Amnesia as a Catalyst for Enhancing Black Box Pixel Attacks in Image Classification and Object Detection

Dongsu Song
Department of Artificial Intelligence
Korea Aerospace University
raister2873@gmail.com

Daehwa Ko
Department of Software
Korea Aerospace University
daehwa001210@gmail.com

Jay Hoon Jung*
Department of Artificial Intelligence
Korea Aerospace University
jhjung@kau.ac.kr

Abstract

It is well known that query-based attacks tend to have relatively higher success rates in adversarial black-box attacks. While research on black-box attacks is actively being conducted, relatively few studies have focused on pixel attacks that target only a limited number of pixels. In image classification, query-based pixel attacks often rely on patches, which heavily depend on randomness and neglect the fact that scattered pixels are more suitable for adversarial attacks. Moreover, to the best of our knowledge, query-based pixel attacks have not been explored in the field of object detection. To address these issues, we propose a novel pixel-based black-box attack called **R**emember and **F**orget **P**ixel **A**ttack using **R**einforcement Learning(RFPAR), consisting of two main components: the Remember and Forget processes. RFPAR mitigates randomness and avoids patch dependency by leveraging rewards generated through a one-step RL algorithm to perturb pixels. RFPAR effectively creates perturbed images that minimize the confidence scores while adhering to limited pixel constraints. Furthermore, we advance our proposed attack beyond image classification to object detection, where RFPAR reduces the confidence scores of detected objects to avoid detection. Experiments on the ImageNet-1K dataset for classification show that RFPAR outperformed state-of-the-art query-based pixel attacks. For object detection, using the MS-COCO dataset with YOLOv8 and DDQ, RFPAR demonstrates comparable mAP reduction to state-of-the-art query-based attack while requiring fewer query. Further experiments on the Argoverse dataset using YOLOv8 confirm that RFPAR effectively removed objects on a larger scale dataset. Our code is available at https://github.com/KAU-QuantumAILab/RFPAR.

1 Introduction

Deep learning models are susceptible to adversarial attacks, which involve subtle modifications of input data that are imperceptible to humans but lead to incorrect predictions by the model[1]. As deep learning technologies become commercialized in the real world, the issue of adversarial attacks has garnered increasing attention.

*Corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-2766

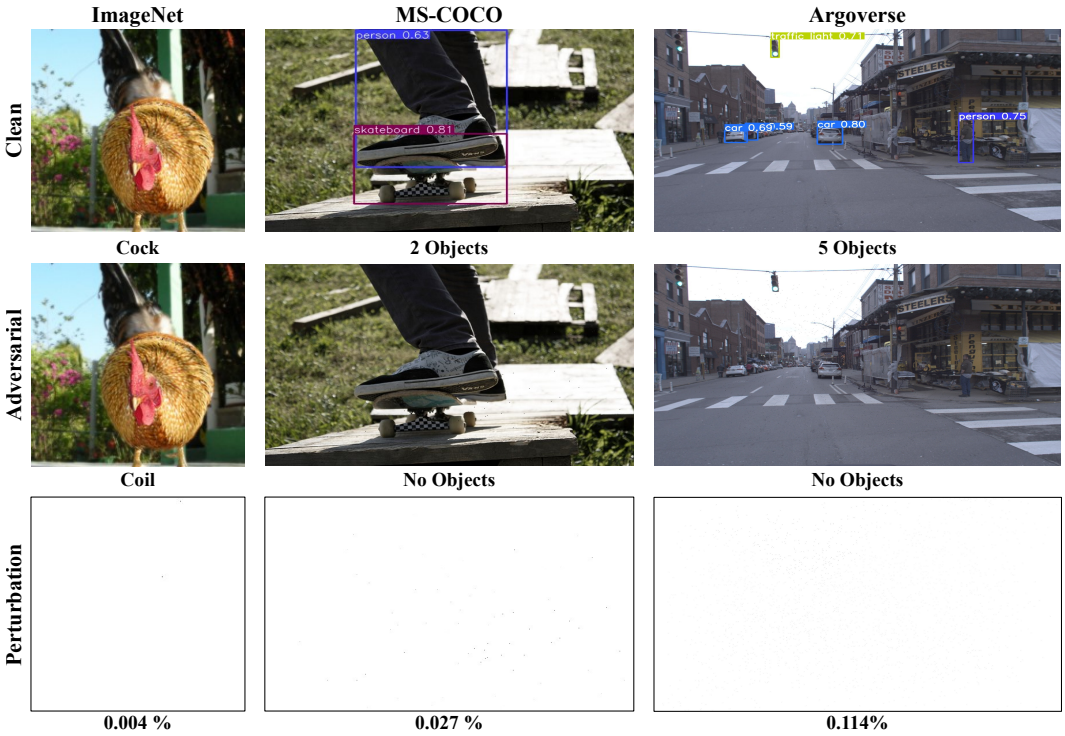

Figure 1: **Adversarial examples generated by RFPAR.** The first column represents images from ImageNet (image classification), the second column from MS-COCO (object detection), and the third column from Argoverse (object detection). Each row represents a different condition: the first row shows clean images, the second row shows adversarially perturbed images, and the third row shows the perturbation levels with the ratio of attacked pixels to total pixels. Labels in the images indicate detected objects or classifications, such as "Cock" in ImageNet, "2 Objects" in MS-COCO, and "5 Objects" in Argoverse. In the adversarial row, labels are altered due to perturbations, resulting in misclassifications or undetected objects, such as "Coil" instead of "Cock" in ImageNet and no objects detected in MS-COCO and Argoverse. The perturbation row indicates the percentage of pixels attacked in the image. The percentages were 0.004% for ImageNet, 0.027% for MS-COCO, and 0.114% for Argoverse.

Adversarial attacks can be broadly categorized into white-box attacks and black-box attacks[2]. In white-box attacks[3, 4, 5], attackers devise attack strategies based on internal information about deep learning models, such as training data, gradients of the outputs with respect to the weights, and other details about the learning process for given samples. Conversely, in black-box attacks[6, 7], attackers can access only limited information such as the probability of the correct prediction for a given sample. Given that real-world attackers typically only possess limited information about the model, black-box attacks are more realistic than white-box attacks. In other words, research on black-box attacks and their defenses is crucial in order to develop robust and secure machine learning systems.

Black-box attacks are also categorized into query-based methods[8, 9, 10, 11] and transfer-based methods[12]. Query-based attacks are generating adversarial examples by repeatedly querying the victim model with modified images[13]. Transfer-based attacks involve generating adversarial examples for a surrogate model that successfully deceive another model[14]. Transfer-based attacks are highly efficient since they do not require knowledge of the victim model. However, the discrepancies in model architecture, training data, and training methodologies between the surrogate and victim models often result in a lower success rate for these attacks compared to query-based attacks[13]. Conversely, although query-based attacks achieve higher success rates, they require a significant number of queries to the victim model. Therefore, reducing the number of queries in query-based attacks is a critical issue.

https://doi.org/10.52202/079017-2766

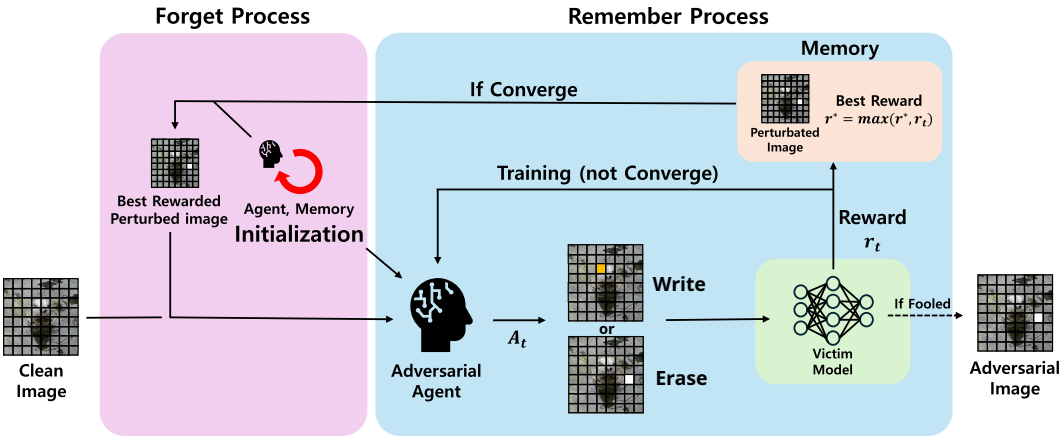

Figure 2: The model architecture of RFPAR: the Remember and Forget process. During the Remember process, the RL model generates perturbed images and corresponding rewards. Memory compares these with previously stored values and retains only the highest reward and its associated image. Once the rewards converge to a certain value, the Forget process starts and resets the RL agent and memory, then reintroduces the perturbed images that gained the highest reward to the Remember process. The process continues until an adversarial image is generated or a predefined number of cycles is reached, at which point it terminates.

The pioneering pixel attack method, OnePixel[8], employed Differential Evolution (DE) to generate adversarial images. An advanced approach, ScratchThat[9], used DE to create curves and applied a parametric model to perturbations, reducing parameters and improving performance. A more recent study, PIXLE[11], enhanced query efficiency and attack success rate by using a simple algorithm instead of DE. Briefly, PIXLE generates adversarial images by selecting arbitrary patches in a clean image and applying the brightness of these pixels to others. Although this method improved performance, it ignored the fact that pixels are independent of each other due to its reliance on patches and exhibited inefficiencies stemming from randomness in brightness mapping. The previous study, PatchAttack[15], utilized RL model to embed textures in specific regions of the clean image, discovering vulnerable patches and reducing randomness, which significantly decreased the number of queries and improved attack success rates. Unfortunately, this method still depended on patches, requiring at least 3% of the image area to be attacked.

Query-based attacks in object detection are more challenging than those in image classification. The first query-based attack in object detection, PRFA[16], generated adversarial images using a parallel rectangle flipping strategy. Recent research, GARSDC[17], employed a genetic algorithm to create adversarial images, improving optimization efficiency by using adversarial examples generated from transfer-based attacks as the initial population. Query-based attacks on black-box models are inherently challenging, and targeting only a few pixels is even more difficult to study. To the best of our knowledge, pixel attacks have been limited to white-box or transfer attack methods[18, 19, 20, 21]. In this study, we extend our proposed attack from image classification to object detection, introducing the first query-based pixel attack. Experiments show that our method achieves a comparable mAP reduction on YOLO[22] to state-of-the-art methods while significantly reducing the number of queries, demonstrating its effectiveness in object detection.

In this study, we introduce a novel method called the Remember and Forget Pixel Attack using Reinforcement Learning (RFPAR). Briefly, in the Remember process, the clean image is initially taken as input by the RL agent, and the loss function is optimized. During this optimization, the highest reward and its corresponding perturbed image are stored in memory. If the highest rewards do not change for a while, we define this as the convergence of rewards. Once the rewards converge, the Forget process is initiated, resetting the RL agent and memory to forget previous information. After resetting, the stored image is fed as input to the RL agent, and the Remember process begins again. Extensive experiments demonstrate that our attack is effective for image classification and successfully extends to object detection.

In summary, our main contributions are:

https://doi.org/10.52202/079017-2766

- We propose a novel query-based black-box pixel attack consisting of the Remember and Forget processes. Our approach outperformed state-of-the-art attacks on the ImageNet-1K classification task, achieving an average attack success rate improvement of 12.1%, while reducing the number of queries by 26.0% and the modified L_0 norm by 41.1%.
- We advance query-based pixel attacks from image classification to object detection, introducing the query-based pixel attack. Our experiments demonstrate that our proposed method effectively compromises object detection systems. It achieves an average mean Average Precision (mAP) reduction of 0.29 in YOLO, comparable to state-of-the-art query-based attacks, while reducing the number of queries by 52.8%. To the best of our knowledge, the proposed method is the first black-box query-based pixel attack for object detection.
- To evaluate performance on a larger scale dataset, we conducted experiments using YOLOv8 as the victim model on the Argoverse-1.1 validation dataset. We also achieved a high removal rate of detected objects above 0.9 in Argoverse, similar to our results in MS-COCO. The results demonstrate that our proposed method effectively reduces the number of detected objects in images with a resolution of 1920×1200. Additionally, examining the adversarial perturbation results of RFPAR on Argoverse and MS-COCO, we observe that it successfully reduces the number of detected objects while attacking only very small areas of 0.1% and 0.02%, respectively.

2 Remember and Forget Pixel Attack Using Reinforcement Learning

In this section, we introduce our proposed method. In Section 2.1, we define the problem mathematically. Section 2.2 details the Remember process, which is the internal iterative structure of our algorithm, including agent, environment, and memory. Finally, we explain the Forget process, which serves as the external iterative structure in Section 2.3.

2.1 The Problem Formalization

We consider an image classifier as $f : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^c$, where C, H, and W represent the channel, height, and width of a given sample x, respectively, and c denotes the number of classes. The classifier f computes probabilities for each class for a sample x. Furthermore, $f_l(x)$ is defined as the probability of a sample x being predicted as the l-th class. The prediction of the classifier can be expressed as $\arg\max_l f_l(x)$. For an image classification attack, the objective is to minimize a pixel perturbation δ such that the classifier's prediction for an input x diverges from its true label y. This problem can be formalized as:

$$\begin{aligned} \min_{\delta} \quad & \arg\max_{l} f_l(x+\delta) \neq y \\ \text{s.t.} \quad & \|\overline{x} - x\|_0 = \|\delta\|_0 \leq \epsilon, \quad \epsilon \in \mathbb{Z}^+. \end{aligned} \tag{1}$$

Here, $\epsilon \in \{1, 2, \cdots\}$ and $\|\cdot\|_0$ denotes the attack level and the L_0 norm. $\overline{x}$ is the perturbed image, defined as $x+\delta$. Notably, the attack levels are positive integers, which implies that the perturbations are at the pixel level.

Similarly, the object detector is defined by the function $f : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^{B\times 6}$, where B represents the maximum number of objects that can be detected by the model. The information about each bounding box location, object's class, and confidence score (indexed by l_0) is encapsulated in 6-dimensional vectors. To prevent the attack from targeting objects that are incorrectly predicted, we establish a confidence threshold of 0.5. If n objects surpass this threshold from among B candidates, then only these n objects are classified as detected. Furthermore, $f^o_{l_0}(x)$ indicates the confidence score that the model identifies the o-th object as belonging to the predicted class from a sample x. In this context, the number $n(x)$ of detected objects from a sample x is $\sum_o \mathbb{1}\{\ f^o_{l_0}(x) > 0.5\}$, where $\mathbb{1}$ signifies the indicator function that takes the value one if the inequality holds and the value zero otherwise.

The goal of the object detector attack is to reduce the number of detected objects by modifying the minimum number of pixels, which is formally expressed as:

$$\begin{aligned} \min_{\delta} \quad & \max\left(n(x) - n(\overline{x})\right) \\ \text{s.t.} \quad & \|\delta\|_0 \leq \epsilon, \quad \epsilon \in \mathbb{Z}^+. \end{aligned} \tag{2}$$

where ϵ denotes the attack level and $\overline{x}$ denotes the perturbed image. Therefore, our objective is to eliminate bounding boxes from the detection model by iteratively accumulating pixel attacks. We address this problem by combining one-step REINFORCE[23] with our approach.

2.2 Remember Process

Agent. We construct an environment where an attacking agent interacts to generate adversarial images. The agent's policy utilizes a CNN-based architecture, where given a sample $x \in \mathbb{R}^{C\times H\times W}$, the agent observes the image and takes actions to determine the location (X, Y coordinates) and brightness (R, G, B) to modify, then generates perturbed images. We define two types of actions for brightness: "Write" and "Erase." The "Write" action overwrites the pixel with the maximum brightness, while the "Erase" action sets the brightness to zero. This configuration is chosen because, based on our experience, the attack success rate is higher when applying maximum changes to the pixels. Figure 1 shows the adversarial images generated by the agent. The agent generates the actions through random sampling of normal distributions, where the means and standard deviations are trained by the neural networks. The set of actions A_t, where t represents the training epoch of RL, contains $\mathcal{N}$ subsets corresponding to the number of attack pixels in each Remember process. These subsets are composed of the X, Y coordinates, and brightness values for each channel. The set A_t is defined as $\{\underbrace{a_1^1, a_2^1, a_3^1, \cdots, a_{C+2}^1}, \cdots, \underbrace{a_1^{\mathcal{N}}, a_2^{\mathcal{N}}, a_3^{\mathcal{N}}, \cdots, a_{C+2}^{\mathcal{N}}}\}$, where a_1 and a_2 represent the X and Y coordinates, respectively, and a_3 to a_{C+2} represent the brightness values for each channel. For the "Write" action, the brightness values are set to the maximum value, whereas for the "Erase" action, they are set to 0. For each pixel, the perturbed image $\overline{x}$ is generated as follows:

$$\overline{x}_{i,j,k} = \begin{cases} a_{i+2}^d & \text{if } j = a_1^d \text{ and } k = a_2^d, \\ x_{i,j,k} & \text{otherwise} \end{cases} \tag{3}$$

where i, j, and k are indices for channel, height, and width, respectively. In other words, $\overline{x}_{i,j,k}$ represents the brightness at position (j, k). Adding a pixel to the image is repeated for d from 1 to $\mathcal{N}$. Hence, the image is perturbed by $\mathcal{N}$ pixels. The equation describes the generation of $\overline{x}$ by repeatedly altering the brightness of each channel at the position (a_1, a_2) in the given sample x. The agent trains by using the gradient of the reward and the log probability of the sampled actions.

Environment. The environment evaluates the image generated by the agent and assigns a reward. The reward r is defined as:

$$r = \sum_{o=1}^{n} f_{l_0}^o(x) - f_{l_0}^o(\overline{x}) + \Omega(\overline{x}), \tag{4}$$

where n represents the number of detected objects in the image x for object detection, while $\Omega(\overline{x}) = (n(x) - n(\overline{x}))$ signifies the number of removed objects after the adversarial attack. l_0 is the index for the confidence score of the detected object. Hence, the reward is defined as the sum of the differences in confidence scores for each object plus the number of objects removed.

For classification, l_0 is the index for the correct class, and n is set to 1. $\Omega(\overline{x})$ is set to 1 if the adversarial image generation is successful and 0 otherwise. In essence, the reward is calculated as the sum of the differences in the probability of the correct class between the perturbed and original image, along with an additional component indicating whether the model successfully created an adversarial example.

Memory. The role of memory is to save the best reward value and its corresponding perturbed images. The stored information is also used to determine when the Forget process should start. Without memory, RL models tend to identify universally shared vulnerabilities in the clean images provided to the victim model. In contrast, our objective is to generate adversarial attacks regardless of these common vulnerabilities. To minimize unnecessary queries that converge on such vulnerabilities, we have incorporated memory concepts into the RL approach.

In our approach, memory stores the maximum reward values r^* and their corresponding perturbed images $\overline{x}^*$ by selectively saving the higher reward as $r^* = \max(r^*, r_t)$, where r_t denotes the rewards

 https://doi.org/10.52202/079017-2766

Table 1: **The results of adversarial attacks on the ImageNet dataset.** Each score represents the mean success rate of the attack, mean L_0 norm and mean the number of queries. In terms of the success rate, a higher value signifies better performance, whereas for the L_0 norm and the number of queries, lower values are indicative of superior performance. The best method is highlighted in bold.

Model	Test accuracy	Attack	Succes rate ↑	L_0 ↓	Query ↓
VIT-B[24]	81.07 %	OnePixel[8]	9.3 %	15	1453
		ScratchThat[9]	40.9 %	420	9418
		Pixle[11]	51.4 %	286	728
		RFPAR(Ours)	**64.1 %**	211	613
ResNeXt50[25]	77.62 %	OnePixel[8]	8.1 %	15	5100
		ScratchThat[9]	38.1 %	95	1400
		Pixle[11]	89.1 %	538	663
		RFPAR(Ours)	**95.3 %**	138	442
RegNetX-32GF[26]	80.62 %	OnePixel[8]	12.3 %	15	1358
		ScratchThat[9]	60.6 %	427	8653
		Pixle[11]	73.7 %	276	705
		RFPAR(Ours)	**88.4 %**	164	484
DenseNet161[27]	77.14 %	OnePixel[8]	14.1 %	15	1248
		ScratchThat[9]	60.6 %	425	8367
		Pixle[11]	82.3 %	243	625
		RFPAR(Ours)	**91.7 %**	152	464
MNASNet[28]	73.46 %	OnePixel[8]	14.2 %	15	1128
		ScratchThat[9]	65.3 %	425	8828
		Pixle[11]	83.7 %	240	607
		RFPAR(Ours)	**95.0 %**	150	442
MobileNet-V3[29]	74.04 %	OnePixel[8]	8.1 %	15	1461
		ScratchThat[9]	51.8 %	420	9293
		Pixle[11]	69.6 %	306	769
		RFPAR(Ours)	**86.6 %**	213	596

given by the environments during the t-th training epoch. After training each epoch of data, the algorithm checks whether the reward values have bounded. We define the rewards as bounded if the following condition is satisfied:

$$\frac{r_t - r^*}{r^*} < \eta \tag{5}$$

where η signifies the bound threshold. This equation indicates that the rate of increase in the reward stored in memory is less than η. The convergence of rewards is defined as the rewards being bounded for a certain period, denoted as T. Both η and T are hyperparameters. If the reward converges, the Remember process ceases and the Forget process starts.

2.3 Forget Process

The goal of the Forget process is to reset the trained RL model and its memory, and to feed the image $\overline{x}^*$ as a new input for the reset RL model. Additionally, the maximum L_0 increase for the reset RL model, as it is determined by the number of reward convergences, attack pixels, and channels. This process is implemented to prevent the agent from overfitting, which can hinder effective exploration of new inputs. The impact of memory and initialization is discussed in Section 3.5.

3 Experiments

Section 3.1 details the dataset, evaluation metrics, victim models, and hyperparameters used in our experiments. In Section 3.2, we evaluate our proposed attack on image classification by comparing

it with previous attack methods. Section 3.3 compares the performance of our method on object detection, varying the attack dimension ($\alpha = 0.01$ to 0.05), and compares the results with other query-based attacks. In Section 3.4, we conduct experiments on the Argoverse dataset, which has larger image dimensions, and discuss the findings. Finally, Section 3.5 presents an ablation study on the memory and initialization components we introduced. Additional experimental results can be found in Appendix C and D.

3.1 Experimental Details

Datasets, Metrics and Hardware. For image classification, we use the validation dataset from ImageNet-1K[30]. To reduce computational costs, we extract one correctly classified image per category from the victim model, resulting in a total dataset of 1000 images for adversarial attack attempts. We evaluate our methods with respect to different victim models by calculating the success rate, L_0 norm, and the number of queries. The *success rate* represents the percentage of successful adversarial attacks out of the 1000 images, with higher values indicating better performance. The L_0 norm refers to the number of non-zero elements in perturbation δ, with lower values indicating better performance. The number of *queries* indicates how often the victim model is queried to generate an adversarial example, with fewer queries indicating better performance. The *ATA* (ATtacked Area) refers to the proportion of pixels in the image that were attacked, a lower value indicates fewer changes. For object detection, we use the 2017 validation set from the MS-COCO dataset[31] and Argoverse-1.1 validation set[32]. To facilitate comparison with PRFA [16] and GARSDC [17], we use mAP to evaluate the attacks. The mAP is calculated as the average over thresholds ranging from IOU = 0.5 to 0.95. Additionally, *RM* indicates the average percentage of objects removed from the clean image, while *RD* refers to the decrease in mAP. Both a lower mAP and a higher RM indicate greater success. Lastly, we used an AMD Ryzen 9 5900X, RTX 3090TI, and 64.0GB of RAM, running on Windows 11 with CUDA version 12.1.

Victim Models. For image classification, we select six pre-trained models on the PyTorch platform as victim models: VIT[24], ResNeXt50[25], RegNetX-32GF[26], DenseNet161[27], MNASNet[28], and MobileNet-V3[29]. We compare the performance of our attack with OnePixel[8], ScratchThat[9], and Pixle[11]. For object detection, we use the pre-trained YOLOv8n model from the YOLOv8[22] platform and the pre-trained DDQ DETR-4scale model[33] from the MMDetection platform.

Hyperparameter. Our attack method utilizes four hyperparameters: the maximum number of iterations, the pixel attack rate α, the bound threshold η, and the duration T for maintaining the convergence condition. In the Remember process, α is a hyperparameter that determines the number of pixels to attack, proportional to the image size. The number of pixels $\mathcal{N}$ to be attacked is defined as $(H + W)/2 \times \alpha$. By default, we set the maximum number of iterations to 100 and η to 0.05. For image classification, we use $T = 3$ and $\alpha = 0.01$. For object detection, we experiment with $T = 20$ and α values ranging from 0.01 to 0.05.

3.2 Evaluation of Classification Attacks

Table 1 presents a performance comparison of various adversarial attack methods on different victim models for image classification. RFPAR consistently achieves the highest success rate, significantly outperforming the other three attack methods. For instance, for the VIT model, RFPAR achieves a success rate of 64.1%, compared to OnePixel's 9.3%, ScratchThat's 40.9%, and Pixle's 51.4%. The trend is similar for other models, with RFPAR showing substantial improvements in success rate. Regarding the L_0 norm, which measures the sparsity of the perturbations, RFPAR generally achieves a lower L_0 norm than ScratchThat and Pixle but higher than OnePixel. For example, in the case of ResNeXt50, RFPAR has an L_0 norm of 138, compared to OnePixel's 15, ScratchThat's 95, and Pixle's 538. While OnePixel has the lowest L_0 norm, its success rate is significantly lower than RFPAR's, indicating a trade-off between perturbation sparsity and attack effectiveness. In terms of the number of queries, RFPAR requires fewer queries than the other methods, except for OnePixel in some cases. This demonstrates that RFPAR is more efficient in terms of query cost, which is crucial for practical adversarial attacks. Overall, RFPAR exhibits superior performance across all victim models in terms of success rate while maintaining competitive L_0 norms and requiring fewer queries compared to other methods, making it an effective and efficient approach.

 https://doi.org/10.52202/079017-2766

Table 2: **Attack Results on Object Detection Models.** The subscripts after RFPAR denote a pixel attack rate, α. RM indicates the average percentage of objects removed from the clean image. L_0 represents the average $\|\delta\|_0$. Query denotes the average number of queries made to the victim model. Higher RM, lower mAP, lower L_0, and lower Query values indicate better performance.

Attacks	YOLOv8[22]				DDQ[33]			
	RM ↑	mAP ↓	L_0 ↓	Query ↓	RM ↑	mAP ↓	L_0 ↓	Query ↓
clean	-	0.398	-	-	-	0.376	-	-
$\text{RFPAR}_{0.01}$	0.65	0.218	521	1403	0.60	0.125	391	1450
$\text{RFPAR}_{0.02}$	0.70	0.187	955	1427	0.73	0.103	787	1690
$\text{RFPAR}_{0.03}$	0.75	0.151	1459	1374	0.76	0.075	1074	1512
$\text{RFPAR}_{0.04}$	0.76	0.150	1814	1348	0.80	0.061	1429	1457
$\text{RFPAR}_{0.05}$	0.91	0.111	2043	1254	0.83	0.054	1780	1528

3.3 Evaluation of Object Detection Attacks

Attacking object detection models is more challenging than attacking image classification models because there are more objects to consider in the object detection task. More pixels need to be modified, adjusted by α from 0.01 to 0.05, to deceive the victim models. Table 2 compares the performance of different α values of the RFPAR method on two object detection models, YOLOv8 and DDQ. The RM rate for YOLOv8 increases from 0.65 ($\text{RFPAR}_{0.01}$) to 0.91 ($\text{RFPAR}_{0.05}$) and for DDQ from 0.60 to 0.83, indicating that stronger attacks remove more detected objects. The mAP also decreases from 0.218 to 0.111 for YOLOv8 and from 0.125 to 0.054 for DDQ. At $\alpha = 0.05$, our attack successfully reduced the mAP by an average of 0.301 and achieved a RM of 0.87. The number of queries remains relatively stable, ranging from 1254 to 1427 for YOLOv8 and from 1450 to 1690 for DDQ, suggesting a consistent query cost despite increasing perturbation intensity. Overall, the results indicate that the RFPAR method is highly effective in generating adversarial attacks on object detection models, balancing perturbation sparsity, and attack effectiveness while maintaining query efficiency.

Table 3: **Comparison to other methods.** RD means reduction in mAP.

Attacks	YOLO	
	RD ↑	Query ↓
PRFA[16]	0.21	2949
GARSDC[17]	**0.29**	2691
RFPAR	**0.29**	**1270**

To demonstrate the effectiveness of our method, we compared it with other query-based black-box attacks. Table 3 shows the performance of three different attack methods - PRFA, GARSDC, and RFPAR - on the YOLO object detection model. In this table, RD refers to the decreased mAP value, and Query indicates the average number of queries. The RFPAR method shows strong performance by achieving the highest RD (tied with GARSDC) and requiring the fewest queries. This indicates that RFPAR is not only effective in reducing the YOLO model's performance but also efficient in terms of the number of queries needed to achieve this reduction. GARSDC also demonstrates high effectiveness with the same reduction as RFPAR but requires more than twice the number of queries. Overall, RFPAR stands out as the most balanced and efficient attack method in this comparison.

3.4 Experiments on a Larger Scale Data

To verify the effectiveness of our proposed method on larger dimensions 1920×1200, we randomly selected one video sample from the Argoverse dataset and conducted experiments using YOLOv8. The experimental results are presented in Table 4. The RM achieved 0.94, indicating a successful reduction in the number of detected objects. Argoverse achieved a RM of 0.94, similar to the RM observed for MS-COCO. The ATA for these datasets was 0.1% and 0.02%, respectively, indicating that only a very small portion of the image area was attacked. However, the

Table 4: **Comparison on dataset.** ATA means the ratio of altered pixels to the image size.

Datasets	YOLO			
	RM ↑	RD ↑	ATA ↓	Query ↓
MS-COCO	0.91	0.29	0.02 %	1270
Argoverse	0.94	0.05	0.10 %	1906

mAP did not decrease as significantly as in previous experiments. This discrepancy can be explained by considering that RFPAR primarily reduces the number of objects detected. If a particular class has many objects, reducing their number may not significantly impact the overall mAP due to the presence of other classes. In summary, while RFPAR successfully removes objects in the larger Argoverse dataset, its effectiveness in reducing mAP is limited in datasets with a high density of objects in specific classes.

3.5 Ablation study

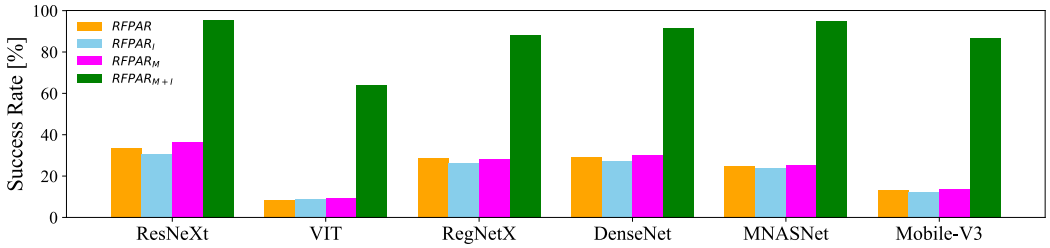

Figure 3: **Ablation study.** The x and y axes show different victim models and the attack success rate, respectively. The notation $_I$ signifies the inclusion of the initialization step in the Forget process, and $_M$ denotes that the Remember process incorporates memory.

In this section, we analyze the impact of Initialization (I) and Memory (M) on our model's performance. If Initialization is ablated in the Forget process, the Agent is not reinitialized and retains information from the previous Remember process. On the other hand, if Memory is ablated, the Agent's reward, instead of the reward stored in Memory, serves as a bound condition. We conduct ablation experiments under similar query conditions and present the results in Appendix G. As shown in Figure 3, RFPAR denotes the baseline state without I and M, while RFPAR$_*$ indicates the inclusion of specific processes. Comparing RFPAR and RFPAR$_M$, it is evident that the introduction of memory significantly enhances the attack success rate. This result suggests that the RL method benefits from storing the highest-reward images of restricted pixels during each Forget process. When comparing RFPAR$_I$ and RFPAR$_{M+I}$, we observe that initialization prevents RL model from overfitting to specific patterns and escaping local optima, thereby improving performance. Conversely, the comparison between RFPAR and RFPAR$_I$ indicates that Initialization alone, without memory, has a negligible impact. This finding implies that RL model without memory fails to generate meaningful adversarial attacks. In summary, memory supports RL model in generating effective adversarial attacks, while Initialization prevents overfitting and enhances overall performance.

4 Conclusion

In this paper, we propose the Remember and Forget Pixel Attack using Reinforcement Learning (RFPAR) for attacking neural network models with limited pixels. Traditional pixel-based attacks have been confined to image classification, but our method extends this approach to include object detection as well. For image classification, we compared the performance of RFPAR against OnePixel, ScratchThat, and PIXLE across six victim models using the ImageNet-1K dataset, and RFPAR demonstrated superior performance. In object detection, we evaluated RFPAR on the MS-COCO dataset using YOLOv8 and DDQ models, comparing it with PRFA and GARSDC attacks. RFPAR achieved performance comparable to the state-of-the-art query-based attack GARSDC, while reducing the number of queries by 52.8%, proving its efficiency. Additionally, we showed that RFPAR is capable of performing pixel attacks on larger datasets, specifically the Argoverse dataset with dimensions, surpassing the sizes of the ImageNet and MS-COCO datasets. Our findings may enable malicious individuals to compromise real-world AI systems. Consequently, research on defenses against adversarial attacks is becoming increasingly important.

Broader Impacts. Defects in camera sensors, such as hot pixels or dead pixels, can impact image quality and degrade the performance of neural network models. Our approach mimics these camera defects. In this paper, RFPAR simulates real-world issues by replacing specific pixels with

https://doi.org/10.52202/079017-2766

values of either zero or one, inducing incorrect predictions by the neural network. Since these types of perturbations can occur in practice, it is crucial for neural networks to be robust against them. However, research on pixel-based L_0 attacks is limited compared to other types of attacks. Our approach helps analyze model vulnerabilities with respect to both adversarial attacks and real-world scenarios, contributing to the development of more robust neural networks that can withstand such defects. Additionally, the phenomenon where the prediction changes with only a small number of pixel modifications that do not alter the overall meaning can be considered an anomaly in artificial neural networks. This type of attack provides important insights into understanding the limitations of neural networks.

Limitations. In this work, the pixel values are either zero or one. While the meaning remains unchanged, this can still make it noticeable in certain cases. Additionally, the Forget process is quite simple. The time complexity of RFPAR is worse than that of other pixel attacks. However, at the ImageNet scale, RFPAR outperforms others in speed. This result is presented in Appendix H. In future work, we will apply meta-learning to the Forget process and aim to reduce not only L_0, but also L_∞.

Negative Impacts. In applications like defective product detection [34] and disease prediction systems [35], adversarial attacks could degrade product quality or lead to incorrect diagnoses, which may have serious, or even fatal, consequences. Our proposed approach increases the effectiveness of query-based black-box attacks, making them more applicable to real-world scenarios. As a result, vision AI systems may face significant threats to their functionality and reliability. Therefore, it is crucial for these systems to proactively identify potential vulnerabilities and implement robust defenses.

Mitigation of Risks. Our method requires an average of over 1000 queries to successfully deceive an object detection model. Similarly, as shown in Table 7 in the Appendix E, transformer-based models also require an average of over 1000 queries to achieve a high success rate. If we limit the number of queries to around 1000 in a short period of time, our method can easily defend the model. For CNN-based models, since fewer queries are needed, limiting the queries to 400 can effectively defend the model. Additionally, according to the attack results on adversarially trained models shown in Table 8 in the Appendix F, adversarial training effectively reduces the attack success rate and increases the number of queries needed. Therefore, by adversarially training the models and appropriately limiting the queries, this attack can be defended against.

Acknowledgments and Disclosure of Funding

This research was supported by the Korea Institute for Advancement of Technology (KIAT) through the Ministry of Trade, Industry, and Energy (MOTIE) of the Korean government (Grant No. P0017124, HRD Program for Industrial Innovation), the Korea Agency for Infrastructure Technology Advancement (KAIA) under the Ministry of Land, Infrastructure, and Transport (Grant No. RS-2022-00156364), and the Ministry of Education and the National Research Foundation of Korea as part of the "Convergence and Open Sharing System (NCCOSS)" Project.

References

[1] Zaremba W. Sutskever I. Bruna J. Erhan D. Goodfellow I. J. Szegedy, C. and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations, ICLR*, 2014.

[2] Muku S. Tulasi A. Bhambri, . and B. Buduru, A. A study of black box adversarial attacks in computer vision. *CoRR*, abs/1912.01667, 2019. http://arxiv.org/abs/1912.01667.

[3] Shlens J. Goodfellow, I. J. and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations, ICLR*, 2015.

[4] Rice L. Wong, E. and J. Z. Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations, ICLR*, 2020.

[5] Makelov A. Schmidt L. Tsipras D. Madry, A. and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations, ICLR*, 2018.

[6] Dong Y. Pang T. Su H. Cheng, S. and J. Zhu. Improving black-box adversarial attacks with a transfer-based prior. In *Advances in Neural Information Processing Systems, NeurIPS*, 2019.

[7] Wang S. Shi, Y. and Y. Han. Curls and whey: Boosting black-box adversarial attacks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2019.

[8] Vargas D. V. Su, J. and K. Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

[9] Hitaj B. Ciocarlie G. F. Jere, M. and F. Koushanfar. Scratch that! an evolution-based adversarial attack against neural networks. *CoRR*, abs/1912.02316, 2019. http://arxiv.org/abs/1912.02316.

[10] Babu A. R. Mousavi S. Ghorbanpour S. Gundecha V. Gutierrez R. L. Guillen A. Sarkar, S. and A. Naug. Reinforcement learning based black-box adversarial attack for robustness improvement. In *IEEE International Conference on Automation Science and Engineering, CASE*, 2023.

[11] Dántoni D. Nicolosi A. Pomponi, J. and S. Scardapane. Rearranging pixels is a powerful black-box attack for RGB and infrared deep learning models. *IEEE Access*, 11:11298–11306, 2023.

[12] Wu W. Zhang J. Deng, Y. and Z. Zheng. Blurred-dilated method for adversarial attacks. In *Advances in Neural Information Processing Systems, NeurIPS*, 2023.

[13] Wu B. Fan Y. Liu L. Li Z. Feng, Y. and S.-T. Xia. Boosting black-box attack with partially transferred conditional adversarial distribution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2022.

[14] Chen X. Liu C. Liu, Y. and D. Song. Delving into transferable adversarial examples and black-box attacks. In *International Conference on Learning Representations, ICLR*, 2017.

[15] Kortylewski A. Xie C. Cao Y. Yang, C. and A. L. Yuille. Patchattack: A black-box texture-based attack with reinforcement learning. In *European Conference on Computer Vision, ECCV*, 2020.

[16] Wu B. Fan Y. Wei X. Liang, S. and X. Cao. Parallel rectangle flip attack: A query-based black-box attack against object detection. In *IEEE/CVF International Conference on Computer Vision, ICCV*, 2021.

[17] Li L. Fan Y. Jia X. Li J. Wu B. Liang, S. and X. Cao. A large-scale multiple-objective method for black-box attack against object detection. In *European Conference on Computer Vision, ECCV*, 2022.

[18] Zolfi A. Demetrio L. Biggio B. Shapira, A. and A. Shabtai. Phantom sponges: Exploiting non-maximum suppression to attack deep object detectors. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV*, 2023.

[19] Li C. Wen S. Han Q.-L. Nepal S. Zhang X. Wang, D. and Y. Xiang. Daedalus: Breaking nonmaximum suppression in object detection via adversarial examples. *IEEE Transactions on Cybernetics*, 52(8):7427–7440, 2022.

[20] He F. Huang X. Chen, S. and K. Zhang. Relevance attack on detectors. *Pattern Recognition*, 124:108491, 2022.

[21] Zhang J. Li, D. and K. Huang. Universal adversarial perturbations against object detection. *Pattern Recognition*, 110:107584, 2021.

[22] Córdova Esparza D. M. Terven, J. R. and J.-A. Romero-González. A comprehensive review of YOLO architectures in computer vision: From yolov1 to yolov8 and YOLO-NAS. *Machine Learning and Knowledge Extraction*, 5(4):1680–1716, 2023.

 https://doi.org/10.52202/079017-2766

[23] McAllester D. A. Singh S. Sutton, R. S. and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems, NeurIPS*, 1999.

[24] Beyer L. Kolesnikov A. Weissenborn D.-Zhai X. Unterthiner T. Dehghani M. Minderer M. Heigold G. Gelly S. Uszkoreit J. Dosovitskiy, A. and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations, ICLR*, 2021.

[25] Girshick R. B. Dollár P. Tu-Z. Xie, S. and K. He. Aggregated residual transformations for deep neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2017.

[26] Kosaraju R. P. Girshick R. B.-He K. Radosavovic, I. and P. Dollár. Designing network design spaces. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2020.

[27] Liu Z. van der Maaten L. Huang, G. and K. Q. Weinberger. Densely connected convolutional networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2017.

[28] Chen B. Pang R. Vasudevan V.-Sandler M. Howard A. Tan, M. and Q. V. Le. Mnasnet: Platform-aware neural architecture search for mobile. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2019.

[29] Pang R. Adam H. Le Q. V. Sandler M. Chen B. Wang W. Chen L.-C. Tan M. Chu G. Vasudevan V. Howard, A. and Y. Zhu. Searching for mobilenetv3. In *IEEE/CVF International Conference on Computer Vision, ICCV*, 2019.

[30] Dong W. Socher R. Li L.-J. Li K. Deng, J. and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2009.

[31] Maire M. Belongie S. J.-Hays J. Perona P. Ramanan D. Dollár P. Lin, T.-Y. and C. L. Zitnick. Microsoft COCO: common objects in context. In *European Conference on Computer Vision, ECCV*, 2014.

[32] Wang Y.-X. Li, M. and D. Ramanan. Towards streaming perception. In *European Conference on Computer Vision, ECCV*, 2020.

[33] Wang X.-Wang J. Pang J.-Lyu C. Zhang W. Luo P. Zhang, S. and K. Chen. Dense distinct query for end-to-end object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2023.

[34] Ciuti G.-Milazzo M. Chiurazzi M.-Roccella S. Oddo C. M. Czimmermann, T. and P. Dario. Visual-based defect detection and classification approaches for industrial applications - A SURVEY. *Sensors*, 20(5):1459, 2020.

[35] Linda Wang, Zhong Qiu Lin, and Alexander Wong. Covid-net: a tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images. *Scientific Reports*, 10:19549, 2020.

[36] Yu Y.-Jiao J. Xing E. P. El Ghaoui L. Zhang, H. and M. I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of the International Conference on Machine Learning, ICML*, 2019.

[37] Hu H.-Lin Y. Yao Z. Xie Z. Wei Y. Ning J. Cao Y. Zhang-Z. Dong L. Wei F. Liu, Z. and B. Guo. Swin transformer V2: scaling up capacity and resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2022.

[38] Cord M.-Douze M. Massa F. Sablayrolles A. Touvron, H. and H. Jégou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the International Conference on Machine Learning, ICML*, 2021.

[39] Chi C.-Yao Y. Lei Z. Zhang, S. and S. Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2020.

[40] Su W.-Lu L. Li B. Wang X. Zhu, X. and J. Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *International Conference on Learning Representations, ICLR*, 2021.

[41] Wu D.-Wang Y. Guo Y. Mo, Y. and Y. Wang. When adversarial training meets vision transformers: Recipes from training to architecture. In *Advances in Neural Information Processing Systems, NeurIPS*, 2022.

[42] Basart S.-Mu N. Kadavath S. Wang F. Dorundo E. Desai R. Zhu T. Parajuli-S. Guo M. Song D. Steinhardt J. Hendrycks, D. and J. Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *IEEE/CVF International Conference on Computer Vision, ICCV*, 2021.

 https://doi.org/10.52202/079017-2766

A Related Work

Adversarial Attack. Adversarial attacks manipulate clean images with imperceptible modifications to fool Deep Neural Networks (DNNs) into making incorrect predictions. These attacks are broadly divided into black-box and white-box attacks. White-box attacks leverage detailed information about the target model, including training data and gradients, to craft adversarial images. In contrast, black-box attacks, which do not rely on any internal information about the victim model, are divided into transfer-based and query-based strategies. Transfer-based attacks create adversarial images using a surrogate model, aiming for these examples to also be effective against the target model. Query-based attacks iteratively modify clean images and query the victim model, using the resulting confidence scores to refine the attack. Typically, attack strategies are evaluated using the L_p norm to restrict the perturbation to remain imperceptible to humans[3, 4, 5, 36].

Black-box Pixel Attack in Image Classification. Unlike other metrics, the L_0 norm, also known as pixel norm, targets only a small subset of pixels in a clean image rather than attacking all of them. The pioneering pixel attack method, OnePixel[8], employed Differential Evolution (DE) to generate adversarial images. An advanced approach, ScratchThat[9], used DE to create curves and applied a parametric model to perturbations, reducing parameters and improving performance. A more recent study, PIXLE[11], enhanced query efficiency and attack success rate by using a simple algorithm instead of DE. Briefly, PIXLE generates adversarial images by selecting arbitrary patches in a clean image and applying the brightness of these pixels to others. Although this method improved performance, it ignored pixel independence due to its reliance on patches and exhibited inefficiencies stemming from randomness in brightness mapping. The previous study, PatchAttack[15], utilized RL to embed textures in specific regions of the clean image, discovering vulnerable patches and reducing randomness, which significantly decreased the number of queries and improved attack success rates. Unfortunately, this method still depended on patches, requiring at least 3% of the image area to be attacked. Our research focuses on eliminating patch dependency by attacking individual pixels and reducing randomness through RL. Extensive experiments demonstrate that our proposed attack outperforms the state-of-the-art methods in both query efficiency and attack success rate.

Query-based Adversarial Attack in Object Detection. Adversarial attacks in object detection are more challenging than those in image classification. The first query-based attack in object detection, PRFA[16], generates adversarial images using a parallel rectangle flipping strategy. Recent research, GARSDC[17], employs a genetic algorithm to create adversarial images, improving optimization efficiency by using adversarial examples generated from transfer-based attacks as the initial population. We extend our proposed attack from image classification to object detection. Experiments show that our method achieves a comparable mAP reduction on YOLO[22] to state-of-the-art methods while significantly reducing the number of queries, demonstrating its effectiveness in object detection.

B Theoretical Insight.

We initially used a multi-step REINFORCE approach but identified issues, leading us to propose the Forget and Remember processes using one-step REINFORCE. Generating adversarial examples with multi-step REINFORCE involves the objective function $U = E\left[\sum_0^\tau \gamma^{\tau-t} R[s_t, a_t|\pi_\theta]\right]$, where γ is the discount factor, s_t is the image at step t, a_t is the action at s_t, and the reward is $R[s_t, a_t|\pi_\theta] = f_{\theta,y}(s_0) - f_{\theta,y}(s_{t+1})$, where $f_{\theta,y}$ is the confidence score of the true label y. Here, a_t is a single pixel perturbation. We find that significant oscillations can be observed in the objective function. Let τ^* be the minimum number of steps to create an adversarial example. The sequence of pixels does not matter, leading to variations in the value of the objective function due to different orderings of a_t. Thus, for $i_t \in \{0, 1, 2, \cdots, \tau^*\}$ and $i_j \neq i_k$, the optimal objective function value is $U^* = E\left[\sum_0^\tau \gamma^{\tau-t} R[s_t, a_{i_t}|\pi_\theta]\right]$, with $\tau^*!$ permutations. This complicates training and increases the queries and L_0. To address this, we proposed the Forget and Remember process using one-step REINFORCE. Pixel perturbations at τ^* are defined as $A_{\tau^*} = \sum_0^{\tau^*} a_t$. By the intermediate value theorem, there exists a C in $[x, x + A_{\tau^*}]$ such that $f_{\theta,y}(x) > f_{\theta,y}(C) > f_{\theta,y}(x + A_{\tau^*})$. We propose a Forget and Remember process using one-step REINFORCE to iteratively find this C, assuming $C \in \{x + a_0, x + a_1, \cdots, x + a_{\tau^*}\}$. This one-step approach avoids the fluctuations of multi-step methods, offering better query efficiency and lower L_0.

C Experimental Results on Image Classification

In this section, we present experimental results that could not be included in the main text. The results of attacking the ResNeXt50 model on the ImageNet-1K dataset are shown in Fig 4. The parameters for the attack were set as follows: the maximum number of iterations was 100, α was 0.01, η was 0.05, and the duration T was 3.

Figure 4: **Adversarial examples generated by RFPAR on the ImageNet dataset.** The "Original Image" is the original unaltered image, the "Delta" represents the difference between the Original Image and the Adversarial Image, and the "Adversarial Image" is the image with the altered prediction. The predicted labels are shown below the Original Image and the Adversarial Image.

https://doi.org/10.52202/079017-2766

D Experimental Results on Object Detection

In this section, we present the experimental results for Object Detection from the main text. The results of attacking the YOLOv8n model on the MS-COCO dataset are shown, with the following parameters: the maximum number of iterations was set to 100, α ranged from 0.01 to 0.05, η was 0.05, and the duration T was 20. These results can be reproduced using the provided code.

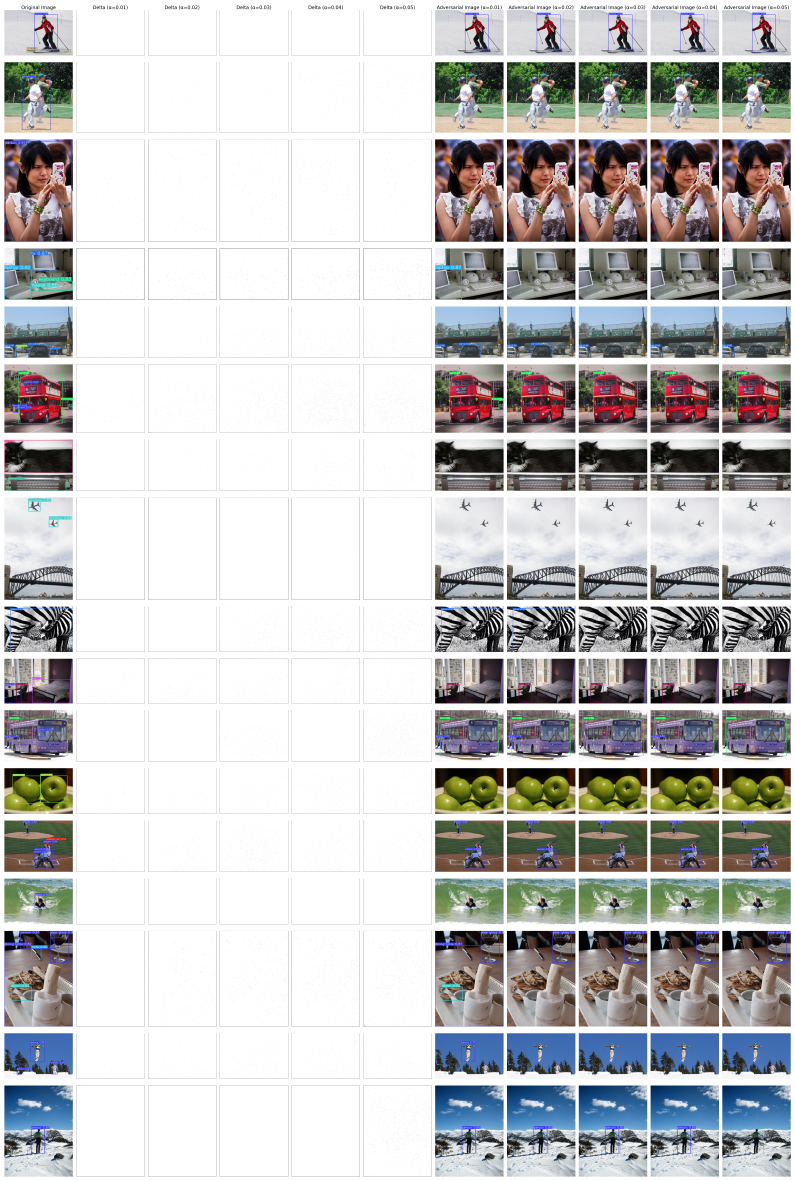

Figure 5: **Adversarial examples generated by RFPAR on the MS-COCO dataset.** The Original Image represents the unaltered image, and the Delta shows the difference between the Original Image and the Adversarial Image. The parameter is a hyperparameter that determines the attack level; a higher value of α attacks more pixels. We conducted experiments with ranging from 0.01 to 0.05. The Delta Image resulting from α values of 0.01 to 0.05 is presented in columns 2 to 6, and the Adversarial Image generated from the same α values is shown in columns 7 to 11. The Adversarial Image typically indicates an image with a changed prediction, but in this context, it also includes unsuccessful attacks. We present the results of Delta and Adversarial Images according to different values of α.

E Additional Experiments

In this section, we present additional experiments that were omitted due to page limitations. Table 5 shows the results on various transformer-based models (ViT-L [24], Swin-V2 [37], and Deit-B [38]). These results demonstrate that RFPAR is effective on transformer-based models. Table 6 provides the results for object detection models, ATSS [39] and Deformable DETR [40], showing that RFPAR is also effective for object detection. Finally, Table 7 presents results for transformer-based models (ViT-B, L, H, Swin-V2, and Deit-B) with iteration limits of 100 and 200. These results indicate that RFPAR requires more queries to achieve a comparable attack success rate on transformer-based models compared to CNN-based models.

Table 5: The results of transformer-based classifiers.

Attacks	ViT-L			Swin-V2			Deit		
	SR ↑	L_0 ↓	Query ↓	SR ↑	L_0 ↓	Query ↓	SR ↑	L_0 ↓	Query ↓
OnePixel	8.9%	15	1654	5.0%	15	1686	8.4%	15	1137
Pixle	66.4%	531	1396	66.8%	1052	1509	71.0%	551	1473
RFPAR	**78.0%**	355	1042	**69.4%**	608	1096	**84.3%**	412	1161

Table 6: The results of object detection models.

Attacks	Atss				Deformable DETR			
	RM ↑	mAP ↓	L_0 ↓	Query ↓	RM ↑	mAP ↓	L_0 ↓	Query ↓
clean	-	0.227	-	-	-	0.339	-	-
$\text{RFPAR}_{0.01}$	0.74	0.048	491	1530	0.61	0.170	333	1466
$\text{RFPAR}_{0.02}$	0.88	0.026	1025	1633	0.69	0.134	512	1502
$\text{RFPAR}_{0.03}$	0.90	0.026	1357	1504	0.72	0.135	869	1488
$\text{RFPAR}_{0.04}$	0.91	0.008	1666	1243	0.76	0.110	1200	1488
$\text{RFPAR}_{0.05}$	0.92	0.006	2074	1288	0.78	0.073	1274	1335

Table 7: The performance of RFPAR on transformer-based models with different iteration limits

Model	maximum of Iteration = 100			maximum of Iteration = 200		
	Succes rate ↑	L_0 ↓	Query ↓	Succes rate ↑	L_0 ↓	Query ↓
ViT-B	64.1%	211	613	**83.4%**	352	995
ViT-L	59.9%	209	618	**78.0%**	355	1042
ViT-H	62.2%	166	582	**73.5%**	229	917
Swin-V2	46.2%	352	611	**69.4%**	608	1096
Deit-B	60.2%	249	676	**84.3%**	412	1161

F Experiments on Adversarially Trained Models

In this section, we present experiments on adversarially trained models (Adv. ViT [41] and Adv. ResNeXt101 [42]). Table 8 shows that RFPAR is effective on these models, although its success rate is lower compared to generally trained models. Proportional calculations indicate that RFPAR reduced ViT's performance from 69.10% to 37.11%, which, according to Appendix D of the Adv. ViT paper [41], is more effective than CW20 (38.92%), PGD-20 (37.96%), and PGD-100 (37.52%), but slightly less effective than AutoAttack (34.62%). This demonstrates that our black-box attack, RFPAR, is nearly as effective as white-box attacks, despite having access to only limited information.

https://doi.org/10.52202/079017-2766

Table 8: The results of adversarial trained models.

Attacks	Adv. ViT			Adv. ResNeXt101		
	Succes rate ↑	L_0 ↓	Query ↓	Succes rate ↑	L_0 ↓	Query ↓
OnePixel	2.9%	15	2083	4.4%	15	1102
Pixle	34.0%	780	1912	42.5%	302	769
RFPAR	**46.3**%	547	1452	**57.4**%	243	626

G Query in Ablation study

In this section, we present Query regarding the ablation study and conduct experiments under similar conditions to ensure a fair comparison of each process.

Table 9: Query for ablation study.

	ViT-B	ResNeXt	RegNetX	DenseNet	MNASNet	MobileNet-V3
RFPAR	614	529	623	534	461	548
RFPAR$_I$	662	404	444	404	364	348
RFPAR$_M$	712	889	820	723	726	659
RFPAR$_{M+I}$	613	442	484	464	442	596

H Time complexity

Given the input dimension size N and constants K_i: OnePixel has $O(K_1)$ complexity, ScratchThat has $O(N^2)$, Pixle has $O(K_2)$, RFPAR has $O(N)$, PRFAR has $O(K_3)$, and GARSDC has $O(N)$. For image classification tasks, RFPAR's linear increase in time complexity with image size is more favorable than ScratchThat's exponential increase but less so than OnePixel and Pixle. In object detection tasks, both RFPAR and GARSDC see linear increases in time complexity with larger images, making them less advantageous than PRFAR.

The higher time complexity compared to most other studies is a limitation of our research. However, RFPAR generates attacks using neural networks, similar to GARSDC, and benefits from the high performance of GPUs, allowing for faster computations despite the increased time complexity. We present the experimental times in Table 10.

To improve efficiency, we propose integrating our method with meta-learning. RFPAR involves the agent learning afresh on the image multiple times, which can mitigate overfitting but also results in unnecessary queries. Meta-learning could enable the agent to quickly adapt to new tasks, enhancing efficiency by learning more rapidly.

Table 10: The experimental times in Table 1 of the main paper

	ViT	RegNetX-32GF	MNASNet	DenseNet161	MobileNet V3
OnePixel	3h 2m	4h 36m	50m	2h 52m	50m
ScratchThat	5d 12h 39m	11d 11h 27m	3d 19h 1m	7d 6h 43m	6d 11h 8m
Pixle	4h 48m	8h 16m	3h 3m	13h 33m	5h 14m
RFPAR	**1h 20m**	**1h 20m**	**20m**	**47m**	**29m**

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and precede the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (12 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: we have summarized our contributions and specified the category to which our method belongs in Section 1.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

https://doi.org/10.52202/079017-2766

Justification: In Section 4, we also describe the limitations of the proposed method.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: We present only experimental results without providing any theoretical results.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have detailed our proposed method in Sections 2.2 and 2.3 to ensure reproducibility.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We have included the code and some of the data used in the experiments in the supplemental material to ensure reproducibility.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so No is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

 https://doi.org/10.52202/079017-2766

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: In Section 3.1, we have specified the key hyperparameters used in the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Error bars are not provided due to the high computational cost.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Although we did not report the time of execution, the environment we used is specified in Section 3.1.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, we conducted in the paper conforms, in every respect. Specifically, we briefly discussed the potential negative impacts on society in Section 4.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We briefly discuss the negative societal impacts in Section 4.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

https://doi.org/10.52202/079017-2766

Answer: [NA]

Justification: It is not yet applicable to real-world scenarios as it is still challenging to cause significant disruption.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: This is explained in Section 3.1.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: This is shown in Fig 2 and Section 2.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: We did not conduct any research involving human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: We did not conduct any research involving human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-2766