Perceptual Quality-preserving Black-Box Attack against Deep Learning Image Classifiers

Diego Gragnaniello, Francesco Marra, Giovanni Poggi and Luisa Verdoliva

Abstract—Deep neural networks provide unprecedented performance in all image classification problems, leveraging the availability of huge amounts of data for training. Recent studies, however, have shown their vulnerability to adversarial attacks, spawning an intense research effort in this field. With the aim of building better systems, new countermeasures and stronger attacks are proposed by the day. On the attacker’s side, there is growing interest for the realistic black-box scenario, in which the user has no access to the neural network parameters. The problem is to design limited-complexity attacks which mislead the neural network without impairing image quality too much, not to raise the attention of human observers. In this work, we put special emphasis on this latter requirement, and propose a powerful and low-complexity black-box attack which preserves perceptual image quality. Numerical experiments prove the effectiveness of the proposed techniques both for tasks commonly considered in this context, and for other applications in biometrics (face recognition) and forensics (camera model identification).

Index Terms—Image classification, deep learning, adversarial attacks, black-box.

I. INTRODUCTION

Deep Neural Networks (DNNs) are by now widespread in industry and society as a whole, finding application in uncontrollable fields, from the movie industry, to autonomous driving, humanoid robots, computer-assisted diagnosis, and so on. Well trained DNNs largely outperform conventional systems, and can compete with human experts on a large variety of tasks. In particular, there has been a revolution in all vision-related tasks, which now rely almost exclusively on deep-learning solutions, starting from the 2012 seminal work of Krizhevsky et al. [1] where state-of-the-art image classification performance was achieved with a convolutional neural network (CNN).

Recent studies [2], however, have exposed some alarming weaknesses of DNNs. By injecting suitable adversarial noise on a given image, a malicious attacker can mislead a DNN into deciding for a wrong class, and even force it to output a desired wrong class, selected in advance by the attacker, a scenario described in Fig. 1. What is worse, such attacks are extremely simple to perform. By exploiting backpropagation, one can compute the gradient of the loss with respect to the input image, and build effective adversarial samples by gradient ascent/descent methods. Large loss variations can be induced by small changes in the image, ensuring that adversarial samples keep a good perceptual quality.

Obviously, such findings spawned intense investigation on this topic. A first line of research was to find countermeasures to render classifiers more resilient to adversarial attacks. One natural solution is adversarial training, that is, augmenting the training set with challenging adversarial samples [3], [4], thereby hardening the classifier. With Distillation [5], [6], instead, the classifier is trained using soft labels (class probabilities from another net) rather than hard labels, so as to reduce the likelihood of leaving blind spots in the decision space. Another approach is to use ad hoc external detectors [7], [8], [9] to reveal the presence of adversarial noise, thus telling apart original from manipulated images. We refer to the recent survey from Akhtar et al. [10] for a more thorough review of these methods.

On the attacker’s side, following [2] several improvements have been proposed [11], [12], [4], [13], mostly gradient-descent methods with the gradient estimated through back-propagation. The first and most widespread approach of this category is the Fast Gradient Sign Method (FGSM) [11] together with its iterative version (I-FGSM) [3]. Such methods, however, require perfect knowledge of the network architecture and weights, a white box scenario which is hardly encountered in real-world applications. The focus is therefore shifting towards black-box attacks, where nothing is known in advance about the network structure, its weights, or the dataset used for training.

In this scenario, the attacker can query the network at will and observe the outcome. This latter can be just a hard label, the distribution of probabilities across the classes (confidence levels), or even a feature vector. Various approaches have been proposed to attack the classifier. The most promising ones rely on the use of surrogate networks [14], [15], [16] or, again, on gradient-descent [17], [18], [19], [20], [21], with the gradient now estimated by means of suitable queries to the network. In the following, we focus on this latter category, query-based...
black-box attacks.

One of the first such methods was proposed in [17], where a greedy local search is carried out to identify the pixels that, once modified, have the highest impact on the output probabilities. This kind of attack (called Pixel-based, in the following) is very effective but needs a large number of queries. In [18] a gradient estimation strategy is used, followed by the classical FGSM attack. The number of queries is limited by means of random sub-sampling and PCA-based prioritization. However, estimating the PCA basis requires significant prior information on the class of images of interest. The attack proposed in [19], instead, named ZOO, relies on the zeroth order stochastic coordinate descent algorithm and on the coordinate-wise ADAM optimizer. A variation of this method, AutoZOO, is proposed in [22] to reduce the number of queries, based on an autoencoder trained off-line on unlabeled data. This approach, however, needs the availability of an unlabeled training dataset. Other approaches choose to modify only a very limited set of pixels, like [20], where a one-pixel adversarial perturbation is applied, or [21], where the attack is extended to a group of five pixels. Such attacks, of course, can be easily spotted by an observer, and are extremely fragile to image compression or specific countermeasures.

Here, we focus on the problem of preserving the perceptual quality of images subject to black box attacks. Indeed, adversarial attacks should satisfy three main requirements at the same time: being i) effective, ii) fast, and iii) inconspicuous. The first requirement needs no comment, as it is the primary goal of the attack. Time efficiency is also a major issue in a black-box scenario, because a large number of network queries are required to perform a successful attack, which may become exceedingly slow. As for the third requirement, ensuring the smallest possible image impairment is necessary in order not to raise the attention of human observers. In fact, the attacked image should not only fool the classifier but also being accepted as genuine by the user. For example, in a forensic scenario, an un-natural looking manipulated image can be selected for further analyses, and with the large variety of complementary forensic tools available nowadays good chances are that the attack will be eventually detected.

In the current literature, un-specific quality constraints are enforced, requiring the adversarial sample to remain close to the original image in some $L_p$ norms, most often $L_1$ and $L_2$. However, it is well known that such norms do not correlate well with perceptual image quality. The presence of abnormal noise, or color inconsistencies, can be easily detected by a trained observer as a sign of possible image manipulation.

In this work, we propose a gradient-descent black box attack where the $L_p$ constraint is replaced by a constraint on a perceptual image quality measure, so as to reduce the visibility of the attack itself. In particular, we use the structural similarity (SSIM) index, proposed in [23], because of its wide acceptance in the community and its simple formulation, with gradient easily computed in closed form. Moreover, to avoid highly visible hue distortions, we modify all three color channels at once. At each step of the algorithm, we compute the current map of the SSIM gradient, and use it to identify the set of pixels which can be attacked without impacting too much on the perceptual quality. Random subsets of such pixels are then modified to query the black box, so as to gradually refine the attack until convergence. Numerical experiments prove the effectiveness of the proposed method both on classification tasks commonly considered in this context, and on some real-world forensic and biometric applications. In all cases, the proposed method ensures a much better visual quality than comparable techniques. A preliminary version of the proposed algorithm ranked first in the recent “Adversarial Attacks on Black Box Face Recognition” competition.

In the rest of the paper, we provide the necessary background (Section II), describe the proposed method (Section III), present experimental results in various scenarios of interest (Section IV), and eventually (Section V) draw conclusions.

II. BACKGROUND

A CNN used for image classification computes a function

$$y = f(x; \theta)$$

which maps an input image, $x \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, into an output vector, $y \in \mathbb{R}^{L}$. In the following, with no loss of generality, we will consider color images, hence $n_3 = 3$. The specific function implemented by the CNN depends on the model parameters, $\theta$, namely, the CNN weights which are learned during the training phase to optimize a suitable loss function.

Here, we consider two scenarios of practical interest. In the first one, the classes are known in advance, and the system is asked to decide which class the query belongs to, and how reliable the decision is. Accordingly, it provides in output a vector of probabilities

$$y(i) = \Pr(x \in C_i), \quad i = 1, \ldots, L$$

also called confidence levels, with $C_i$ the $i$-th class, and $L$ equal to the number of classes. The decision is made in favor of the maximum-probability class. In the second scenario, a scalable system is considered, where the classes are not known in advance, and their number grows with time. This applies, for example, to biometric identification systems, where new users keep being enrolled all the time. In this case the CNN is used as a feature extractor. For each input image, $x$, the CNN generates a discriminative vector of features, $y$, with length $L$ unrelated with the number of classes, which is then used to perform the actual classification, for example, with a minimum distance rule. In both cases, we assume the CNN to be already trained, with parameters $\theta$ defined once and for all, and to guarantee a satisfactory performance.

As recalled in the Introduction, CNN-based systems are vulnerable to adversarial attacks. The attacker’s goal is to modify the input image in an imperceptible way, so as to induce the system into generating a desired output. To formalize the problem, we define: $x_0$, the original image, $y_0 = f(x_0)$, the output vector associated with it, $\omega = x_0 + W$, the modified image, with $W$ the additive adversarial noise, $y^\omega = f(x^\omega)$, the output vector associated with it, and $y^f$, the target output vector. Moreover, we introduce $L(y_1, y_2)$, a suitable loss function measuring vector mismatch in the output domain, and

1 We neglect the dependence on the fixed parameters $\theta$ from now on.
$D(x_1, x_2)$, a suitable measure of the image-domain distortion. The attacker’s aim is to generate a new image, $x^a$, that is close to the original in the source domain, hence small $D(x^a, x_0)$, but whose associated vector is close to the target vector in the output domain, small $L(y^a, y^t)$, as described pictorially in Fig.1.

We cast the problem as a constrained optimization, setting a limit on the acceptable image distortion, $D_{\text{max}}$. Accordingly, the attacker looks for the image $x^a$ defined by

$$x^a = \arg \min_x L(f(x), y^t), \quad \text{s.t.} \quad D(x, x_0) \leq D_{\text{max}} \quad (3)$$

For typical classification problems, the loss of choice is the cross-entropy. Instead, when the CNN is used for feature extraction, most algorithms proposed thus far use the $L_1$ or $L_2$ norms. Instead, we will consider the SSIM (a measure of similarity, strictly speaking), in order to better take into account the perceptual distortion between original and attacked image. The structural similarity of two image patches $p$ and $q$ is defined as

$$\text{SSIM}(p, q) = \frac{(2\mu_p\mu_q + \epsilon_1)(2\sigma_{pq} + \epsilon_2)}{\mu_p^2 + \mu_q^2 + \epsilon_1(\sigma_p^2 + \sigma_q^2 + \epsilon_2)} \quad (4)$$

where $\mu_p, \mu_q, \sigma_p, \sigma_q$ are mean and standard deviation of the two patches, $\sigma_{pq}$ their covariance, and the small constants $\epsilon_1$ and $\epsilon_2$ are used to avoid division by zero. The SSIM is unitary norms with the visual perception of quality. Moreover, it is a differentiable measure, allowing one to implement simple SSIM-based attacks.

As for the attack strategies, starting from a given image, $x$, the small perturbation $\Delta W$ that maximizes the loss decrease is, by definition, proportional to the gradient of the loss itself with respect to $x$

$$\Delta W \sim \nabla_x L(f(x), y^t) \quad (5)$$

If the loss is differentiable and the CNN is perfectly known with its parameters, namely, in the white box scenario, the gradient can be computed through backpropagation, allowing very effective attack strategies. In this paper, however, we consider a black-box (BB) scenario, in which the attacker has no information about the model architecture, its parameters, $\theta$, or the training set used to learn them. However, the attacker can query the system at will, and observe the corresponding outputs. Therefore, even if the gradient cannot be directly computed, it can be estimated by means of suitable queries. For any unit-norm direction of interest, $\phi$, the attacker collects the outputs, $y^+$ and $y^-$, corresponding to the opposite queries $x^+ = x + \epsilon \phi$ and $x^- = x - \epsilon \phi$, with $\epsilon$ suitably small, and estimates the derivative of $L$ along $\phi$ as,

$$\frac{\partial L}{\partial \phi} \approx \frac{L(y^+, y^t) - L(y^-, y^t)}{2\epsilon} \quad (6)$$

If the selected directions are the image components (pixel color components) the whole gradient can be computed, even though at the cost of a large number of queries.

Actually, since querying the system has a non-negligible cost, computing the whole gradient is not a viable approach, and practical algorithms approximate gradient-based BB attacks through more efficient strategies. In addition, since images are typically quantized, for example to 256 levels, effective perturbations cannot be arbitrarily small, but must involve an integer number of levels per each pixel.

III. PROPOSED METHOD

The main peculiarity of the proposed BB gradient-based attack is its focus on preserving the perceptual quality of attacked images. We measure quality by the image SSIM, averaging patch-wise SSIM values computed over homologous patches of original and attacked image. Since SSIM measures similarity, we reformulate the attack goal as

$$x^a = \arg \min_x L_t(x) \quad \text{s.t.} \quad \text{SSIM}_0(x^a) \geq \text{SSIM}_{\text{min}} \quad (7)$$

where we have simplified notation introducing $L_t(x) = L(f(x), y^t)$ and $\text{SSIM}_0(x) = \text{SSIM}(x, x_0)$, considering that $y^t$ and $x_0$ are given once and for all.

Procedure 1 A naive SSIM-guided attack

Input: $x_0, y^t, \text{SSIM}_{\text{min}}$

Output: $x^a$

1: $x = x_0$
2: while $\text{SSIM}_0(x) > \text{SSIM}_{\text{min}}$ do
3: \hspace{1em} $G = \nabla_x \text{SSIM}_0(x)$
4: \hspace{1em} $(s^*, c^*) = \arg \min_{s, c} G(s, c)$
5: \hspace{1em} $L^+ = L_t(x + 1s^*, c^*)$
6: \hspace{1em} $L^- = L_t(x - 1s^*, c^*)$
7: \hspace{1em} if $L^+ < L^-$ then
8: \hspace{2em} $x = x + 1s^*, c^*$
9: \hspace{1em} else
10: \hspace{2em} $x = x - 1s^*, c^*$
11: \hspace{1em} end if
12: \hspace{1em} end while
13: $x^a = x$

A naive algorithm pursuing this goal is described in Procedure 1. The algorithm modifies the original image iteratively until the stopping condition on the SSIM is reached. At each step, the gradient of the SSIM with respect to the current attack, $x$, is computed, $G = \nabla_x \text{SSIM}_0(x)$, which allows to identify the image component2, $x(s^*, c^*)$, whose perturbation causes the least increase of the SSIM. Of course, the gradient provides information only on how much a perturbation reduces the SSIM, not on its effects on the loss. Therefore, the net is queried twice, by increasing/decreasing by one level only the selected component, $x \pm 1s^*, c^*$. The query that most reduces the loss becomes the new attacked image, and these steps are repeated until convergence.

This “ideal” algorithm captures faithfully the spirit of our approach but, in practice, requires too many queries, and may even not work at all. In fact, by modifying only one pixel at a time, and by the minimum possible amount, one follows a very conservative schedule, causing a very slow convergence to the solution. Moreover, the strong constraint on the pixels to modify leads to local minima with high probability. These can

2Components are indexed by 2-D spatial coordinates $s$, and color band $c$. 

be escaped in various ways, at the cost of further increasing the computational cost.

To solve these problems, we propose the perceptual quality-preserving (PQP) algorithm, with three main differences with respect to the naive version

- the direction of the perturbation is chosen at random in a large low-SSIM-gradient set;
- perturbations act on all pixel components at once, so as to better preserve the hue;
- larger perturbations are used.

We now discuss these variations in turn.

First of all, in order not to be trapped in local minima, after computing the gradient of the SSIM with respect to the current attack, we segment the image in two regions, corresponding loosely to low-gradient and high-gradient areas. A simple thresholding of the gradient is used to this end. Modified pixels are chosen randomly in the low-gradient region, which is taken large enough to allow a wide variety in the choice of the direction. Experiments discussed in Section IV suggest to enable attacks on more than half the pixels, making sure to exclude pixels that cause a sharp increase of the SSIM.

A second modification concerns hue preservation. Modifying the color components of a given pixel in opposite directions may cause a significant alteration of the pixel hue, easily perceived by a human observer. Hence, it is advisable to carry out the attack only on the luminance component. However, while computing the SSIM gradient is computationally efficient, going back and forth from the RGB to another color space has a non-negligible cost. Moreover, small changes on the luminance may disappear after back-transform and quantization. So, in order to avoid visible artifacts, we modify the three color channels of a selected pixel by the same quantity. This also causes some hue distortion, but much less noticeable that in the former case. To implement this strategy, thresholding works on the channel-wise maximum gradient \( \nabla \text{SSIM}(s,c) \) of each site \( s \).

Finally, to allow a faster convergence, we use larger perturbations. This can be achieved by modifying each pixel by \( \delta > 1 \) levels, or else by modifying \( N > 1 \) pixels at once, or combining both strategies. Considering this latter most general case, if we pick a random set of \( N \) spatial sites, \( S = \{ s_1, \ldots, s_N \} \), in the low-gradient region. Then, we define a perturbation \( \Delta W \) which is zero for \( s \notin S \) and equal to \( +\delta \) or \( -\delta \) over \( S \), more precisely

\[
\Delta W(s_i,c) = Z_i \delta, \quad i = 1, \ldots, N, \quad c = 1, \ldots, 3
\]

where the \( Z_i \)’s are independent random variables with \( \Pr(Z_i = +1) = \Pr(Z_i = -1) = 1/2 \). So, the perturbation modifies \( N \) pixels at once, increasing or decreasing them (all color channels in the same way) by \( \delta \) levels. Range boundaries are managed through obvious exceptions. As in the naive algorithm, the net is queried twice, with \( x + \Delta W \) and \( x - \Delta W \), and the query which most reduces the loss is accepted. However, with such relatively large perturbations, it may happen that neither queries reduce the loss, an event that becomes more and more likely as the attack proceeds. In such a case, a new random perturbation is generated, with the same rules. After \( k_{\text{max}} \) unsuccessful attempts, the image is modified anyway to escape a possible local minimum, an approach reminding of simulated annealing methods. Like before, we leave to experiments in Section IV the study of which values of the algorithm parameters best fit our needs.

**Procedure 2 PQP: efficient SSIM-guided attack**

**Input:** \( x_0, y^*, \text{SSIM}_{\text{min}}, \text{SSIM}_{\text{low}}, N, \delta, k_{\text{max}} \)

**Output:** \( x_a \)

1: \( x = x_0 \)
2: while \( \text{SSIM}(x) > \text{SSIM}_{\text{min}} \) and \( \ell(x) > \text{SSIM}_{\text{low}} \) do
3: \( \delta = \nabla_x \text{SSIM}(x) \) \( \triangleright \) select low-gradient pixels
4: \( k = 0 \)
5: \( \Delta \ell = 0 \)
6: \( \text{while } k < k_{\text{max}} \text{ and } \Delta \ell \geq 0 \text{ do} \)
7: \( \Delta W = \text{Perturbation}(\text{SSIM}_{\text{low}}, N, \delta) \) \( \triangleright \) generate \( \Delta W \)
8: \( \ell^+ = \ell(x + \Delta W) \) \( \triangleright \) BB query
9: \( \ell^- = \ell(x - \Delta W) \) \( \triangleright \) BB query
10: \( \Delta \ell = \min(\ell^+, \ell^-) - \ell(x) \)
11: \( k = k + 1 \) \( \triangleright \) if \( k = k_{\text{max}} \) accept anyway
12: end while
13: if \( \ell^+ < \ell^- \) then
14: \( x = x + \Delta W \)
15: end if
16: else
17: \( x = x - \Delta W \)
18: end if
19: end while
20: \( x^* = x \).

Procedure 2 provides a more formal description of the algorithm, depicted also graphically in Fig.2. The function “Segment” generates the map \( M_{\text{low}} \) with the smallest SSIM-gradient pixels, where the parameter \( Q \in [0,1] \) defines which fraction of the \( n_1 \times n_2 \) image pixels, e.g. 50\%, to keep in the map. The function “Perturbation”, instead, selects \( N \) pixels from \( M_{\text{low}} \) and generates a perturbation matrix, \( \Delta W \), which is non-zero only in the selected pixels, where a value of \( \pm \delta \) is assigned, keeping the same sign for all color components of the same pixel.

Notice that, partially deviating from the formalization of Eq.(7), the attack stops not only if the minimum allowed SSIM is reached, but also if the loss goes below a given level, \( \text{SSIM}_{\text{min}} \). This is convenient when the attack is very easy and a very small loss is obtained quickly, in which case proceeding with the attack would be a waste of resources. On the contrary, when the attack is very difficult, it may go on for an exceedingly long time. Therefore, a further stopping condition on the maximum number of iterations is included, not shown in the pseudocode for the sake of clarity.

**IV. EXPERIMENTAL ANALYSIS**

In this Section, we analyze the performance of the proposed black-box attack with reference to various classification tasks, contexts, and datasets, that is:

- general-purpose object recognition with CIFAR dataset;
- biometric face recognition with MCS2018 dataset;
- forensic source identification with SPCup2018 dataset.

Tasks and datasets will be described in detail in the following dedicated subsections.

As performance metrics we will consider attack success rate, SR, average distortion of successfully attacked images,
in terms of both PSNR (peak signal-to-noise ratio) and SSIM, and number of black-box queries, NQ, necessary to complete the attack. Note that, to ensure a meaningful comparison, PSNR, SSIM, and NQ are all computed only on successfully attacked images.

As reference methods we will consider the Pixel-based (PB) and the Random-subsampling (RS) attacks, both described in [18], with all parameters set as suggested in the original paper, to which the reader is referred for any further detail. In addition, we show results also for the white-box I-FGSM attack [3], as a sort of upper bound for the performance of black-box methods. For the proposed method, based on preliminary experiments, we set $Q = 66\%$, $N = 20$, $\delta = 1$, and $k_{\text{max}} = 20$. Robustness with respect to these parameters is analyzed in Tab. IV with reference to the challenging face recognition task.

A. Defense strategies

As reported in the literature, and confirmed by our experiments, deep learning classifiers can be attacked with success even in a black-box scenario. However, the defender can enact a number of strategies to detect or avoid such attacks. Therefore, together with the basic scenario of an unaware defender, we will also consider a more realistic scenario in which the defender adopts some simple defense strategies, independent of the attack method.

**JPEG defense.** A first straightforward and effective defense consists in including in the black-box model a suitable transformation to wash out the adversarial noise possibly introduced by the attacker. Indeed, under a signal processing point of view, an adversarial perturbation can be seen as additive high-frequency noise, effectively removed by a denoising filter or even by simple JPEG compression [24], [25]. Because of the non-differentiable nature of such transformations, the attacker cannot circumvent them, even in a white-box scenario. On the contrary, the defender can train the classifier directly on transformed images, thereby maximizing the effectiveness of the defense mechanism. We implement this defense strategy, called in the following JPEG defense, by JPEG compressing all images with quality factor 90, aligning the JPEG grid with the original 8×8 grid to avoid double compression artifacts.

**NN defense.** For very small images, such as the CIFAR10 ones, JPEG compression does not make sense. Hence we consider a second defense mechanism, at the architectural level. We build on the empirical observation that sophisticated classifiers are less robust to attacks than simpler ones. The final fully connected layers of deep nets, trained with cross-entropy loss, are rather complex classifiers, definitely prone to attacks. Therefore, as a defensive strategy, in the following NN defense, we replace this classifier with a simple nearest-neighbor classifier. In this context, the network works as a feature extractor, which is trained, by means of the triplet loss [26], to map same/different-class images into close/far features.

B. Object recognition with the CIFAR10 dataset

The popular CIFAR10 object recognition dataset comprises 60000 32×32-pixel RGB images, equally distributed among 10 classes. Due to the very low resolution, talking of perceptual quality does not make much sense, here. However, low resolution implies lightweight images and fast processing. For this reason, CIFAR10 is considered in most papers dealing with black-box attacks, and represents a standard benchmark.³ The dataset is already split in 50000 training and 10000 test samples, always equally distributed among classes.

In the experiments, we attack all the images of the test set, considering both an “easy” and a “challenging” task. In the first case, we target the class with the highest confidence score, except for the true class, while in the second case we target the class with the lowest confidence score. The attack stops either when the confidence of the selected target class exceeds 0.9, in which case we label the attack as successful, or when the SSIM goes below 0.95, a case of unsuccessful attack. Fig.3 depicts the CIFAR10 attack scenario. Classification is performed by two deep networks, ResNet32 and ResNet56 [27], trained on the CIFAR10 training set. Note that both models misclassify a small fraction of the test images, around 7%. However, when this happens, the confidence is always below 0.9, so we keep using these samples also for the “easy” task.

³This is also true for the MNIST dataset on character recognition, not included here for brevity.
Table I: Attacking ResNet32 on CIFAR10. Easy (top), hard (bottom).

| Attack                     | SR  | SSIM | PSNR | NQ |
|----------------------------|-----|------|------|----|
| white-box (I-FGSM)         | 99.65 | 0.997 | 47.52 | 3  |
| Pixel-based                | 99.61 | 0.997 | 47.54 | 10446 |
| RS                         | 99.57 | 0.994 | 45.44 | 2077 |
| PQP                        | 99.79 | 0.996 | 43.10 | 562  |
| white-box (I-FGSM)         | 98.69 | 0.991 | 42.65 | 9   |
| Pixel-based                | 99.01 | 0.991 | 42.67 | 26639 |
| RS                         | 95.84 | 0.985 | 40.84 | 5467  |
| PQP                        | 98.55 | 0.989 | 38.18 | 1846  |

Table II: Attacking ResNet56 on CIFAR10. Easy (top), hard (bottom).

| Attack                     | SR  | SSIM | PSNR | NQ |
|----------------------------|-----|------|------|----|
| white-box (I-FGSM)         | 99.16 | 0.997 | 47.09 | 4  |
| Pixel-based                | 99.63 | 0.997 | 47.10 | 10944 |
| RS                         | 99.46 | 0.994 | 45.01 | 2219  |
| PQP                        | 99.80 | 0.996 | 42.57 | 623  |
| white-box (I-FGSM)         | 98.71 | 0.991 | 42.41 | 9   |
| Pixel-based                | 99.05 | 0.991 | 42.42 | 26451 |
| RS                         | 96.01 | 0.985 | 40.59 | 5518  |
| PQP                        | 98.57 | 0.989 | 37.91 | 1842  |

Tab.I and Tab.II show results for both “easy” (top) and “challenging” (bottom) attacks to ResNet32 and ResNet56, respectively. First of all, it is clear that the two models provide almost identical results, therefore all further comments apply to both of them. A second obvious fact is that all attacks are highly successful, with a success rate (SR) always above 98.5% except for the 96% of RS on the “hard” task. Therefore, all methods attack quite easily these classifiers, ensuring also a very low distortion. The worst PSNR, about 38 dB, is observed for PQP in the “challenging” case, but even in this case, the SSIM remains very high, almost 0.99, and better than with RS. With such high success rates and good quality indicators, comparable to those of the white-box attack, the truly discriminating metric is the number of queries (NQ). Under this point of view, differences are more significant. In particular, the proposed method is about 3 times faster than RS, and 15 times faster than the Pixel-based attack (a comparison with the white-box attack makes no sense). This is very important in real-world scenarios, because the defender may discourage attacks by allowing only single-image queries or by even blocking a client after a certain number of queries. Therefore, reducing NQ may be the only way to perform the attack. PQP ensures a sharp reduction in NQ, paid for with only a minor decrease in PSNR, immaterial in practice, and no SSIM loss.

Tab.III shows results for attacks to ResNet32 in the presence of NN defense, see also Fig.4. For each test image, x, the net extracts a feature vector, called F here, computes its distance from the centroid of all classes, \( \bar{F}_t \), and decides in favor of the closest one. In the absence of attacks, this classifier is somewhat inferior to the standard one, but it proves more robust to adversarial attacks.

In this experiment, we consider the worst-case attack, where the target class is \( t = \arg \max_c \| F - \bar{F}_c \|_2 \). As before, the attack stops in two cases i) the feature vector of the attacked image, \( F^a \), becomes closer to \( \bar{F}_t \) than to all other centroids, and the distance \( \| F^a - \bar{F}_t \|_2 \) goes below a given threshold \( \gamma \) (success) or ii) the SSIM of the attacked image goes below \( \text{SSIM}_{\min}=0.95 \) (failure). Three different values are considered for \( \gamma \), 0.4, 0.3 and 0.2, with lowest values corresponding to more challenging tasks. In particular, 0.2 is the average distance between the features of a class and the corresponding centroid, while the average distance between the test images and the centroid of their target-class (distance before attack) is 0.81.

By comparing Tab.I and Tab.III, it is clear that the new classifier is more robust to attacks. Success rates reduce significantly, even for the white-box reference and for the Pixel-based method which emulates it at the cost of a very...
large number of queries. The impairment is more dramatic for the RS attack, with success rate going down to 81% for \( \gamma = 0.2 \), despite a much larger number of queries. Despite the increased robustness of the classifier, the proposed PQP attack keeps ensuring a good success rate, now the best among all methods, even in the most challenging case. Moreover, it requires the smallest number of queries among black-box methods, 3 times less than RS and (obviously) guarantees a good image quality, especially in terms of SSIM.

C. Face recognition with the MCS2018 dataset

In the second set of experiments, we attack a face recognition system. The problem was originally proposed in the "Adversarial Attacks on Black Box Face Recognition" competition, in the context of the MachinesCanSee conference (MCS2018) held in Moscow in June 2018. The competition was hosted on the Codalab platform.

The reference scenario is depicted in Fig. 5. A black-box face recognition system is provided, which extracts a 512-component unit-norm feature vector from each submitted 112×112 RGB image. This vector is then fed to a database to single out the best matching identity. In the competition, the attacker was asked to imperceptibly modify an input image associated with a given identity, with the aim of tricking the system into recognizing a different specific identity. 1000 pairs of source-target identities \((S_i, T_i)\) were provided, with 5 images for each identity; \(S_i \rightarrow \{x_{i1}^s, \ldots, x_{i5}^s\}\), and \(T_i \rightarrow \{x_{i1}, \ldots, x_{i5}\}\). The goal of the attacker was to move the feature vector of the attacked image close to those of the target identity. To make things precise, let \(F_{ij} = BB(x_{ij}^s)\) and \(G_{ij} = BB(x_{ij}^t)\) be the feature vectors generated by the black-box system for source and target images, with \(F_{ij}^a\) the feature vector associated with the attacked image \(x_{ij}^a\). The goal is to minimize

\[
D = \frac{1}{1000} \sum_{i=1}^{1000} \left( \frac{1}{25} \sum_{j=1}^{5} \sum_{k=1}^{5} \| F_{ij}^a - G_{ik} \|^2 \right) \simeq \\
\frac{1}{1000} \sum_{i=1}^{1000} \left( \frac{1}{5} \sum_{j=1}^{5} \| F_{ij}^a - \overline{G}_i \|^2 \right)
\]

with \(\overline{G}_i\) the centroid of the target identity features, subject to SSIM\(\left(x_{ij}^a, x_{ij}^s\right) \geq 0.95\). Note that this systems includes implicitly the NN defense. In the competition, we (the GRIP team) obtained eventually the lowest distance\(^5\), \(D=0.928\), using a preliminary version of the proposed algorithm.

Here, we modify slightly the attacker’s goal, in order to define a success rate and allow for a meaningful comparison with the other tasks. Not knowing the details of the classifier, we declare a success for image \(x_{ij}^a\) when \(\| F_{ij}^a - \overline{G}_i \| < \gamma\), with \(\gamma\) a suitably chosen threshold. Otherwise, if the attack stops because the SSIM goes below 0.95, a failure is declared. Since the feature vectors have unitary norm, unrelated vectors are nearly orthogonal, with a distance close to \(\sqrt{2} \simeq 1.414\). Features associated to the same identity are much closer, with average intra-class distance 0.903. Therefore, we consider three meaningful values for \(\gamma\), 1.1, 1.0, and 0.9, the latter being the most challenging.

Before proceeding to the comparative experiments, we carried out some preliminary tests to select the best parameters for the proposed method. To speed up this process, we used a subset containing only 500 source images out of the original 5000, taking care to preserve the same distribution of source-target distances, as shown in Fig.6. In the end, we chose the parameters \(Q=66\%\), \(N=20\) and \(\delta=1\). Describing in detail all these tests would be tedious. Instead, in Tab.IV we show, for \(\gamma=1\), the effect of perturbing parameters, one at a time, with respect to their default value. Note that SSIM, PSNR, and NQ are computed only on the successful attacks. Therefore, they might be overly optimistic when SR \(\ll 1\), in which case we mark results with an asterisk. These tests show, for example, that increasing \(\delta\) should be avoided, as it causes a

\(^4\)https://competitions.codalab.org/competitions/19090

\(^5\)https://competitions.codalab.org/competitions/19090#results
sharp decrease of the success rate. Also, attacking all pixels irrespective of the SSIM increase (\(Q=100\%\)) does not seem advisable, as it lowers success rate and quality with a small reduction in NQ. If necessary, NQ can be more effectively reduced by increasing \(N\) to 40.

Table V shows results for attacks to the MCS2018 face recognition system\(^6\). Given the stringent MCS2018 rules, attacks are much more difficult than in the CIFAR10 case. Even with a large distance threshold, \(\gamma=1.1\), a level which does not guarantee reliable face identification, the RS attack fails 60\% of the times. At the same level, the proposed PQP has a success rate close to 100\%, even better than I-FGSM and Pixel-based. With lower values of \(\gamma\), of course, all performance metrics worsen, and at \(\gamma=0.9\) only PQP keeps ensuring a success rate over 80\%. It is also worth underlining the significant improvement with respect to PQP-\(\alpha\), the preliminary version which ranked first in the MCS2018 competition, called “grip” in that context. Turning to the number of queries, PQP is much faster than the Pixel-based attack, and somewhat slower than RS. For this latter method, however, the PQP data are not reliable, since they refer only to the successful attacks, 40\% of the total or less in all cases. Image quality is guaranteed in all cases by the 0.95 constraint on the SSIM. In any case, and with the warning on data significance, PQP provides always the best SSIM and a pretty good PSNR.

In order to gain a better insight into the performance of all methods, we consider a “lighter” version of the MCS2018 system, where the constraint on the SSIM is replaced by a much weaker constraint on PSNR, required only to exceed 30 dB. With these rules, success rates are always very high, never less than 98\%, allowing for the collection of significant performance data. In these conditions, PQP is about as fast as RS, but ensures a much better quality, in terms of both PSNR and SSIM. For the most interesting case of \(\gamma=0.9\), PQP keeps providing an average SSIM above 0.95, while for RS this drops well below 0.9, which entails very likely visible distortions.

This is confirmed by Fig.7, showing visual results for the various types of attacks on two MCS2018 images. Apparently, the “bearded man” image is relatively simple to attack, and all methods introduce only limited distortion. Nonetheless, for \(\gamma=0.9\), weird geometrical patterns and color distortions appear in the flat areas of images attacked by all methods except PQP. The “girl” image is obviously more difficult to attack, and visible distortions arise even for \(\gamma=1.1\). Also in this case, however, the PQP images appear more natural than the others, showing a lower quality than the original but without heavy patterns and color distortions.

### D. Results on the SPCup2018 dataset

Finally, we consider the source identification problem proposed in the 2018 IEEE Signal Processing Cup. The goal was to identify which one of a given set of cameras acquired the image under test, without the help of any metadata, see Fig.8. A dataset of 2250 images was provided, called SPCup2018 dataset from now on, with 2000 images for training and 250 for validation, evenly distributed among 10 different camera models. The open phase of the competition was hosted on Kaggle\(^7\), followed by a restricted final phase held in the context of IEEE ICASSP, in Calgary, march 2018.

For the competition we proposed a solution based on the ensemble of several very deep CNNs, with a suitable fusion of patch-level results\(^8\). Here, however, we consider a much simpler classifier, with a single deep network, XceptionNet\(^9\), which proved very powerful for camera model identification [29]. The net is trained on 96×96-pixel image patches. In the absence of attacks, the classifier has an error rate of 8.4\%, but

\(^{6}\)Note that, to carry out experiments in a reasonable time, we implemented a replica of the competition system based on the organizer’s description, and testing strict compliance of results.

\(^{7}\)https://www.kaggle.com/c/sp-society-camera-model-identification

\(^{8}\)https://www.kaggle.com/c/sp-society-camera-model-identification/leaderboard

| Attack         | SR   | SSIM | PSNR  | NQ  |
|----------------|------|------|-------|-----|
| white-box (I-FGSM) | 97.12 | 0.981 | 42.14 | 6   |
| Pixel based     | 95.90 | 0.978 | 41.53 | 337965 |
| RS             | 41.80 | *0.966 | *39.14 | *7035 |
| PQP-\(\alpha\) (grip) | 90.22 | 0.970 | 35.60 | 8329 |
| PQP            | 99.80 | 0.985 | 39.21 | 8335 |

| \(\gamma=1.1\) | white-box (I-FGSM) | 84.62 | 0.975 | 40.40 | 7   |
|----------------|-------------------|-------|------|-------|-----|
| Pixel based     | 81.74 | 0.971 | 39.84 | 454743 |
| RS             | 16.44 | *0.962 | *38.04 | *8217 |
| PQP-\(\alpha\) (grip) | 66.68 | *0.965 | *34.46 | *11883 |
| PQP            | 95.80 | 0.978 | 37.13 | 13400 |

| \(\gamma=1.0\) | white-box (I-FGSM) | 62.76 | *0.970 | *39.25 | *9   |
|----------------|-------------------|-------|------|-------|-----|
| Pixel based     | 56.86 | *0.967 | *38.93 | *536919 |
| RS             | 4.66  | *0.959 | *37.33 | *9264 |
| PQP-\(\alpha\) (grip) | 39.88 | *0.962 | *33.87 | *14550 |
| PQP            | 83.52 | 0.973 | 35.90 | 18244 |

| \(\gamma=0.9\) | white-box (I-FGSM) | 98.98 | 0.949 | 37.63 | 15  |
|----------------|-------------------|-------|------|-------|-----|
| Pixel based     | 99.16 | 0.944 | 37.31 | 893743 |
| RS             | 98.36 | 0.878 | 33.56 | 27410 |
| PQP-\(\alpha\) (grip) | 97.20 | 0.939 | 33.05 | 59440 |
| PQP            | 99.60 | 0.965 | 35.39 | 30967 |

| Attack         | SR   | SSIM | PSNR  | NQ  |
|----------------|------|------|-------|-----|
| white-box (I-FGSM) | 100.00 | 0.980 | 42.00 | 6   |
| Pixel based     | 100.00 | 0.976 | 41.37 | 351987 |
| RS             | 100.00 | 0.939 | 37.05 | 11133 |
| PQP-\(\alpha\) (grip) | 99.98 | 0.967 | 35.39 | 9377 |
| PQP            | 100.00 | 0.985 | 39.20 | 8378 |

| \(\gamma=1.1\) | white-box (I-FGSM) | 100.00 | 0.968 | 39.71 | 9   |
|----------------|-------------------|-------|------|-------|-----|
| Pixel based     | 100.00 | 0.964 | 39.18 | 544140 |
| RS             | 99.98 | 0.912 | 35.02 | 17498 |
| PQP-\(\alpha\) (grip) | 99.86 | 0.954 | 33.90 | 21286 |
| PQP            | 100.00 | 0.977 | 36.98 | 14620 |

| \(\gamma=1.0\) | white-box (I-FGSM) | 98.98 | 0.949 | 37.63 | 15  |
|----------------|-------------------|-------|------|-------|-----|
| Pixel based     | 99.16 | 0.944 | 37.31 | 893743 |
| RS             | 98.36 | 0.878 | 33.56 | 27410 |
| PQP-\(\alpha\) (grip) | 97.20 | 0.939 | 33.05 | 59440 |
| PQP            | 99.60 | 0.965 | 35.39 | 30967 |
the wrong class has a relatively low confidence. Therefore, we consider the “easy” attack, targeting the best wrong class. Table VII shows results on the images of the validation set, both in the absence of any defense and in the presence of a defense based on JPEG compression with QF=90. In the absence of defenses, the proposed method outperforms all other techniques for all performance metrics. The success rate is at least 10% higher than the references, including I-FGSM and Pixel-based, with a better image quality. Compared with PQP, RS requires also 20% more queries, while the Pixel-based approach is basically useless, calling for more than 100000 queries.

When the JPEG compression is performed, some interesting results are obtained. First of all, NQ increases significantly for all methods, testifying on the higher difficulty of the problem. However, contrary to intuition, the Pixel-based method achieves a much better performance, with success rate close to 100%. This happens because most small perturbations occurring in homogeneous areas are simply canceled by the JPEG compression. Therefore, actual perturbations take place in the same areas selected by our approach, and similar results are eventually obtained, at a much higher cost. The RS technique, instead, is strongly impaired by the defense strategy, with a success rate dropping below 30%. As for the proposed method, the defense has little or no effect, besides the increased number of queries. Again, this is explained by the fact that the attack concerns higher-activity regions which are barely affected by JPEG compression.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a new method for attacking deep learning-based classifiers, aimed at better preserving the perceptual quality of images. Experiments carried out in three quite different scenarios testify on the potential of the proposed approach. Results are especially good when some defense strategies are enacted. The visual inspection of attacked images shows indeed a significant improvement in perceptual quality, even though visible distortion is introduced on challenging images. However, this is also due to the relatively small size of images used in the experiments, necessary to conduct large scale validation. Our conjecture, supported by sample tests, is that effective adversarial noise can be more easily introduced on large images.

As next step of this research, we plan to extend our perceptual quality-preserving approach to the white-box context.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” in Neural Information Processing Systems (NIPS), 2012, pp. 1097–1105.
[2] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in International Conference on Learning Representations, 2014.
[3] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” arXiv preprint arXiv:1607.02533, Jul. 2016.
[4] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” arXiv:1706.06083, 2017.
[5] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” NIPS Deep Learning Workshop, 2014.
[6] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, “Distillation as a defense to adversarial perturbations against deep neural networks,” in 2016 IEEE Symposium on Security and Privacy (SP), May 2016, pp. 582–597.

[7] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, “On detecting adversarial perturbations,” in International Conference on Learning Representations (ICLR), 2017.

[8] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. D. McDaniel, “On the (statistical) detection of adversarial examples,” arXiv preprint arXiv:1702.06280, 2017.

[9] D. Meng and H. Chen, “Magnet: A two-pronged defense against adversarial examples,” in ACM SIGSAC Conference on Computer and Communications Security, 2017, pp. 135–147.

[10] N. Akhtar and A. Mian, “Threat of adversarial attacks on deep learning in computer vision: a survey,” arXiv preprint arXiv:1801.00553, 2018.

[11] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in International Conference on Learning Representations, 2015.

[12] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in IEEE Symposium on Security and Privacy, 2017, pp. 39–57.

[13] M. Cisse, Y. Adi, N. Neverova, and J. Keshet, “Houdini: Fooling deep structured prediction models,” arXiv preprint arXiv:1707.05373, 2017.

[14] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples,” https://arxiv.org/abs/1605.07277, 2016.

[15] S. M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “Deepfool: A simple and accurate method to fool deep neural networks,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2574–2582.

[16] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. Celik, and A. Swami, “Practical black-box attacks against deep learning systems using adversarial examples,” in ACM Asia Conference on Computer and Communications Security, 2017.

[17] N. Narodytska and S. Kasiviswanathan, “Simple black-box adversarial perturbations for deep networks,” in CVPR Workshops, 2017.

[18] A. Bhagoji, W. He, B. Li, and D. Song, “Exploring the space of black-box attacks on deep neural networks,” arXiv preprint arXiv:1712.09491, 2017.

[19] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, “Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models,” in Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, 2017, pp. 15–26.

[20] J. Su, D. Vargas, and K. Sakurai, “One pixel attack for fooling deep neural networks,” arXiv preprint arXiv:1710.08864, 2017.

[21] ——, “Attacking convolutional neural network using differential evolution,” arXiv preprint arXiv:1804.07062, 2018.

[22] C.-C. Tu, P. Ting, P.-Y. Chen, S. Liu, H. Zhang, J. Yi, C.-J. Hsieh, and S.-M. Cheng, “Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks,” arXiv preprint arXiv:1805.11770, 2018.

[23] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600–612, April 2004.

[24] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and D. H. Chau, “Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression,” arXiv preprint arXiv:1705.02900, 2017.

[25] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, “Countering adversarial images using input transformations,” arXiv preprint arXiv:1711.00117, 2017.

[26] K. Q. Weinberger and L. K. Saul, “Distance metric learning for large margin nearest neighbor classification,” Journal of Machine Learning Research, vol. 10, no. Feb, pp. 207–244, 2009.

[27] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[28] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1800–1807.

[29] F. Marra, D. Gragnaniello, and L. Verdoliva, “On the vulnerability of deep learning to adversarial attacks for camera model identification,” Signal Processing: Image Communication, vol. 65, pp. 240–248, July 2018.