UAP: Feature-gathering universal adversarial perturbation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jun. 2023, pp. 1–8.

[19] G. Tao, S. An, S. Cheng, G. Shen, and X. Zhang, "Hard-label black-box universal adversarial patch attack," in *Proc. 32nd USENIX Secur. Symp. (USENIX Security)*. Anaheim, CA, USA: USENIX Association, Aug. 2023, pp. 697–714.

[20] F. Qin, W. Na, S. Gao, and S. Yao, "Sigma-UAP: An invisible semi-universal adversarial attack against deep neural networks," in *Proc. Int. Conf. Pattern Recognit., Mach. Vis. Intell. Algorithms (PRMVIA)*, Mar. 2023, pp. 34–41.

[21] H. E. Oskouie and F. Farnia, "Interpretation of neural networks is susceptible to universal adversarial perturbations," in *Proc. ICASSP - IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.

[22] A. Ghorbani, A. Abid, and J. Zou, "Interpretation of neural networks is fragile," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 3681–3688.

[23] 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.*, early access, pp. 1–13, 2023.

[24] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," 2017, *arXiv:1710.09829*.

[25] W. J. Murdoch, P. J. Liu, and B. Yu, "Beyond word importance: Contextual decomposition to extract interactions from LSTMs," 2018, *arXiv:1801.05453*.

[26] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," 2017, *arXiv:1705.07857*.

[27] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014.

[28] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 3145–3153.

[29] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smooth-Grad: Removing noise by adding noise," 2017, *arXiv:1706.03825*.

[30] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.

[31] R. C. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3429–3437.

[32] J. Chen, J. Raghuram, J. Choi, X. Wu, Y. Liang, and S. Jha, "Revisiting adversarial robustness of classifiers with a reject option," in *Proc. AAAI Workshop Adversarial Mach. Learn. Beyond*, 2021.

[33] D. Stutz, M. Hein, and B. Schiele, "Confidence-calibrated adversarial training: Generalizing to unseen attacks," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9155–9166.

[34] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[35] T. DeVries and G. W. Taylor, "Learning confidence for out-of-distribution detection in neural networks," 2018, *arXiv:1802.04865*.

[36] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4700–4708.

[37] 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. (CVPR)*, Jun. 2016, pp. 2818–2826.

[38] W. Zhou, 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.

[39] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2961–2969.

[40] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[41] E. Abdukhamidov, F. Juraev, M. Abuhamad, and T. Abuhmed, "Black-box and target-specific attack against interpretable deep learning systems," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, 2022, pp. 1216–1218.

[42] H. Cao, S. Li, Y. Zhou, M. Fan, X. Zhao, and Y. Tang, "Towards black-box attacks on deep learning apps," 2021, *arXiv:2107.12732*.

[43] *Soot-oss/Soot: Soot—A Java Optimization Framework*. Accessed: Feb. 2023. [Online]. Available: https://github.com/soot-oss/soot

[44] Secure-Software-Engineering. *Secure Software Engineering/Flowdroid: Flowdroid Static Data Flow Tracker*. Accessed: Feb. 2023. [Online]. Available: https://github.com/secure-software-engineering/FlowDroid

[45] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1139–1147.

[46] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, "Countering adversarial images using input transformations," 2017, *arXiv:1711.00117*.

[47] G. W. Ding, L. Wang, and X. Jin, "Advertorch v0.1: An adversarial robustness toolbox based on PyTorch," 2019, *arXiv:1902.07623*.

[48] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," 2017, *arXiv:1711.01991*.

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