algorithms

MDPI

Article

An Evolutionary, Gradient-Free, Query-Efficient, Black-Box Algorithm for Generating Adversarial Instances in Deep Convolutional Neural Networks

Raz Lapid, Zvika Haramaty and Moshe Sipper *

Department of Computer Science, Ben-Gurion University, Beer-Sheva 8410501, Israel
* Correspondence: sipper@bgu.ac.il

Abstract: Deep neural networks (DNNs) are sensitive to adversarial data in a variety of scenarios, including the black-box scenario, where the attacker is only allowed to query the trained model and receive an output. Existing black-box methods for creating adversarial instances are costly, often using gradient estimation or training a replacement network. This paper introduces **Qu**ery-Efficient **E**volutiona**ry** **Attack**—**QuEry Attack**—an untargeted, score-based, black-box attack. QuEry Attack is based on a novel objective function that can be used in gradient-free optimization problems. The attack only requires access to the output logits of the classifier and is thus not affected by gradient masking. No additional information is needed, rendering our method more suitable to real-life situations. We test its performance with three different, commonly used, pretrained image-classifications models—Inception-v3, ResNet-50, and VGG-16-BN—against three benchmark datasets: MNIST, CIFAR10 and ImageNet. Furthermore, we evaluate QuEry Attack's performance on non-differential transformation defenses and robust models. Our results demonstrate the superior performance of QuEry Attack, both in terms of accuracy score and query efficiency.

Keywords: deep learning; computer vision; adversarial attack; evolutionary algorithm

Citation: Lapid, R.; Haramaty, Z.; Sipper, M. An Evolutionary, Gradient-Free, Query-Efficient, Black-Box Algorithm for Generating Adversarial Instances in Deep Convolutional Neural Networks. *Algorithms* **2022**, *15*, 407. https://doi.org/10.3390/a15110407

Academic Editors: Luca Mariot and Luca Manzoni

Received: 13 September 2022
Accepted: 31 October 2022
Published: 31 October 2022

1. Introduction

Deep neural networks (DNNs) have become the central approach in modern-day artificial intelligence (AI) research. They have attained superb performance in multifarious complex tasks and are behind fundamental breakthroughs in a variety of machine-learning tasks that were previously thought to be too difficult. Image classification, object detection, machine translation, and sentiment analysis are just a few examples of domains revolutionized by DNNs.

Despite their success, recent studies have shown that DNNs are vulnerable to adversarial attacks. A barely detectable change in an image, for example, can cause a misclassification in a well-trained DNN. Targeted adversarial examples can even evoke a misclassification of a specific class (e.g., misclassify a car as a cat). Researchers have demonstrated that adversarial attacks are successful in the real world and may be produced for data modalities beyond imaging, e.g., natural language and voice recognition [1–4]. DNNs' vulnerability to adversarial attacks has raised concerns about applying these techniques to safety-critical applications.

To discover effective adversarial instances, most past work on adversarial attacks has employed gradient-based optimization [5–9]. Gradient computation can only be executed if the attacker is fully aware of the model architecture and weights. Thus, these approaches are only useful in a white-box scenario, where an attacker has complete access and control over a targeted DNN. Attacking real-world AI systems, however, might be far more arduous. The attacker must consider the difficulty of implementing adversarial instances in a black-box setting, in which no information about the network design, parameters, or training data is provided. In this situation, the attacker is exposed only to the classifier's input-output

pairs. In this context, a typical strategy has been to attack trained replacement networks and hope that the generated examples transfer to the target model [10]. The substantial mismatch of the model between the alternative model and the target model, as well as the significant computational cost of alternative network training, often renders this technique ineffective.

In our work we assume a real-world, black-box attack scenario, wherein a DNN's input and output may be accessed but not its internal configuration. We focus on a scenario in which a specific DNN is an image classifier, specifically, a convolutional neural network (CNN), which accepts an image as input and outputs a probability score for each class.

Herein, we present an evolutionary, gradient-free optimization approach for generating adversarial instances, more suitable for real-life scenarios, because usually there is no access to a model's internals, including the gradients; thus, it is important to craft attacks that do not use gradients. Our proposed attack can deal with either constrained (ϵ value that constrains the norm of the allowed perturbation) or unconstrained (no constraint on the norm of the perturbation) problems, and focuses on constrained, untargeted attacks. We believe that our framework can be easily adapted to the targeted setting.

In the next section we review the literature on adversarial attacks. Section 3 summarizes the threat model we assume for our proposed evolutionary attack algorithm. The algorithm itself—*QuEry Attack* (for **Qu**ery-Efficient **E**volutiona**ry Attack**)—is delineated in Section 4. The experiments conducted to test the method, along with results, are described in Section 5. We discuss our findings and present concluding remarks in Section 6.

Figure 1 shows examples of successful and unsuccessful instances of images generated by QuEry Attack, evaluated against ImageNet, CIFAR10, and MNIST.

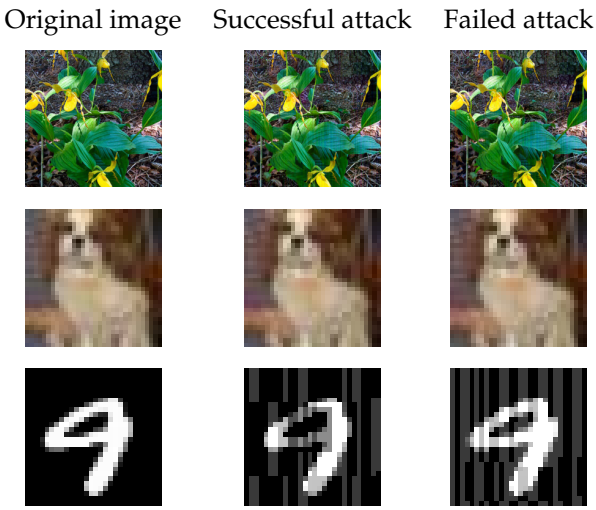

Figure 1. Examples of adversarial attacks generated by QuEry Attack. With higher resolution the attack becomes less visible to the naked eye. The differences between images that successfully attack the model and those that do not are subtle. (**Top row**): Imagenet ($l_\infty = 6/255$). (**Middle row**): CIFAR10 ($l_\infty = 6/255$). (**Bottom row**): MNIST ($l_\infty = 60/255$). (**Left**): the original image. (**Middle**): a successful attack. (**Right**): A failed attack.

2. Related Work

Adversarial attacks against DNNs have become an important research field in the last few years. For a comprehensive survey, we refer the reader to [11].

An important distinction is between 'white box' and 'black box' attacks. In white-box attacks, the attacker has knowledge of the attacked model's internal structure and parameters, and exploits that knowledge. It is commonly performed by using the model's gradients [5,7,12].

Although white-box methods achieved good results they usually do not represent a real-world scenario. More realistic is a black-box attack, where the attacker has no

knowledge of the model's structure and parameters. The attacker can only query the model—and act upon its outputs. We can further distinguish between the more common 'light black box' attacks, where the model gives the prediction's confidence (which can be exploited), and 'dark black box' attacks where the model only gives the final class prediction (e.g., [10]).

The first effective black-box attack traded a priori information of the model with extensive runtime querying [10]. Using a large number of queries it builds a substitute model, and attacks it with traditional white-box methods. It uses the transferability property, namely, an attack that succeeds on one model will likely succeed on a similar—though not identical—model.

Other works used the more permissive 'light black-box' scenario, which can use the prediction's confidence value. Some works estimate the gradient with this information and then use traditional gradient-based attacks [13].

All these black-box methods rely on gradients, and, thus, are sensitive to many defense methods that obscure gradients [14–16]. This has given rise to methods that do not rely on gradients at all, e.g., [17], which uses random search and is also query-efficient.

Instead of randomness, one can use evolutionary methods. In *Evolutionary Algorithms* (EAs), core concepts from evolutionary biology—inheritance, random variation, and selection—are harnessed in algorithms that are applied to complex computational problems. EA techniques have been shown to solve numerous difficult problems from widely diverse domains, and also to produce human-competitive machine intelligence [18]. EAs also present many important benefits over popular Machine Learning methods, including [19]: less reliance on the existence of a known or discoverable gradient within the search space; ability to handle design problems, where the objective is to design new entities from scratch; fewer required a priori assumptions about the problem at hand; seamless integration of human expert knowledge; ability to solve problems where human expertise is very limited; support of interpretable solution representations; and support of multiple objectives.

The evolutionary method GenAttack is a *targeted* attack (thus not directly comparable to ours, which is *untargeted*) that used a fitness function that tries to increase the probability of the target class and decrease the probability of the other classes [20]. Its fitness function ignored the distance between the images. Interestingly, GenAttack uses fitness-proportionate selection, which is employed less often nowadays due to known problems. It uses an adaptive mutation rate to balance between exploration in early phases and exploitation in later phases.

Ref. [21] treated the adversarial problem as one of multi-objective optimization: minimize the class prediction's score on one hand, and the distance between the original image and a perturbed one on the other hand.

Another attack method changes a single pixel [22]. This method uses differential evolution (DE) [23] without crossover. However, it sometimes required thousands of queries.

Ref. [24] also uses DE, but unlike the other evolutionary computation (EC) methods reviewed, it uses EC to approximate the gradients.

Unlike the above methods, which tried to minimize the perturbation as much as possible and make it as unnoticeable to the human eye as possible, [25] makes a small but noticeable change, which looks like a regular scratch (a similar approach in the domain of Natural Language Processing creates sentences that do not make sense to a human reader [26]). The approach uses DE as well, and also Covariance-Matrix Adaptation Evolution Strategies (CMA-ES) [27]. Unlike most attacks, which use the l_∞ or l_2 norms, this one is based on the l_0 norm.

In the EC methods seen so far, evolution is run against a *single* image, and each individual is a perturbation added to that image. Ref. [28], on the other hand, used the transferability property mentioned earlier to evolve a universal perturbation. An individual is an image mask that can be applied as a perturbation to any image.

Our extensive scrutiny of the literature and software repositories revealed that many authors compare their work to prior works that do not use the same threat model: there might be a mismatch in norms (e.g., l_2 vs. l_∞), white box vs. black box, or other subtle differences. Moreover, having investigated numerous software repositories, we found that running the code of many papers is far from straightforward.

3. Threat Model

In the black-box attack setting, queries to the network are permitted but access to internal states is prohibited (e.g., executing backpropagation). Hence, our threat model, or scenario, is as follows:

- The attacker is unaware of the network's design, settings, or training data.
- The attacker has no access to intermediate values in the target model.
- The attacker can only use a black-box function to query the target model.

Note that the above threat model determines the comparisons we perform, which focus on attacks that are:

1. Black-box,
2. Untargeted,
3. l_∞-norm bounded.

We can consider a network model to be a function:

$$f : [0,1]^d \rightarrow \mathbb{R}^C, \tag{1}$$

where d is the number of input features and C is the number of classes. The c-th value $f_c(x) \in \mathbb{R}$ specifies the predicted score of classifying input image x as class c. The classifier assigns class $y = \text{argmax}_{c=1,\ldots,C} f_c(x)$ to the input x.

A *targeted* attack aims to create an image that will be incorrectly classified into a given (incorrect) class. An *untargeted* attack aims to create an image that will be incorrectly classified into *any* class except the correct one. An image $\hat{x}$ is termed an adversarial example, with an l_p-norm bound of ϵ for x, if:

$$\begin{aligned} &\underset{c=1,\ldots,C}{\text{argmax}}\, f_c(\hat{x}) \neq y, \\ &s.t.\ \|\hat{x} - x\|_p \leq \epsilon \text{ and } \hat{x} \in [0,1]^d. \end{aligned} \tag{2}$$

To wit, the model should classify $\hat{x}$ incorrectly, while preserving similarity between x and $\hat{x}$ under an l_p distance metric.

We focus on a black-box, score-based attack, wherein the only information of the threat model is the raw output (logits).

Our suggested black-box approach may theoretically be used in conjunction with classic machine-learning models, with the same input–output relationship. Because DNNs have reached state-of-the-art performance in a variety of image tasks, we focus on them in this paper.

4. QuEry Attack

QuEry Attack is an evolutionary algorithm (EA) that explores a space of images, defined by a given input image and a given input model, in search of adversarial instances. It ultimately generates an attacking image for the given input image. Unlike white-box approaches, we make no assumptions about the targeted model, its architecture, dataset, or training procedure. We assume that we have an image x, which a black-box neural network, f, classifies by outputting a probability distribution over the set of classes, as stated in Section 3. The actual label y is computed as $y = \text{argmax}\, f(x)$.

Our objective is to find a perturbed image, $\hat{x}$, of image x, such that, $\|\hat{x} - x\|_\infty \leq \epsilon$, which causes the network to predict $y' = \text{argmax}\, f(\hat{x})$, such that $y' \neq y$. Finding $\hat{x}$ may be cast as a constrained optimization problem:

$$\min_{\hat{x} \in [0,1]^d} \mathcal{L}(f(\hat{x}), y), \;\; s.t. \;\; \|\hat{x} - x\|_\infty \leq \epsilon, \tag{3}$$

for a given loss function $\mathcal{L}$.

We use loss $\mathcal{L}$ as the fitness function, defined in our case as:

$$\mathit{fitness}(\hat{x}) = f_y(\hat{x}) - \max_{c \neq y} f_c(\hat{x}) + \lambda \|\hat{x} - x\|_2, \tag{4}$$

where $\hat{x}$ is a perturbed image, f_y is the predicted score that $\hat{x}$ belongs to class y, f_c is the predicted score that $\hat{x}$ belongs to class $c \neq y$. In order to guarantee that the adversarial perturbation is as imperceptible as possible we penalize the l_2 distortion of the perturbation by including a regularization component in the fitness function. We use l_2 regularization because we noticed that most of the evolved adversarial examples were on the edges of the ϵ-ball, and we wanted to give precedence to examples which were closer to the original input. This penalization is determined by the λ value, which is the regularization strength. In our experiments we used $\lambda = 1$.

The ultimate goal is to minimize the fitness value: Essentially, the lower the logit of the correct class, and the higher the maximum logit of the other classes, the better the value.

Algorithm 1 provides the pseudo-code of QuEry Attack. The original image x, along with a number of hyperparameters, are given as input to the algorithm. QuEry Attack generates an adversarial image $\hat{x}$, with the model classifying $\hat{x}$ as y', such that $y' \neq y$ and $\|\hat{x} - x\|_\infty \leq \epsilon$.

The main goal of QuEry Attack is to produce a successful attack, using as few queries to the model as possible. The maximal number of queries equals generation count $(G) \times$ population size (N).

4.1. Initialization

Initialization is crucial for optimization problems, e.g., in deep-learning training, gradient descent reaches a local minimum that is significantly determined by the initialization technique [29,30]. QuEry Attack generates an initial population of perturbed images by randomly selecting images from the edges of the sphere centered on the original image x with radius = ϵ. This is accomplished by adding vertical stripes of width 1 along the image, with the color of each stripe sampled uniformly at random from $\{-\epsilon, \epsilon\}$ per channel (i.e., the pixels of each stripe can be either $-\epsilon$ or ϵ); in [17], they discovered that convolutional neural networks (CNNs) are especially vulnerable to such perturbations.

4.2. Mutation

Considering the use of (square-shaped) convolutional filters by convolutional neural networks, we used square-shaped perturbations. Specifically, we employed [17]'s technique for determining square size. Let $p \in [0, 1]$ be the proportion of elements to be perturbed for an image of shape $h \times w$. The nearest positive integer to $\sqrt{p \cdot h \cdot w}$ determines the length of the square's edge, with p being a hyperparameter. We set it initially to $p = 0.1$, then halved it after {40, 200, 800, 4000, 8000, 16,000, 24,000, 32,000} queries, respectively (similar to [17]).

4.3. Crossover

We experimented both with single-point and two-point crossover, eventually settling on the latter as it performed better. The operator works by flattening both (two-dimensional image) parents, randomly picking two indices, then swapping the pixels between the chosen pixels.

The EA then proceeds by evaluating the fitness of each individual, selecting parents, and performing crossover and mutation to generate the next generation. This latter is

obtained by adding one elite individual from the current generation, with all other next-generation individuals derived through crossover and mutation. The process is repeated until a successful perturbation is found or until the termination condition is met.

A major advantage of QuEry Attack is its amenability to parallelization—due to being evolutionary—in contrast to most other adversarial, iterative (non-evolutionary) attacks in this field.

Algorithm 1 QuEry Attack

Input:

$x \leftarrow$ original image
$y \leftarrow$ original label
$\epsilon \leftarrow$ maximum l_∞ distance
$p \leftarrow$ proportion of elements in x to be perturbed
$N \leftarrow$ population size
$G \leftarrow$ maximum number of generations
$T \leftarrow$ tournament size

Output:

$\hat{x} \leftarrow$ adversarial image

Main loop
$gen \leftarrow 0$
$pop \leftarrow$ INIT()
while not TERMINATION_CONDITION(pop, gen) **do**
    for $\hat{x} \in pop$ **do**
        compute fitness of $\hat{x}$ using Equation (4)
    $new\_pop \leftarrow \emptyset$
    $elite \leftarrow$ ELITISM(pop)
    add $elite$ to new_pop
    for $i \leftarrow 1$ to $\frac{P-1}{3}$ **do**
        $parent_1 \leftarrow$ SELECTION(pop)
        $parent_2 \leftarrow$ SELECTION(pop)
        $offspring_1, offspring_2 \leftarrow$ CROSSOVER(pop)
        $mut_1 \leftarrow$ SQUARE_MUTATION($offspring_1$)
        $mut_2 \leftarrow$ SQUARE_MUTATION($offspring_2$)
        add $offspring_1, mut_1, mut_2$ to new_pop
    $pop \leftarrow new\_pop$
    $gen \leftarrow gen + 1$

return best $\hat{x}$ from pop # QuEry Attack's final output
function INIT( )
    $pop \leftarrow \emptyset$
    for $i \leftarrow 1$ to N **do**
        $\hat{x} \leftarrow$ STRIPES_INIT(x)
        add $\hat{x}$ to pop
    return pop
function ELITISM(pop)
    return best $\hat{x}$ from pop

Algorithm 1 *Cont.*

function TERMINATION_CONDITION(pop, gen)
    if $gen = G$ **then**
        return true
    for $\hat{x} \in pop$ **do**
        $\hat{y} \leftarrow$ predicted label of $\hat{x}$
        if $\hat{y} \neq y$ **then**
            return true
    return false
function SELECTION(pop)
    $tournament \leftarrow$ randomly and uniformly pick T individuals from pop
    return best $\hat{x}$ from $tournament$
function STRIPES_INIT($\hat{x}$)
    for $i \leftarrow 1$ to c **do** # c is the image's number of channels
        $stripe \leftarrow$ create a vertical stripe of width 1, randomly positioned, with random values $\in \{-\epsilon, \epsilon\}$
        $\hat{x} \leftarrow \hat{x} + stripe$
    $\hat{x} \leftarrow \Pi_\epsilon(\hat{x})$ # Π_ϵ: clipping operator to ensure pixel values are within ϵ-ball
    return $\hat{x}$
function SQUARE_MUTATION($\hat{x}$)
    $c \leftarrow$ number of channels
    $f \leftarrow$ number of features $(h \times w)$ # h: height, w: width
    $k \leftarrow \lceil \sqrt{p \cdot f} \rceil$
    $\delta \leftarrow$ array of ones of size $k \times k \times c$.
    $row, col \leftarrow \mathcal{U}(\{0, \ldots, w - h\})$ # $\mathcal{U}$ randomly and uniformly selects from given set
    for $i \leftarrow 1$ to c **do**
        $\tau \leftarrow \mathcal{U}(\{-2\epsilon, 2\epsilon\})$
        $\delta_{row+1:row+h,col+1:col+h,i} \leftarrow \tau \cdot \delta$
        $\hat{x} \leftarrow \hat{x} + \delta$
    $\hat{x} \leftarrow \Pi_\epsilon(\hat{x})$
    return $\hat{x}$
function CROSSOVER($parent_1$, $parent_2$)
    Flatten $parent_1$ and $parent_2$
    Perform standard two-point crossover (as explained in text), creating $offspring_1, offspring_2$
    $offspring_1, offspring_2 \leftarrow \Pi_\epsilon(offspring_1), \Pi_\epsilon(offspring_2)$
    return $offspring_1, offspring_2$

5. Experiments and Results

To evaluate QuEry Attack we set out to collect commonly used algorithms for comparative purposes. A somewhat disconcerting reality we then encountered involved our struggle to find good benchmarks and software for comparison purposes. Sadly, we found ourselves wasting many a day (which, alas, turned into weeks) trying to run buggy software, chasing down broken links, issuing GitHub issues, and so forth. Perhaps this is due in part to the field of adversarial attacks being young.

Our experimental results are summarized in Table 1. The code is available at https://github.com/razla/QuEry-Attack, accessed on 13 September 2022.

Table 1. Experimental results. ASR: Attack Success Rate. Queries: Number of model queries. Each value represents the median of 200 runs (images). Epsilon: $\mathcal{E} \in \{0 \dots 255\}$ (8-bits pixel values). Top results are marked in boldface.

ImageNet							
Model	$\mathcal{E}$	QuEry Attack		Square		AdversarialPSO	
		ASR	Queries	ASR	Queries	ASR	Queries
Inception-v3	24	**100%**	**1**	**100%**	5	98.5%	51
	18	**100%**	**2**	**100%**	8	98.5%	69
	12	**100%**	**22**	95.5%	32	97%	102
	6	**99.5%**	276	99.0%	**263**	95%	285
ResNet-50	24	**100%**	1	99.5%	5	98.5%	51
	18	**100%**	**1**	**100%**	5	98.5%	69
	12	**100%**	**13**	**100%**	26	97%	102
	6	**100%**	**211**	99.0%	248	95%	285
VGG-16-BN	24	**100%**	**1**	**100%**	5	98.5%	51
	18	**100%**	**1**	**100%**	5	98.5%	69
	12	**100%**	**1**	**100%**	5	97%	102
	6	**100%**	**77**	**100%**	86	95%	285
CIFAR10							
Model	$\mathcal{E}$	QuEry Attack		Square		AdversarialPSO	
		ASR	Queries	ASR	Queries	ASR	Queries
Inception-v3	24	**100%**	**1**	89.5%	5	97.5%	31
	18	**100%**	**2**	92.5%	17	96.0%	41
	12	**97.5%**	**20**	94.5%	77	95.0%	54
	6	91.0%	428	**95.5%**	776	94.5%	**223**
ResNet-50	24	**100%**	**2**	92.0%	8	97.5%	31
	18	**100%**	**8**	91.0%	23	96.0%	41
	12	**100%**	96	87.0%	110	95.0%	**54**
	6	**99.0%**	565	87.0%	449	94.5%	**223**
VGG-16-BN	24	**100%**	**1**	89.5%	5	97.5%	31
	18	**99.0%**	**2**	87.0%	14	96.0%	41
	12	**98.0%**	60	87.0%	86	95.0%	**54**
	6	**95.5%**	741	88.5%	890	94.5%	**223**
MNIST							
Model	$\mathcal{E}$	QuEry Attack		Square		AdversarialPSO	
		ASR	Queries	ASR	Queries	ASR	Queries
Conv Net	80	**100%**	**5**	86.0%	14	76%	2675
	60	93.5%	**72**	93.0%	77	**99%**	292

We evaluated QuEry Attack by executing experiments against three different pre-trained image-classification models, taken from PyTorch [31]—Inception-v3 [32], ResNet-50 [33], and VGG-16-BN [34]—over three image datasets: ImageNet, CIFAR-10, and MNIST. We employed 200 randomly picked and correctly classified images from the test sets. For ImageNet, Inception-v3 has an accuracy of 78.8%, ResNet-50 has an accuracy of 76.1%, and VGG-16-BN has an accuracy of 73.3% (these are top-1 accuracy values; for ImageNet, top-5 accuracy values are also sometimes given, which in our case are: Inception-v3—94.4%, ResNet-50–92.8%, VGG-16-BN—91.5%). CIFAR10 accuracy values are: Inception-v3—93.7%, ResNet-50—93.6%, and VGG-16-BN—94.0%. For the MNIST dataset we trained a convolutional neural network (CNN), whose architecture is delineated in Table 2, which attained 98.9% accuracy.

We chose to use the above models since they are the most commonly used CNN architectures. QuEry Attack exploits the nature of convolution layers by using square

mutations and stripes initialisation, which were shown to be very effective [17]. Further, these architectures serve as a backbone for many other downstream tasks, such as object detection [35], semantic segmentation [36], and image captioning [37]. Recently, vision transformers have beaten CNNs in image classification tasks [38], but they require huge amounts of data and resources that are not available too many (including us). Future work will be dedicated to expand the attack to vision transformers.

All accuracy values are over test images. We used the Adversarial Robustness Toolbox (ART) [39] to evaluate QuEry Attack against other attacks. We restricted all attacking algorithms to a maximum of 42K queries to the model ($N = 70$, $G = 600$) for MNIST and CIFAR10, and 84K queries ($N = 70$, $G = 1200$) for ImageNet. A query refers to a prediction supplied by the model for a given image. To make the most of the computational resources we had available we prioritized actual, experimental runs over hyperparameter runs, so hyperparameters were chosen through limited trial and error. In the future, we plan to perform a more thorough hyperparameter sweep using Optuna [40]. The only hyperparameters we set were the population size $N = 70$, tournament size $T = 25$, and $p = 0.1$; these are used for all experiments reported herein. The number of generations (G) was derived from the query budget.

AdversarialPSO [41] results were obtained by running the code in the GitHub repository referred to in their paper. Due to technical difficulties it was run against the original models that this attack was planned to run against, namely, Inception-v3 for ImageNet, and their own trained networks for CIFAR-10 and MNIST. We duplicated these results in the table for all models.

Table 2. CNN used for MNIST.

Conv Block		**Hyperparameters**	
Layer	Layer type	Hyperparameter	Value
1	Convolution	Epochs	300
2	BatchNorm2d	Batch size	64
3	ReLU	Optimizer	*Adam*
		Learning rate	0.01
		Weight decay	1×10^{-6}
CNN Architecture			
Layer	Layer type	No. channels	Filter size	Stride
1	Conv Block	32	3×3	1
2	Max Pooling	N/A	2×2	2
3	Conv Block	64	3×3	1
4	Max Pooling	N/A	2×2	2
5	Conv Block	128	3×3	1
6	Max Pooling	N/A	2×2	2
7	Dropout ($p = 0.5$)	N/A	N/A	N/A
8	Fully Connected	128	N/A	N/A
9	Fully Connected	10	N/A	N/A

5.1. Attacking Defenses

We show how QuEry Attack breaks a collection of defense strategies designed to boost the robustness of models against adversarial attacks.

5.1.1. Attacking Non-Differentiable Transformations

Gradient masking is achieved via non-differentiable input transformations, which rely on manipulating gradients to defeat gradient-based attackers [42,43]. Further, randomized transformations make it more difficult for the attacker to be certain of success. It is possible to foil such a defense by altering the defense module that performs gradient masking, but this is not an option within the black-box scenario. Herein, we investigated three non-differentiable transformations against QuEry Attack: JPEG compression, bit-depth

reduction (also known as feature squeezing), and spatial smoothing. We show that QuEry Attack can defeat these input modifications, due to its gradient-free nature.

JPEG compression [44] tries to generate patterns in color values to minimize the amount of data that has to be captured, resulting in a smaller file size. Some color values are estimated to match those of surrounding pixels in order to produce these patterns. This compression means that slight imperfections in the quality of the image will not be as noticeable. The degree of compression may be tweaked, providing a customizable trade-off between image quality and storage space. An example of the different compression degrees is shown in Figure 2. The results in Table 3 were evaluated with image quality $q = 70$.

Figure 2. JPEG compression examples, sorted from left to right by quality value, ranging from 10 to 100 (original image). (**Top**) images: from ImageNet, (**middle**): CIFAR10, and (**bottom**): MNIST.

Bit-depth reduction [45] can be done both by reducing the color depth of each pixel in an image and using spatial smoothing to smooth out individual pixel discrepancies. By merging samples that correspond to many different feature vectors in the original space into a single sample, bit-depth reduction decreases the search space accessible to an opponent. An example of different bit-depth values is shown in Figure 3. The results in Table 3 were evaluated with bit depth $d = 3$.

Figure 3. Bit-depth compression examples, sorted from left to right by bit-depth values, ranging from 1 to 8 (original image). (**Top**) images: from ImageNet, (**middle**): CIFAR10, and (**bottom**): MNIST.

The term "spatial smoothing" refers to the averaging of data points with their neighbors [46]. This has the effect of a low-pass filter, with high frequencies of the signal being eliminated from the data while low frequencies are enhanced. As a result, an image's crisp "edges" are softened, and spatial correlation within the data becomes more prominent, as shown in Figure 4. Data averaging is determined according to a given window size. The results in Table 3 were evaluated with window $w = 5$.

Our results are delineated in Table 3. For this experiment we used a total budget of 82K queries to the model ($N = 70$, $G = 1200$)—which was Inception-v3. For each given image, we first checked that it was correctly classified after applying the defense on the image, then we applied QuEry Attack on it. The different input values for the transformations were chosen such that applying them would not be destructive.

For these experiments we used the same budget of queries as in the previous experiments. For both CIFAR-10 and ImageNet, QuEry Attack has a high success rate against all non-differentiable transformations.

Table 3. QuEry Attack's resistance to non-differentiable transformation defenses. JPEG compression, bit-depth reduction, and spatial smoothing were tested on CIFAR10 and ImageNet.

Defense	$\mathcal{E}$	CIFAR10		ImageNet	
		ASR	Queries	ASR	Queries
JPEG Compression ($q = 70$)	18	100%	2	100%	2
	12	98.0%	13	97.5%	14
	6	93.5%	287	99.5%	311
Bit-Depth Reduction ($d = 3$)	18	100%	2	100%	2
	12	99.5%	5	100%	27
	6	96.0%	72	99.5%	145
Spatial Smoothing ($w = 5$)	18	100%	1	100%	1
	12	100%	1	100%	2
	6	99.0%	6	99.5%	144

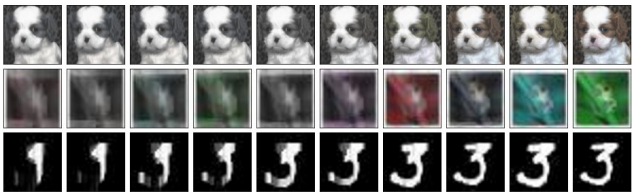

Figure 4. Examples of spatial-smoothing compression, sorted from left to right by window-size values, ranging from 10 to 1 (original image). (**Top**) images: from ImageNet, (**middle**): CIFAR10, and (**bottom**): MNIST.

5.1.2. Attacking Robust Models

A model is considered to be robust when some of the input variables are largely perturbed, but the model still makes correct predictions. Recently, several techniques have been proposed to render the models more robust to adversarial attacks. One commonly used technique to improve model robustness is adversarial training. Adversarial training integrates adversarial inputs—generated with other trained models—into the models' training data. Adversarial training has been proved to be one of the most successful defense mechanisms for adversarial attacks [47–50].

We conducted an experiment to see how well QuEry Attack performs on robust models. For CIFAR10 we used the robust model, WideResNet-70-16 [51], wherein they used generative models trained only on the original training set in order to enhance adversarial robustness to l_p norm-bounded perturbations. For ImageNet we used WideResNet-50-2 [52], which is a variant of ResNet wherein the depth of the network is decreased and the width of the network is increased. This is achieved through the use of wide residual blocks. Both of these top models were taken from the RobustBench repository [53]. We used the same 200 randomly selected images from our previous experiments, a budget of 84K queries ($N = 70$, $G = 1200$) for CIFAR10, and a budget of 126K queries ($N = 70$, $G = 1800$) for ImageNet. As seen in Table 4, QuEry Attack succeeds at breaking those strongly defended models.

Table 4. QuEry Attack's performance on robust models over CIFAR10 and ImageNet.

Model	$\mathcal{E}$	ImageNet	
		ASR	Queries
Wide ResNet-50-2	12	98.0%	98
	6	93.5%	1187
		CIFAR10	
Wide ResNet-70-16	18	94.5%	156
	12	85.0%	487

5.2. Transferability

An adversarial example for one model can often serve as an adversarial example for another model, even if the two models were trained on different datasets, using different techniques; this is known as transferability [54]. White-box attacks may overfit on the source model, as evidenced by the fact that black-box success rates for an attack are almost always lower than those of white-box attacks [7,13,55,56]. Herein, we checked transferability of our proposed black-box attack on 200 correctly classified ImageNet images by both the source model and the target model, using different ϵ values. The results, summarized in Table 5, show a positive correlation between the ϵ values and the transferability success rate. We noted that attacks are better transferred between ResNet-50 to VGG-16-BN models and surmise this is due to the fact that ResNets models were mostly inspired by the philosophy of VGG models, wherein they use relatively small 3×3 convolutional layers.

Table 5. QuEry Attack's transferability on ImageNet models. TSR: Transferability Success Rate.

Source Model → Target Model	$\mathcal{E}$	TSR
Inception-v3 → ResNet-50	24	67.0%
	18	47.5%
	12	33.5%
	6	13.5%
ResNet-50 → VGG-16-BN	24	90.0%
	18	81.5%
	12	61.5%
	6	28.5%
VGG-16-BN → Inception-v3	24	59.0%
	18	46.5%
	12	30.0%
	6	12.5%

6. Discussion and Concluding Remarks

We presented an evolutionary, score-based, black-box attack, showing its superiority in terms of ASR (attack success rate) and number-of-queries over previously published work. QuEry Attack is a strong and fast attack that employs a gradient-free optimization strategy. We tested QuEry Attack against MNIST, CIFAR10, and ImageNet models, comparing it to other commonly used algorithms. We evaluated QuEry Attack's performance against non-differential transformations and robust models, and it proved to succeed in both scenarios.

As noted, we discovered that the software scene in adversarial attacks is a tad bit muddy. We encourage researchers to place executable code on public repositories—code that can be used with ease. Furthermore, we feel that the field lacks standard means of measuring and comparing results. We encourage the community to establish common baselines for these purposes.

We came to realize the importance of a strong initialization procedure. Although this is true of many optimization algorithms, it seems doubly so where adversarial optimization is concerned. Table 1 shows that successful attacks are sometimes found during initialization—the vertical-stripes initialization in particular proved highly potent—and even if not, the number of queries (and generations) is significantly curtailed.

Figures 5 and 6 show that adversarial examples are barely distinguishable to the human eye. Clearly, neural networks function quite differently than humans, capturing entirely different features. More work is needed to create networks that are robust in a human sense.

We think that evolutionary algorithms are well-suited for this kind of optimization problems and our findings imply that evolution is a potential research avenue for developing gradient-free black-box attacks. Furthermore, evolution needs to be evaluated against a fully black-box model.

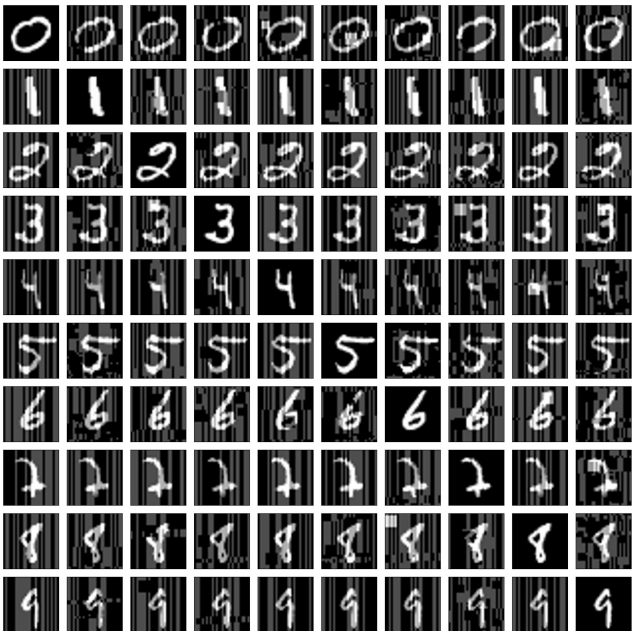

Figure 5. Adversarial examples generated by QuEry Attack on MNIST, with $\epsilon = 60/255$. An image at $row, col = i, i$ shows the original image for class i. An image at $row, col = i, j, i \neq j$ shows a targetted attack on class i, with the target being class j.

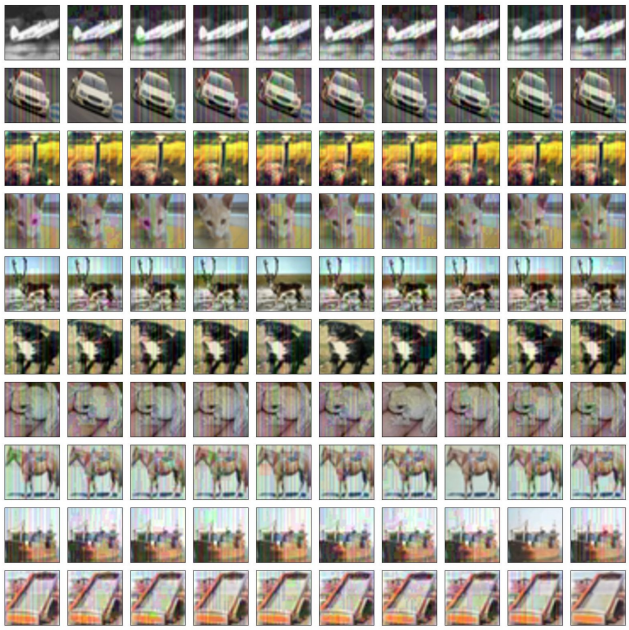

Figure 6. Adversarial examples generated by QuEry Attack on CIFAR10, with $\epsilon = 12/255$. An image at $row, col = i, i$ shows the original image for class i. An image at $row, col = i, j, i \neq j$ shows a targetted attack on class i, with the target being class j.

Evolution may also be a solution for rendering models more robust. In [57] it was shown that combining different activation functions could be used to increase model accuracy; this approach might also be used for obtaining robustness.

Author Contributions: Conceptualization, M.S.; Formal analysis, R.L.; Investigation, R.L. and Z.H.; Project administration, M.S.; Software, R.L. and Z.H.; Supervision, M.S.; Writing—original draft, R.L., Z.H. and M.S.; Writing—review & editing, R.L. and M.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We thank Ofer Hadar and Itai Dror for helpful discussions. This research was partially supported by the Israeli Innovation Authority and the Trust.AI consortium.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, X.; Jin, H.; He, K. Natural Language Adversarial Attack and Defense in Word Level. 2019. Available online: https://openreview.net/forum?id=BJl_a2VYPH (accessed on 13 September 2022).
2. Morris, J.X.; Lifland, E.; Yoo, J.Y.; Qi, Y. Textattack: A Framework for Adversarial Attacks in Natural Language Processing. Proceedings of the 2020 EMNLP. 2020. Available online: https://qdata.github.io/secureml-web/4VisualizeBench/ (accessed on 13 September 2022).
3. Carlini, N.; Wagner, D. Audio adversarial examples: Targeted attacks on speech-to-text. In Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 24–24 May 2018; pp. 1–7.
4. Schönherr, L.; Kohls, K.; Zeiler, S.; Holz, T.; Kolossa, D. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding. *arXiv* **2018**, arXiv:1808.05665.
5. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
6. Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z.B.; Swami, A. The limitations of deep learning in adversarial settings. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbruecken, Germany, 21–24 March 2016; pp. 372–387.
7. Carlini, N.; Wagner, D. Towards evaluating the robustness of neural networks. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017; pp. 39–57.
8. Gu, S.; Rigazio, L. Towards deep neural network architectures robust to adversarial examples. *arXiv* **2014**, arXiv:1412.5068.
9. Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. Deepfool: A simple and accurate method to fool deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2574–2582.
10. Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z.B.; Swami, A. Practical black-box attacks against machine learning. In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, Abu Dhabi, United Arab Emirates, 2–6 April 2017; pp. 506–519.
11. Xu, H.; Ma, Y.; Liu, H.C.; Deb, D.; Liu, H.; Tang, J.L.; Jain, A.K. Adversarial Attacks and Defenses in Images, Graphs and Text: A Review. *Int. J. Autom. Comput.* **2020**, *17*, 151–178. [CrossRef]
12. Qureshi, A.U.H.; Larijani, H.; Yousefi, M.; Adeel, A.; Mtetwa, N. An Adversarial Approach for Intrusion Detection Systems Using Jacobian Saliency Map Attacks (JSMA) Algorithm. *Computers* **2020**, *9*, 58. [CrossRef]
13. Chen, P.Y.; Zhang, H.; Sharma, Y.; Yi, J.; Hsieh, C.J. ZOO: Zeroth order optimization based black-box atacks to deep neural networks without training substitute models. In Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, Dallas, TX, USA, 3 November 2017; pp. 15–26. [CrossRef]
14. Buckman, J.; Roy, A.; Raffel, C.; Goodfellow, I. Thermometer Encoding: One Hot Way to Resist Adversarial Examples. International Conference on Learning Representations. 2018. Available online: https://openreview.net/forum?id=S18Su--CW (accessed on 13 September 2022).
15. Guo, C.; Rana, M.; Cisse, M.; Van Der Maaten, L. Countering adversarial images using input transformations. *arXiv* **2017**, arXiv:1711.00117.
16. Dhillon, G.S.; Azizzadenesheli, K.; Lipton, Z.C.; Bernstein, J.; Kossaifi, J.; Khanna, A.; Anandkumar, A. Stochastic activation pruning for robust adversarial defense. *arXiv* **2018**, arXiv:1803.01442.
17. Andriushchenko, M.; Croce, F.; Flammarion, N.; Hein, M. Square attack: A query-efficient black-box adversarial attack via random search. In Proceedings of the European Conference on Computer Vision, Online, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 484–501.
18. Kannappan, K.; Spector, L.; Sipper, M.; Helmuth, T.; La Cava, W.; Wisdom, J.; Bernstein, O. Analyzing a decade of human-competitive ("HUMIE") winners: What can we learn? In *Genetic Programming Theory and Practice XII*; Riolo, R., Worzel, W.P., Kotanchek, M., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 149–166.
19. Sipper, M.; Olson, R.S.; Moore, J.H. Evolutionary computation: The next major transition of artificial intelligence? *BioData Min.* **2017**, *10*, 26. [CrossRef]
20. Alzantot, M.; Sharma, Y.; Chakraborty, S.; Zhang, H.; Hsieh, C.J.; Srivastava, M.B. GenAttack: Practical black-box attacks with gradient-free optimization. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '19), Prague, Czech Republic, 13–17 July 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 1111–1119. [CrossRef]

21. Prochazka, S.; Neruda, R. Black-box evolutionary search for adversarial examples against deep image classifiers in non-targeted attacks. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020. [CrossRef]
22. Su, J.; Vargas, D.V.; Sakurai, K. One Pixel Attack for Fooling Deep Neural Networks. *IEEE Trans. Evol. Comput.* **2019**, *23*, 828–841. [CrossRef]
23. Das, S.; Suganthan, P.N. Differential evolution: A survey of the state-of-the-art. *IEEE Trans. Evol. Comput.* **2011**, *15*, 4–31. [CrossRef]
24. Lin, J.; Xu, L.; Liu, Y.; Zhang, X. Black-Box Adversarial Sample Generation Based on Differential Evolution. *J. Syst. Softw.* **2020**, *170*, 110767. [CrossRef]
25. Jere, M.; Rossi, L.; Hitaj, B.; Ciocarlie, G.; Boracchi, G.; Koushanfar, F. Scratch that! An evolution-based adversarial attack against neural networks. *arXiv* **2019**, arXiv:1912.02316.
26. Di Giovanni, M.; Brambilla, M. EFSG: Evolutionary fooling sentences generator. In Proceedings of the 2021 IEEE 15th International Conference on Semantic Computing (ICSC), Laguna Hills, CA, USA, 27–29 January 2021; pp. 171–178. [CrossRef]
27. Hansen, N.; Ostermeier, A. Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.* **2001**, *9*, 159–195. [CrossRef] [PubMed]
28. Wang, S.; Shi, Y.; Han, Y. Universal perturbation generation for black-box attack using evolutionary algorithms. In Proceedings of the 2018 24th International Conference on Pattern Recognition (ICPR), Beijing, China, 20–24 August 2018; pp. 1277–1282.
29. Kumar, S.K. On weight initialization in deep neural networks. *arXiv* **2017**, arXiv:1704.08863.
30. Koturwar, S.; Merchant, S. Weight initialization of deep neural networks (DNNs) using data statistics. *arXiv* **2017**, arXiv:1710.10570.
31. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An imperative style, high-performance deep learning library. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Curran Associates, Inc.: Red Hook, NY, USA, 2019; pp. 8024–8035.
32. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
33. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
34. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
35. Xavier, A.I.; Villavicencio, C.; Macrohon, J.J.; Jeng, J.H.; Hsieh, J.G. Object Detection via Gradient-Based Mask R-CNN Using Machine Learning Algorithms. *Machines* **2022**, *10*, 340. [CrossRef]
36. Zhang, R.; Du, L.; Xiao, Q.; Liu, J. Comparison of backbones for semantic segmentation network. *J. Phys. Conf. Ser.* **2020**, *1544*, 012196. [CrossRef]
37. Song, P.; Guo, D.; Zhou, J.; Xu, M.; Wang, M. Memorial GAN With Joint Semantic Optimization for Unpaired Image Captioning. *IEEE Trans. Cybern.* **2022**. [CrossRef]
38. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
39. Nicolae, M.I.; Sinn, M.; Tran, M.N.; Buesser, B.; Rawat, A.; Wistuba, M.; Zantedeschi, V.; Baracaldo, N.; Chen, B.; Ludwig, H.; et al. Adversarial Robustness Toolbox v1. 0.0. *arXiv* **2018**, arXiv:1807.01069.
40. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A next-generation hyperparameter optimization framework. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 2623–2631.
41. Mosli, R.; Wright, M.; Yuan, B.; Pan, Y. They Might NOT Be Giants Crafting Black-Box Adversarial Examples Using Particle Swarm Optimization. *arXiv* **2019**, arXiv:1909.07490.
42. Athalye, A.; Carlini, N.; Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2019; pp. 274–283.
43. Qiu, H.; Zeng, Y.; Zheng, Q.; Zhang, T.; Qiu, M.; Memmi, G. Mitigating advanced adversarial attacks with more advanced gradient obfuscation techniques. *arXiv* **2020**, arXiv:2005.13712.
44. Dziugaite, G.K.; Ghahramani, Z.; Roy, D.M. A study of the effect of jpg compression on adversarial images. *arXiv* **2016**, arXiv:1608.00853.
45. Xu, W.; Evans, D.; Qi, Y. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv* **2017**, arXiv:1704.01155.
46. Mikl, M.; Mareček, R.; Hluštík, P.; Pavlicová, M.; Drastich, A.; Chlebus, P.; Brázdil, M.; Krupa, P. Effects of spatial smoothing on fMRI group inferences. *Magn. Reson. Imaging* **2008**, *26*, 490–503. [CrossRef]
47. Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; Lempitsky, V. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.* **2016**, *17*, 2096–2130.
48. Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; McDaniel, P. Ensemble adversarial training: Attacks and defenses. *arXiv* **2017**, arXiv:1705.07204.
49. Shafahi, A.; Najibi, M.; Ghiasi, M.A.; Xu, Z.; Dickerson, J.; Studer, C.; Davis, L.S.; Taylor, G.; Goldstein, T. Adversarial Training for Free! Available online: https://proceedings.neurips.cc/paper/2019/file/7503cfacd12053d309b6bed5c89de212-Paper.pdf (accessed on 13 September 2022).

50. Wong, E.; Rice, L.; Kolter, J.Z. Fast is better than free: Revisiting adversarial training. *arXiv* **2020**, arXiv:2001.03994.
51. Gowal, S.; Rebuffi, S.A.; Wiles, O.; Stimberg, F.; Calian, D.A.; Mann, T.A. Improving robustness using generated data. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–14 December 2021; Volume 34, pp. 4218–4233.
52. Salman, H.; Ilyas, A.; Engstrom, L.; Kapoor, A.; Madry, A. Do adversarially robust imagenet models transfer better? In Proceedings of the Advances in Neural Information Processing Systems, 6–12 December 2020; Volume 33, pp. 3533–3545.
53. Croce, F.; Andriushchenko, M.; Sehwag, V.; Debenedetti, E.; Flammarion, N.; Chiang, M.; Mittal, P.; Hein, M. RobustBench: A Standardized Adversarial Robustness Benchmark. Available online: https://openreview.net/forum?id=SSKZPJCt7B (accessed on 13 September 2022).
54. Papernot, N.; McDaniel, P.; Goodfellow, I. Transferability in machine learning: From phenomena to black-box attacks using adversarial samples. *arXiv* **2016**, arXiv:1605.07277.
55. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv* **2017**, arXiv:1706.06083.
56. Brendel, W.; Rauber, J.; Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv* **2017**, arXiv:1712.04248.
57. Lapid, R.; Sipper, M. Evolution of activation functions for deep learning-based image classification. In Proceedings of the Genetic and Evolutionary Computation Conference Companion, Boston, MA, USA, 9–13 July 2022; pp. 2113–2121.