SingleADV: 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 Abuhamed®, 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

D EEP 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 feature importance (i.e., identifying parts of the input that are most significant for the model’s predictions); 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 (IDLSeSes), is essential to understand the behavior of models and ensure trustworthiness in detecting adversarial input. IDLSeSes 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 SingleADV, a single-class target-specific adversarial attack method designed to generate targeted perturbations that deceive IDLSeSes 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 IDLSeSes to detect the attack. In a targeted attack threat model, SingleADV generates stealthy perturbations that effectively deceive IDLSeSes, therefore hindering human involvement in analyzing the interpretation of...
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 SingleADV 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. SingleADV 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 SingleADV, 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 SingleADV 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 SingleADV, 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 SingleADV 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.
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 SingleADV and previous studies are summarized in Table I.

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.
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 Class Activation Map (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 \( \sum_j a_i(j, k) \) denote the outcome of global average pooling. So, the softmax function receives: \( \psi_y(x) = \sum_j \sum_k w_{i,y} a_i(j, k) \), 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 \( m_y = \frac{\partial f_y(x)}{\partial x} \). 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 \( \tilde{x} \), the gradient of the ReLU \( R(z) \) function is approximated as:

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

where \( h(z) \) approximates the gradient of ReLU \( R(z) \) with a small constant parameter \( \tau \) (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:

\[
\min_{\text{mask}} : f_y(\phi(x; \text{mask})) + \lambda \parallel 1 - \text{mask} \parallel \quad s.t. \quad 0 \leq \text{mask} \leq 1,
\]

where \( \phi(x; \text{mask}) \) is the operator that generates perturbation to decrease the probability of the current prediction category \( y \) and the second term \( \lambda \parallel 1 - \text{mask} \parallel \) 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:

\[
f(s + p) \rightarrow y_i, \text{where } y \neq y_i, \parallel p \parallel_\infty \leq \eta
\]
In Equation (2), confining $y_t$ to a chosen category and $\ell_p$-norm to a predefined value of $\eta$ 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 $|\cdot|$ denotes the cardinality of a set, we maximize the objective of Equation (2) as follows:

$$\max_p P_{D(p)}(f(s + p) \rightarrow y_t),$$

where:

$$P_{D(p)}(f(s + p) \rightarrow y_t) \geq \gamma, \|p\|_p \leq \eta,$$

and $|D(p)| \geq 1$ (3)

In Equation (3), $P$ 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 $\gamma$ 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 ensure that the model’s prediction confidence of the adversarial sample remains high enough to avoid two potential issues: the rejection of the predicted confidence of the adversarial sample remains high enough

2) Rationale for Perturbations: In Equation (3), $P$ 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.

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. $\ell_\infty$ norm is applied to bound 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 $g$, $\eta$ for the $\ell_p$-norm of the perturbation, target category $y_t$, batch size $b$, adversarial confidence $\gamma$ (confidence level as a target category), pre-set first- and second-moment hyperparameters, $\beta_1$ and $\beta_2$, the algorithm starts the initialization process by setting $p_0$, $\nu_0$, $\omega_0 \in \mathbb{R}^d$ as zero vectors.

The algorithm first randomly samples the selected category and the other categories in sets $S_c$ 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 $g(S_c; f)$ and $g(S_o; f)$, respectively (line 4).

In line 5, all sets are perturbed by subtracting the currently estimated perturbation $p_t$ 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 $E$ in the algorithm) of the gradients of

- Triggering an interpreter $g$ to generate target attribution maps: $g(s + p; f) \xrightarrow{\text{source}} m_t$, such that $g(s; f) \rightarrow m_t$.

We focus on the single-class attack in our research for the following reasons.

1. Targeting a specific class allows fine-tuning adversarial perturbations to exploit unique vulnerabilities, leading to a higher success rate and more controlled optimization.

2. 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.

3. Adversarial samples can appear less suspicious when targeting a single class, creating an additional layer of stealth and posing challenges in detecting such samples.
Algorithm 1 SingleADV: Attack’s Main Algorithm

Data: Target model \( f \), interpreter \( g \), selected category samples \( S \), non-selected category samples \( \bar{S} \) \(\text{s.t. } S_i \neq \bar{S}_i \in \mathbb{R}^d \), target category \( y_t \), perturbation norm \( \eta \), balance factor \( \lambda \), batch size \( b \), and adversarial confidence \( \gamma \).

Result: Universal Adversarial Perturbation \( p \in \mathbb{R}^d \)

1. **Initialization:** \( i = 0 \), and setting \( p_0, \psi_0, \omega_0 \in \mathbb{R}^d \)
2. while adversarial confidence \( \gamma \) do
   3. \( S_i \leftarrow S \), \( \bar{S}_i \leftarrow \bar{S} \) \(\text{s.t. } |S_i| = |\bar{S}_i| = \frac{b}{2} \)
   4. \( M \leftarrow g(S_x; f) \) and \( \bar{M} \leftarrow g(\bar{S}_x; f) \)
   5. \( X \leftarrow C(S_x \oplus p_i), \bar{X} \leftarrow C(\bar{S}_x \oplus p_i) \)
   6. \( i \leftarrow i + 1 \)
   7. \( \hat{\theta} \leftarrow \frac{1}{2} \left( \mathbb{E}_{i \in X_m \in M} \left( \nabla_{x_i} l_{prd}(f(x_i, y_i)) + \lambda \cdot l_{int}(g(x_i; f, m_i)) \right) \right) \)
   8. \( \psi_i \leftarrow -\frac{1}{2} \mathbb{E}_{i \in \bar{X}_m \in \bar{M}} \left( \nabla_{x_i} l_{prd}(f(x_i, y_i)) + \lambda \cdot l_{int}(g(x_i; f, m_i)) \right) \)
   9. \( \psi_i \leftarrow \beta_1 \psi_{i-1} + (1 - \beta_1) \psi_i \)
   10. \( \omega_i \leftarrow \beta_2 \omega_{i-1} + (1 - \beta_2) \psi_i \odot \psi_i \)
   11. \( \hat{\rho} \leftarrow \frac{1 - \beta_2}{1 - \beta_1} \cdot \text{diag} \left( \text{diag} (\omega_i) \right)^{-1} \psi_i \)
   12. \( p_i \leftarrow p_{i-1} + \frac{\hat{\rho}}{\|\hat{\rho}\|_{\infty}} \)
   13. \( p_i \leftarrow \text{sign}(p_i) \odot \text{min}(|p_i|, \eta) \)
   14. end

The objective of SingleADV 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. SINGLEADV 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 \( \geq 60\% \) 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: 1) Is our technique effective in attacking DL models by targeting a single category? 2) 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 (Intersection-over-Union):** 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: $\text{IoU}(m, m_o) = \frac{\text{area}(O(m) \cap O(m_o))}{\text{area}(O(m) \cup O(m_o))}$, where $m$ denotes the binary attribution map of samples with the universal perturbation, and $m_o$ represents the binary attribution map of samples without perturbation. Here, $O(\cdot)$ 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 SingleADV 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 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.

<table>
<thead>
<tr>
<th>Interpreter</th>
<th>Model</th>
<th>Target 1 Misclassification Confidence</th>
<th>Leakage Rate</th>
<th>Fooling Ratio</th>
<th>Target 2 Misclassification Confidence</th>
<th>Leakage Rate</th>
<th>Fooling Ratio</th>
<th>Target 3 Misclassification Confidence</th>
<th>Leakage Rate</th>
<th>Fooling Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAM</td>
<td>VGG-16</td>
<td>0.85</td>
<td>0.82</td>
<td>0.30</td>
<td>0.87</td>
<td>0.65</td>
<td>0.55</td>
<td>0.71</td>
<td>0.80</td>
<td>0.32</td>
</tr>
<tr>
<td></td>
<td>ResNet-50</td>
<td>0.90</td>
<td>0.84</td>
<td>0.24</td>
<td>0.79</td>
<td>0.54</td>
<td>0.34</td>
<td>0.75</td>
<td>0.79</td>
<td>0.35</td>
</tr>
<tr>
<td></td>
<td>DenseNet-169</td>
<td>0.78</td>
<td>0.81</td>
<td>0.34</td>
<td>0.75</td>
<td>0.60</td>
<td>0.33</td>
<td>0.76</td>
<td>0.77</td>
<td>0.34</td>
</tr>
<tr>
<td></td>
<td>Inception-V3</td>
<td>0.63</td>
<td>0.79</td>
<td>0.30</td>
<td>0.62</td>
<td>0.31</td>
<td>0.22</td>
<td>0.63</td>
<td>0.41</td>
<td>0.27</td>
</tr>
<tr>
<td>Grad</td>
<td>VGG-16</td>
<td>0.87</td>
<td>0.78</td>
<td>0.34</td>
<td>0.65</td>
<td>0.56</td>
<td>0.35</td>
<td>0.75</td>
<td>0.77</td>
<td>0.31</td>
</tr>
<tr>
<td></td>
<td>ResNet-50</td>
<td>0.83</td>
<td>0.88</td>
<td>0.34</td>
<td>0.81</td>
<td>0.37</td>
<td>0.37</td>
<td>0.72</td>
<td>0.78</td>
<td>0.37</td>
</tr>
<tr>
<td></td>
<td>DenseNet-169</td>
<td>0.60</td>
<td>0.84</td>
<td>0.33</td>
<td>0.73</td>
<td>0.35</td>
<td>0.35</td>
<td>0.76</td>
<td>0.78</td>
<td>0.35</td>
</tr>
<tr>
<td></td>
<td>Inception-V3</td>
<td>0.72</td>
<td>0.80</td>
<td>0.31</td>
<td>0.68</td>
<td>0.35</td>
<td>0.30</td>
<td>0.69</td>
<td>0.72</td>
<td>0.30</td>
</tr>
<tr>
<td>MASK</td>
<td>VGG-16</td>
<td>0.81</td>
<td>0.79</td>
<td>0.34</td>
<td>0.76</td>
<td>0.34</td>
<td>0.72</td>
<td>0.70</td>
<td>0.33</td>
<td>0.76</td>
</tr>
<tr>
<td></td>
<td>ResNet-50</td>
<td>0.82</td>
<td>0.82</td>
<td>0.37</td>
<td>0.74</td>
<td>0.58</td>
<td>0.70</td>
<td>0.72</td>
<td>0.34</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td>DenseNet-169</td>
<td>0.77</td>
<td>0.83</td>
<td>0.37</td>
<td>0.71</td>
<td>0.68</td>
<td>0.70</td>
<td>0.75</td>
<td>0.34</td>
<td>0.72</td>
</tr>
<tr>
<td></td>
<td>Inception-V3</td>
<td>0.69</td>
<td>0.76</td>
<td>0.30</td>
<td>0.64</td>
<td>0.36</td>
<td>0.63</td>
<td>0.70</td>
<td>0.31</td>
<td>0.65</td>
</tr>
<tr>
<td>N/A [10]</td>
<td>VGG-16</td>
<td>0.85</td>
<td>0.84</td>
<td>0.31</td>
<td>0.82</td>
<td>0.76</td>
<td>0.82</td>
<td>0.81</td>
<td>0.34</td>
<td>0.81</td>
</tr>
<tr>
<td></td>
<td>ResNet-50</td>
<td>0.83</td>
<td>0.78</td>
<td>0.36</td>
<td>0.80</td>
<td>0.32</td>
<td>0.73</td>
<td>0.79</td>
<td>0.37</td>
<td>0.79</td>
</tr>
<tr>
<td></td>
<td>DenseNet-169</td>
<td>0.78</td>
<td>0.77</td>
<td>0.37</td>
<td>0.78</td>
<td>0.38</td>
<td>0.73</td>
<td>0.78</td>
<td>0.38</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td>Inception-V3</td>
<td>0.75</td>
<td>0.87</td>
<td>0.39</td>
<td>0.73</td>
<td>0.37</td>
<td>0.68</td>
<td>0.78</td>
<td>0.40</td>
<td>0.72</td>
</tr>
</tbody>
</table>

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

According to the leakage rate analysis, SingleADV 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 SingleADV 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, SingleADV 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. 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, SingleADV produces perturbations that only affect
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 \( \geq 0.5 \) 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 SingleADV 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. SingleADV in Black-Box Settings

Since white-box attacks have limited practicality in real-life cases, we also employ our proposed method in black-box
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 SingleADV with a wider range of models and interpretation tools.

A. Methodology

To apply SingleADV 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 SingleADV against black-box models.

VI. COUNTERMEASURES

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

Table III provides the fooling ratio of SingleADV when a pair of defense methods is applied to preprocess the adversarial samples generated by SingleADV. 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 SingleADV.

### 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 SingleADV 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 SingleADV 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 SingleADV.

### 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 process of adversarial samples 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 irrelevant area and 1 indicates a relevant area. The mask is used to enhance the robustness of DL models against adversarial samples. It is important to note that relying on a single pre-processing 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 SingleADV.

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 irrelevant area and 1 indicates a relevant area. The mask is used to enhance the robustness of DL models against adversarial samples. It is important to note that relying on a single pre-processing 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 SingleADV.

### Table III

<table>
<thead>
<tr>
<th>Pair of Defenses</th>
<th>Fooling Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit-Depth Reduction - Median Smoothing</td>
<td>0.13</td>
</tr>
<tr>
<td>Median Smoothing - R&amp;P</td>
<td>0.07</td>
</tr>
<tr>
<td>R&amp;P - Bit-Depth Reduction</td>
<td>0.10</td>
</tr>
</tbody>
</table>

Fig. 8. The fooling ratio and IoU scores results are based on the universal perturbation generated by SingleADV 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 SingleADV following the teacher-student learning approach. A student refers to a white-box DL model (ResNet-50), and a teacher refers to a black-box DL model used by an app. Source and target categories are selected randomly.
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 $\epsilon$ 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.

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 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 SingleADV, 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 SingleADV 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.


