What if Adversarial Samples were Digital Images?

Benoît Bonnet, Teddy Furon, Patrick Bas

What if Adversarial Samples were Digital Images?

Benoît Bonnet
benoit.bonnet@inria.fr
Univ. Rennes, Inria, CNRS, IRISA
Rennes, France

Teddy Furon
teddy.furon@inria.fr
Univ. Rennes, Inria, CNRS, IRISA
Rennes, France

Patrick Bas
patrick.bas@centralelille.fr
Univ. Lille, CNRS, Centrale Lille, UMR 9189, CRIStAL, Lille, France

ABSTRACT

Although adversarial sampling is a trendy topic in computer vision, very few works consider the integral constraint: The result of the attack is a digital image whose pixel values are integers. This is not an issue at first sight since applying a rounding after forging an adversarial sample trivially does the job. Yet, this paper shows theoretically and experimentally that this operation has a big impact. The adversarial perturbations are fragile signals whose quantization destroys its ability to delude an image classifier.

This paper presents a new quantization mechanism which preserves the adversariality of the perturbation. Its application outcomes to a new look at the lessons learnt in adversarial sampling.

CCS CONCEPTS

• **Security and privacy** → *Domain-specific security and privacy architectures*; **Intrusion/anomaly detection and malware mitigation**; **Malware and its mitigation**;

KEYWORDS

Image classification, neural networks, adversarial samples

ACM Reference Format:
Benoît Bonnet, Teddy Furon, and Patrick Bas. 2020. What if Adversarial Samples were Digital Images?. In *Proceedings of -*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Adversarial samples [12] are small, usually imperceptible perturbations of images (or other data) that can arbitrarily modify the prediction of a classifier. The Computer Vision community has extended adversarial samples to other tasks than image classification like optical flow computation [10], object tracking [15], captioning [16], face recognition[2], and image retrieval [8, 13]. These perturbations are not random but carefully crafted by an attacker. In a *white-box* setting, the attacker has full knowledge of the classifier internals and uses the gradient of the model to find the appropriate perturbation for a given image. They are becoming increasingly important because they reveal the *sensitivity* of neural networks to their inputs. That sensitivity is a vulnerability when the system is deployed in security application.

Adversarial samples are typically evaluated by the *probability of success*, *i.e.* the probability that the attack deludes the classifier, and by the *distortion* between the original and the attacked images. State-of-the-art white-box attacks lead to a probability of success near one combined with a small distortion. This shows that the perturbation is almost imperceptible and reflects the difficulty with which adversarial samples can be detected. The *speed* of an attack is another criterion recently introduced. The fast single-step FGSM attack [4] produces high-distortion examples where adversarial patterns can easily be recognized. At the other extreme, the *Carlini & Wagner* (CW) attack [1], considered state of the art, is notoriously expensive. *Decoupling direction and norm* (DDN) [11] has recently shown impressive progress in the trade-off between distortion and speed. Speed is important for adversarial retraining [9]. This procedure robustifies a network by training it with many adversarial samples.

A perusal of the 25 papers dealing with this topic and recently published in the 2019 editions of the well known conferences CVPR and ECCV shows the following fact: 88% of these research works propose attacks forging adversarial samples which are not images! Their outputs are *adversarial samples* in the form of matrices with continuous variables (implemented with float point single precision in 4 Bytes). This paper challenges this working assumption and investigates what happens if attacks are constrained to forge *digital images* with discrete pixel values encoded with a depth of 8 bits. Several works have addressed this issue using quantized steps within their iterative attacks [11, 14]. Our goal is however *not* to propose a new attack, but to see how to quantify adversarial samples at best. This is not trivial : Rounding each pixel value is an obvious but inefficient solution that almost always turns an adversarial sample into a *non* adversarial image.

The paper contains four contributions:

- A theoretical explanation why quantization by rounding fails.
- A near optimal quantization procedure that keeps the adversarial nature and the small distortion of the perturbation
- A review which analyses whether well-known facts in the adversarial sample literature still hold with near optimal quantization.
- The integration of our quantization scheme into an iterative attack.

The outline of the paper is the following. Section 2 first challenges the working assumption in literature that adversarial samples need not to be quantized. Section 3 shows that simply rounding the adversarial samples does not yield adversarial images. Section 4 presents our near optimal quantization procedure. Section 5 reports experimental results over ImageNet when quantization is applied after some classical attacks against 'natural' and 'robust' networks. Section 6 exposes how to integrate our quantization inside an iterative attack.

2 MOTIVATION

Adversarial attacks in the literature usually output real numbers stored in matrices. We call them *adversarial samples* in contrast

-, -, -
2020. 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

to *digital images* that are tables of integers. There are many pre-processing recipes adapting the input image before feeding it to the neural network. This section argues that these steps cannot explain the assumption of working with unquantized adversarial samples too easily admitted in this literature.

Over the 25 publications dealing with adversarial attacks in CVPR 2019 and ECCV 2019, almost all of them define a pixel as a real value unquantized. They consider the problem of find a real matrix close to the original image and whose prediction is wrong. The words 'quantization' or 'discrete' are never mentioned. The images included in these papers as illustration of adversarial samples have been quantized for publication purpose. Therefore, it is not sure that these published images indeed are adversarial.

Only three of these 25 papers are clearly working with quantized images. Two of them indeed investigate attacks in the *physical world* to delude optical flow reconstruction in autonomous cars [10] or object tracking in video surveillance [15]. They obviously deal with quantized images because they print patches on stickers or they display patterns on a screen present in the scene. The only reference coping with *digital* adversarial samples which are truly images proposes the so-called DDN attack: In [11, Sect. 3], one reads "*Besides this step, we can also consider quantizing the image in each iteration, to ensure the attack is a valid image*". It is very surprising that the authors compares DDN with state-of-the-art but unconstrained attacks issuing adversarial samples not images.

In the same way, the report [7] on the "*Adversarial Attacks and Defences Competition*" organised at NIPS 2017 clearly states that "*the adversary has direct access to the actual data fed into the model. In other words, the adversary can choose specific float32 values as input for the model*". There is clearly here a misconception in the threat analysis. In the *white box* scenario, the attacker knows all the internals of the targeted network. She/He can reproduce it in her/his garage and has "*direct access*" into this copy. Yet, the goal is to produce an adversarial image that will delude the same classifier outside the garage, where access inside is forbidden. For example, if the attacker knows the source code of the classifier used to tag images on a social network platform, he still needs to publish the attacked image in a format readable by the platform, for example in tiff or jpeg format for images.

Note that a common procedure in image classification is to apply a transformation to the pixel values of the query image before feeding the neural network. Contrary to the vast majority of adversarial examples represented in the continuous domain, these processed inputs remain discrete. This is an affine scaling ranging from $\mathcal{V} := \{0, 1, \ldots, 255\}$ to $[0, 1]$ (or $[-1, 1]$ depending on the neural network). Even if the output is encoded as float, this bijection results in 255 discretized possible values in $[0, 1]$. Yet, most of papers explicitly state that the attack aims at finding the adversarial sample anywhere in $[0, 1]^n$ with the smallest distortion.

Another practice is to perform an object (ImageNet dataset) or a character (MNIST dataset) detection first and then to query the network with the cropped image framing the object. If the bounding box is too large, a stretching is performed (see the CIFAR dataset). The result of this downscaling need not to be quantized and again can be encoded as float. Nevertheless, this process is done inside the image classifier, and the attacker has no direct access to this auxiliary data outside the garage. Consequently, the aim of the attacker is to modify the image before it is transmitted to the classifier, and therefore before any of these pre-processing tasks.

The report [7] states that "*In a real world setting, this might occur when an attacker uploads a PNG file to a web service, and intentionally designs the file to be read incorrectly.*" This motivates the choice of the floating point representation for unquantized pixel values. On one hand, this community has well accepted that these values should only lie in $[0, 1]$. So, if the attacker is so powerful that she/he can inject any floating point value, then why should we restrict the range to $[0, 1]$? On the other hand, we argue here that this is a security threat not targeting the classifier but the image loader software that decodes the received file into a matrix of pixel values. There are known conception rules in computer security to avoid hacks like buffer overflows.

There are almost as many defense papers as attack papers in this literature. Authors validate that their defense does not degrade the classifier accuracy on the original images of the test dataset, which are of course quantized. In the same time, they measure the defense ability on the attacked versions of these images, which are not quantized. It is funny to notice that detecting whether the input is quantized would be the simplest defense able to block most of the attacks.

The next section shows that quantization is not an inoffensive processing step: it can strongly impact the success rate of an attack.

3 PROBLEM FORMULATION

This section introduces basic notations of image classification with neural network, briefly presents some well known white-box attacks, and exposes the impact of quantization when rounding weak perturbations.

3.1 Background

3.1.1 Notations. Let $\mathcal{X} := \{0, 1, \ldots, 255\}^n$ denote the set of *images*. This means that, for sake of clarity, an image is a flattened vector $\mathbf{x} \in \mathcal{X}$ whose length n is the total number of pixels (grayscale like MNIST dataset) times 3 color channels (ImageNet or CIFAR dataset). The classifier usually pre-processes the input image by a function $\mathrm{a} : \mathcal{X} \to \mathcal{S}$ where $\mathcal{S} := [0, 1]^n$ (or $\mathcal{S} := [-1, 1]^n$ for some implementations of ResNet). We call $\mathbf{s} := \mathrm{a}(\mathbf{x}) \in \mathcal{S}$ a sample. A classifier is a function $\mathrm{f} : \mathcal{S} \to \mathcal{P}_C$ where $\mathcal{P}_C$ is the simplex of dimension C: $\mathbf{p} = \mathrm{f}(\mathbf{s})$ is a vector of C positive components summing up to 1, $p(k)$ being the predicted probability that sample $\mathbf{s}$ belongs to class $k \in [C]$ (with $[n] := \{1, 2, \ldots, n\}$). The classifier top-1 prediction $\pi : \mathcal{S} \to [C]$ maps the sample $\mathbf{s}$ to the class label having the maximum probability:

$$\pi(\mathbf{s}) := \arg\max_{k \in [C]} p(k). \tag{1}$$

The prediction is correct if $\pi(\mathbf{s}) = \mathrm{t}(\mathbf{s})$, the *true label* of sample $\mathbf{s}$.

3.1.2 Problem formulation. In the *untargeted* scenario, the aim of the attacker is to delude the classifier in whatever manner, *i.e.* its predicted class is not the true label.

In the literature, an adversarial sample $\mathbf{s}_a$ is a quasi-copy of a given original sample $\mathbf{s}_o$ where $\pi(\mathbf{s}_a) \neq \mathrm{t}(\mathbf{s}_o)$ although $\|\mathbf{s}_a - \mathbf{s}_o\|_L$ is small (the L-norm of $\mathcal{S}$, with $L \in \{0, 1, 2, +\infty\}$).

In this paper, an adversarial image $\mathbf{x}_q$ is a quasi-copy of an original image $\mathbf{x}_o$ where $\pi(\mathsf{a}(\mathbf{x}_q)) \neq \mathsf{t}(\mathsf{a}(\mathbf{x}_o))$. The constraint is that $\mathbf{x}_q$ is a digital image, *i.e.* it belongs to $\mathcal{X}$. The distortion in this paper is measured by the Euclidean norm in the image pixel domain $\mathcal{X}$ (and not in $\mathcal{S}$ as in many papers).

The framework considers an original image $\mathbf{x}_o$, the output of an attack $\mathbf{x}_a$ which is *not* a priori a digital image, and the quantization of $\mathbf{x}_a$ into $\mathbf{x}_q \in \mathcal{X}$. We denote by $\mathbf{u}$ the unquantized perturbation, $\mathbf{q}$ the quantization noise and by $\mathbf{e}$ the final adversarial perturbation:

$$\mathbf{u} := \mathbf{x}_a - \mathbf{x}_o, \tag{2}$$

$$\mathbf{q} := \mathbf{x}_q - \mathbf{x}_a, \tag{3}$$

$$\mathbf{e} := \mathbf{x}_q - \mathbf{x}_o = \mathbf{u} + \mathbf{q}. \tag{4}$$

3.1.3 The classification loss. In a white-box scenario, the attacker gauges how close he/she is from his/her goal with a measure called the classification loss. This is typically the negative cross-entropy $\mathsf{L}_A(\mathbf{s}_a) = \log p_a(\mathsf{t}(\mathbf{s}_o))$ (with $\mathbf{p}_a := \mathsf{f}(\mathbf{s}_a)$) or whatever increasing function of $p_a(\mathsf{t}(\mathbf{s}_o))$. The role of the attack is to decrease this loss so that $p_a(\mathsf{t}(\mathbf{s}_o))$ is so small that the sample is no longer classified as the ground truth. In other words, $\mathbf{s}_a$ is repelled from the original class region.

Another option is to attract $\mathbf{s}_a$ to another class region, for instance the most likely other prediction:

$$\mathsf{L}_A(\mathbf{s}_a) = \log p_a(\mathsf{t}(\mathbf{s}_o)) - \log \max_{k \neq \mathsf{t}(\mathbf{s}_o)} p_a(k). \tag{5}$$

This has the advantage of indicating by $\mathsf{L}_A(\mathbf{s}_a) < 0$ that the attack succeeds. Indeed, if $\mathsf{L}_A(\mathbf{s}_a) = m < 0$ then $p_a(\mathsf{t}(\mathbf{s}_o))$ is e^m smaller than the estimated probability of the predicted class.

3.2 Well known attacks

This section summarizes well-known attacks in the literature, which we consider in the experimental body in Sect. 5. The linear preprocessing $\mathsf{a}(\cdot)$ mapping each pixel to $\mathcal{S}$ is integrated in the neural network, and hence in the loss L_A. This allows to describe the attacks in the domain $[0, 255]^n$.

Fast Gradient Sign Method. FGSM is the oldest and simplest attack [4]. It has one unique parameter $\epsilon > 0$. Its expression is simply:

$$\mathbf{x}_a = \mathsf{cl}\left(\mathbf{x}_o - \epsilon \mathsf{sign}\left(\nabla_{\mathbf{x}} \mathsf{L}_A(\mathbf{x})|_{\mathbf{x}_o}\right)\right), \tag{6}$$

where cl clips the pixel values to $[0, 255]$. Note that $\mathbf{x}_a \in \mathcal{X}$ if and only if $\epsilon \in \mathbb{N}$. The final distortion is $\|\mathbf{x}_a - \mathbf{x}_o\|^2 = n\epsilon^2$ (neglecting the clipping).

Iterative FGSM.. This is the iterated version of FGSM introduced in [6]. We consider the version that repeats the update:

$$\mathbf{x}_a^{(i+1)} = \mathsf{cl}\left(\mathbf{x}_a^{(i)} - \alpha \mathsf{sign}\left(\nabla_{\mathbf{x}} \mathsf{L}_A(\mathbf{x})|_{\mathbf{x}_a^{(i)}}\right)\right), \tag{7}$$

until a maximum number N of iterations is met or until $\mathbf{x}_a^{(i)}$ is adversarial. It has two parameters α and N. The distortion is at most $n(\alpha N)^2$ (achieved if the gradient is a constant vector).

Projected Gradient Descent. We refer to PGD_2 as the Euclidean version of the projected gradient descent [9]. Its update is given by

$$\mathbf{x}_a^{(i+1)} = \mathsf{cl}\left(\mathsf{proj}_\epsilon\left(\mathbf{x}_a^{(i)} - \alpha \frac{\nabla_{\mathbf{x}} \mathsf{L}_A(\mathbf{x})|_{\mathbf{x}_a^{(i)}}}{\| \nabla_{\mathbf{x}} \mathsf{L}_A(\mathbf{x})|_{\mathbf{x}_a^{(i)}} \|}\right)\right) \tag{8}$$

where proj_ϵ is the projection on the ball of center $\mathbf{x}_o$ and radius ϵ. It means that the update is scaled back onto the sphere of radius ϵ if it goes outside that ball. This attack has three parameters: α, ϵ and the maximum number N of iterations. Usually, we set ϵ as a fraction of αN.

Carlini and Wagner. We refer to CW as the attack invented by Carlini and Wagner, authors of [1]. It uses the ADAM solver to find the minimum of the Lagrangian formulation:

$$J(\mathbf{x}, \mu) = \|\mathbf{x} - \mathbf{x}_o\|^2 + \mu \, |\mathsf{L}_A(\mathbf{x}) - m|_+, \tag{9}$$

where $m \leq 0$ is a margin and $|a|_+ = a$ if $a > 0$, and 0 otherwise. Then, an outer loop tests different values of μ in a line search. The adversarial sample with the lowest distortion is the final output. The parameters are usually the number of iterations for the inner loop (ADAM) and for the outer loop.

Decoupling Direction and Norm. This attack denoted DDN is defined in [11] by its update:

$$\mathbf{x}_t^{(i+1)} = \mathbf{x}_a^{(i)} - \alpha \frac{\nabla_{\mathbf{x}} \mathsf{L}_A(\mathbf{x})|_{\mathbf{x}_a^{(i)}}}{\| \nabla_{\mathbf{x}} \mathsf{L}_A(\mathbf{x})|_{\mathbf{x}_a^{(i)}} \|}, \tag{10}$$

$$\mathbf{x}_a^{(i+1)} = \mathsf{cl}\left(\mathbf{x}_o + \rho^{(i+1)} \frac{\mathbf{x}_t^{(i+1)} - \mathbf{x}_o}{\|\mathbf{x}_t^{(i+1)} - \mathbf{x}_o\|}\right). \tag{11}$$

where $\rho^{(i+1)} = (1+\gamma)\rho^{(i)}$ if $\mathbf{x}_a^{(i)}$ is not adversarial and $\rho^{(i)} = (1-\gamma)\rho^{(i)}$ otherwise (with $\rho^{(1)} = \alpha$). Since $\gamma > 0$, DDN increases (resp. decreases) the budget distortion $\rho^{(i+1)}$ if $\mathbf{x}_a^{(i)}$ is still not adversarial (resp. is already adversarial). In its quantized version, the function $\mathsf{cl}(\cdot)$ not only clips to $[0, 255]$ but also rounds each component to the nearest integer. This is done at the end of each iteration. The adversarial sample with the lowest distortion is the final output. This attack has 3 parameters: α, γ, and N.

3.3 Why rounding fails

We suppose that an attack produces $\mathbf{x}_a = \mathbf{x}_o + \mathbf{u}$ which a priori does not belong to the set of discrete values $\mathcal{X}$. A solution is then to quantize back onto $\mathcal{X}$ by applying the rounding to the nearest integer $\mathsf{R}(\cdot)$:

$$\mathbf{x}_q = \mathsf{R}(\mathbf{x}_o + \mathbf{u}) = \mathbf{x}_o + \mathsf{R}(\mathbf{u}), \tag{12}$$

where we make the abuse of notation: $\mathsf{R}(\mathbf{x})$ means rounding each component of the vector $\mathbf{x}$. We also assume that $\mathbf{x}_o + \mathsf{R}(\mathbf{u}) \in \mathcal{X}$ without clipping. The last equality comes from the fact that $\mathbf{x}_o \in \mathcal{X}$.

The following study aims at predicting the norm of the update after quantization, assuming that rounding is independent from the computation of the perturbation. Denote by $\mathbf{e} := \mathsf{R}(\mathbf{u})$. Pixel j is quantized to

$$x_q(j) = x_o(j) + e(j), \tag{13}$$

when $u(j) \in (e_j - 1/2, e_j + 1/2]$ for some $e(j) \in \mathbb{Z}$. Border effects where $x(j) + e(j) \notin \mathcal{X}$ are neglected here.

We now take a statistical point of view where the update is modelled by a random vector $\mathbf{U}$ uniformly distributed over the hypersphere of radius ρ. That parameter ρ is the norm of the perturbation *before* quantization. This yields random quantized values,

denoted by $E(j) \in \mathbb{Z}$ for pixel j. The distortion *after* the quantization is given by:

$$D^2 = \|\mathbf{E}\|^2 = \sum_{j=1}^{n} E(j)^2. \tag{14}$$

A common approach in source coding theory is the additive noise model for quantization error in the high resolution regime [3]. It states that $E(j) = U(j) + Q(j)$ where $Q(j) \in (-1/2, 1/2]$ is the quantization error. In the high resolution regime where $\rho \gg 1$, Q becomes uniformly distributed (s.t. $\mathbb{E}(Q(j)) = 0$ and $\mathbb{E}(Q(j)^2) = 1/12$) and independent of $U(j)$ (s.t. $\mathbb{E}(U(j)Q(j)) = \mathbb{E}(U(j))\mathbb{E}(Q(j)) = 0$). Under these assumptions, Eq. (14) simplifies in expectation to:

$$\mathbb{E}(D^2) = \mathbb{E}\left(\sum_{j=1}^{n} U(j)^2 + Q(j)^2 + 2U(j)Q(j)\right) = \rho^2 + \frac{n}{12}. \tag{15}$$

This shows that rounding increases the distortion on expectation.

Yet, this simple analysis is wrong outside the high resolution regime, and we need to be more careful. The expectation of a sum is always the sum of the expectations, whatever the dependence between the summands: $\mathbb{E}(D^2) = \sum_{j=1}^{n} \mathbb{E}(E(j)^2) = n\mathbb{E}(E(j)^2)$ with

$$\mathbb{E}(E(j)^2) = \sum_{\ell=0}^{255} \ell^2 \mathbb{P}(|E_j| = \ell). \tag{16}$$

We need the distribution of $E(j)$ to compute the expected distortion after rounding. This random variable $E(j)$ takes a value depending on the scalar product $U(j) := \mathbf{U}^\top \mathbf{c}_j$, where $\mathbf{c}_j$ is the j-th canonical vector. This scalar product lies in $[-\rho, \rho]$, so that $\mathbb{P}(E(j) \geq \ell) = 0$ if $\rho < \ell - 1/2$. Otherwise, $|E(j)| \geq \ell$ when $|U(j)| \geq \ell - 1/2$, which happens when $\mathbf{U}$ lies inside the dual hypercone of axis $\mathbf{c}_j$ and semi-angle $\theta(\ell) = \arccos(c(\ell))$ with $c(\ell) := (2\ell - 1)/2\rho$ as shown in Fig. 1.

Since $\mathbf{U}$ is uniformly distributed over an hypersphere, the probability of the event $\{|U(j)| \geq \ell - 1/2\}$ is equal to the ratio of the solid angles of this dual hypercone and the full space $\mathbb{R}^n$. This quantity can be expressed via the incomplete regularized beta function I. In

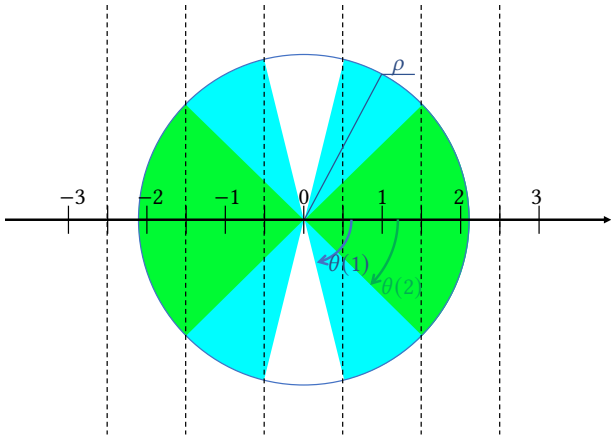

Figure 1: The dual hypercones related to $\ell = 1$ and 2. Since $\rho < 3 - 1/2$, the other hypercones do not exist.

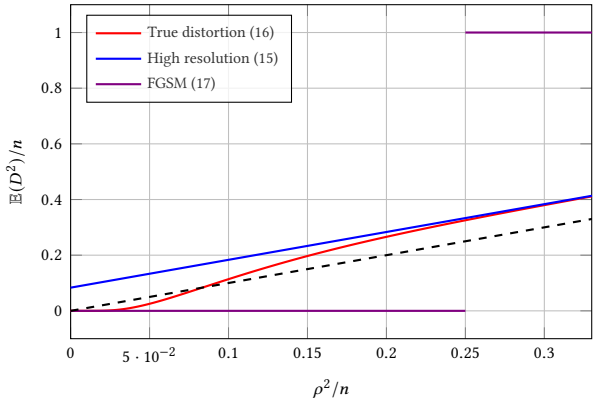

Figure 2: Expected power $\mathbb{E}(D^2)/n$ after rounding as a function of the perturbation power ρ^2/n before rounding.

the end, $\forall \ell \in \{0, \ldots, L-1\}$,

$$\mathbb{P}(|E(j)| \geq \ell) = \begin{cases} 1, & \text{if } \ell = 0 \\ 1 - I_{c(\ell)^2}(1/2, (n-1)/2), & \text{if } 0 \leq c(\ell) \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

Note that it approximately equals $2\Phi(\sqrt{n}c(\ell)/2)$ for large n, where Φ is the cumulative distribution function.

Computing $\mathbb{E}(D^2)$ is now possible with the simple trick for discrete r.v.: $\mathbb{P}(|E(j)| = \ell) = \mathbb{P}(|E(j)| \geq \ell) - \mathbb{P}(|E(j)| \geq \ell + 1)$. This expected distortion after rounding depends on n and ρ, the norm of the perturbation before rounding. Figure 2 shows that rounding reduces the distortion outside the high resolution regime. Indeed, the distortion after quantization is smaller than before quantization when $\rho^2/n < 0.08$ for $n = 3 * 299^2$ (*i.e.* the dimension of images from ImageNet). When the adversarial perturbation has a smaller norm ρ, rounding is likely to kill it and $\mathsf{R}(\mathbf{x}_a) = \mathbf{x}_o$, or at least to drastically decrease its amplitude. Since known attacks achieve rather low distortion, we are not in the high resolution regime and the rounding likely pulls down the attack.

This is a statistical study working on expectation. This phenomenon happens systematically on some attacks when the perturbation on each pixel value has a small amplitude. For instance with FGSM (6), we have $u(j) = \pm\epsilon$, $\forall j \in [n]$ which gives $\rho = \epsilon\sqrt{n}$. Then, each pixel perturbation is rounded to $e(j) = \pm\mathsf{R}(\epsilon)$ so that

$$D^2 = n\mathsf{R}^2\left(\frac{\rho}{\sqrt{n}}\right). \tag{17}$$

As shown in Fig. 2, rounding systematically cancels FGSM for $\epsilon < 1/2$, whereas it amplifies it a lot for $1/2 < \epsilon < 1$.

4 OUR APPROACH

Our approach considers quantization as a post-processing independent of the attack. The quantization does not interfere with the attack, which is a state-of-art implementation taken off-the-shelf as a black-box.

From an original image $\mathbf{x}_o \in \mathcal{X}$, this attack has produced a sample $\mathbf{x}_a$ which does not a priori belong to $\mathcal{X}$. This sample may or may not be adversarial depending whether the attack succeeded

or not. We aim at finding the best quantization process producing $\mathbf{x}_q = \mathsf{Q}(\mathbf{x}_a) \in \mathcal{X}$ adversarial with high probability.

We make the following assumptions. Since we are in a white-box setting, the quantization mechanism has access to:

- the original image $\mathbf{x}_o$,
- the sample $\mathbf{x}_a = \mathbf{x}_o + \mathbf{u}$ produced by the attack,
- the prediction function of the classifier,
- the gradient of this function.

Our approach constrains the quantization to consider only two options per pixel [1]: $\forall j \in [n], x_q(j) \in \{\lceil x_a(j) \rceil, \lfloor x_a(j) \rfloor\}$. This can be rewritten as $x_q(j) \in \mathbf{x}_o(j) + \{\lceil u(j) \rceil, \lfloor u(j) \rfloor\}$. Note that if $u(j) \in \mathbb{Z}$, $\lceil u(j) \rceil = \lfloor u(j) \rfloor = u(j)$. Assuming this special case rarely occurs, this leaves almost 2^n possibilities in total.

As in Sect. 3, the quantization is modelled by the addition of a noise $\mathbf{q}$ as follows

$$\mathbf{x}_q = \mathbf{x}_a + \mathbf{q} = \mathbf{x}_o + \mathbf{u} + \mathbf{q}. \tag{18}$$

For each pixel, we see that $q(j)$ takes the value $\lceil u(j) \rceil - u(j) \geq 0$ or $\lfloor u(j) \rfloor - u(j) \leq 0$.

4.1 Distortion based quantization

We define the function D of $\mathbf{q}$ as the final distortion after quantization w.r.t. $\mathbf{x}_o$:

$$\mathsf{D}(\mathbf{q}) := \|\mathbf{x}_q - \mathbf{x}_o\|^2 = \|\mathbf{u} + \mathbf{q}\|^2. \tag{19}$$

We define Q_0 the quantization that minimises the distortion. For the j-th pixel, it makes $q(j) + u(j) = \lceil u(j) \rceil$ if $\lceil u(j) \rceil^2 < \lfloor u(j) \rfloor^2$, *i.e.* if $u(j) < 0$.

$$\mathsf{Q}_0(x_a(j)) := x_o(j) + \begin{cases} \lceil u(j) \rceil & \text{if } u(j) \leq 0, \\ \lfloor u(j) \rfloor & \text{if } u(j) > 0. \end{cases} \tag{20}$$

Note that the distortion is lower after this quantization: $\|\mathbf{x}_a - \mathbf{x}_o\|^2 \geq \|\mathsf{Q}_0(\mathbf{x}_a) - \mathbf{x}_o\|^2$. However, we don't have any guarantee that $\mathsf{Q}_0(\mathbf{x}_a)$ is adversarial. For instance, if $|u(j)| < 1$, $\forall j \in [n]$, then either $\lceil u(j) \rceil$ or $\lfloor u(j) \rfloor$ equals 0, and $\mathsf{Q}_0(\mathbf{x}_a) = \mathbf{x}_o$, which is not adversarial.

4.2 Gradient based quantization

Another option is to quantize in order to strengthen the adversariality of the image. We define a new classifier loss as follows:

$$\mathsf{L}_Q(\mathbf{q}) := p_q(t(\mathbf{x}_o)) - p_q(\kappa_a), \tag{21}$$

$$\mathbf{p}_q := \mathsf{f}(\mathsf{a}(\mathbf{x}_a + \mathbf{q})), \tag{22}$$

$$\kappa_a := \arg \max_{k \neq t(\mathbf{x}_o)} p_a(k). \tag{23}$$

In words, the loss is the difference between the predicted probabilities that $\mathbf{x}_q$ belongs to the true class of $\mathbf{x}_o$ minus the one of a given class $\kappa_a \in [C]$. That class κ_a is indeed the class region where the attack tried to drive sample $\mathbf{x}_a$ to, with or without success. Our quantization works with this loss function whatever the loss L_A used by the attack before.

We define quantization Q_∞ which aims at getting[2] $\mathsf{L}_Q(\mathbf{q}) < 0$, indicating that $\mathbf{x}_q$ is adversarial. Since $\mathbf{q}$ is a small quantization noise, we approximate this loss to the first order:

$$\mathsf{L}_Q(\mathbf{q}) \approx \mathsf{L}_Q(\mathbf{0}) + \mathbf{q}^\top \mathbf{g}, \tag{24}$$

where $\mathsf{L}_Q(\mathbf{0})$ is the classifier loss computed at $\mathbf{q} = 0$ (*i.e.* when $\mathbf{x}_q = \mathbf{x}_a$, it is lower than 0 if the previous attack succeeded), and $\mathbf{g} := \nabla \mathsf{L}_Q(\mathbf{q})\big|_{\mathbf{0}}$ is its gradient computed at $\mathbf{q} = 0$.

Quantization Q_∞ minimizes the classifier loss through its approximation which is a correlation over the pixels. Therefore, it makes the signs of $q(j)$ and $g(j)$ opposite:

$$\mathsf{Q}_\infty(x_a(j)) := x_o(j) + \begin{cases} \lceil u(j) \rceil & \text{if } g(j) < 0, \\ \mathsf{Q}_0(x_a(j)) & \text{if } g(j) = 0, \\ \lfloor u(j) \rfloor & \text{if } g(j) > 0. \end{cases} \tag{25}$$

Since the quantization is not impacting the approximation loss when $g(j) = 0$, we choose the option that minimizes the distortion.

4.3 Our approach: Lagrangian quantization

In our approach, the quantization Q_λ minimizes a linear combination of the distortion and the classifier loss: For $\lambda \geq 0$:

$$\mathsf{Q}_\lambda(\mathbf{x}_a) := \mathbf{x}_o + \mathbf{u} + \arg\min_{\mathbf{q}} \mathsf{D}(\mathbf{q}) + \lambda \mathsf{L}_Q(\mathbf{q}), \tag{26}$$

under the constraint that $q(j) \in \{\lceil u(j) \rceil, \lfloor u(j) \rfloor\} - u(j)$, $\forall j \in [n]$. Thanks to the first order approximation (24) of the classifier loss, the functional to be minimized in (26) becomes a sum over all pixels. The optimization problem can be solved by considering the quantization of each pixel independently, $\forall j \in [n]$:

$$q(j) = \arg \min_{q \in \{\lceil u(j) \rceil, \lfloor u(j) \rfloor\} - u(j)} (u(j) + q)^2 + \lambda g(j) q. \tag{27}$$

Consequently the complexity of the quantization breaks down from $O(2^n)$ to $O(n)$ by solving n trivial optimization problems: When $u(j) \notin \mathbb{Z}$ then $\lfloor u(j) \rfloor = \lceil u(j) \rceil - 1$, and the solution is found as:

$$\mathsf{Q}_\lambda(x_a(j)) = x_o(j) + \begin{cases} \lceil u(j) \rceil & \text{if } 1 - 2\lceil u(j) \rceil \geq \lambda g(j) \\ \lfloor u(j) \rfloor & \text{otherwise.} \end{cases} \tag{28}$$

Note that we find back the previous rule (20) when $\lambda = 0$ because $(1 - 2\lceil u(j) \rceil) > 0$ if and only if $u(j) \leq 0$. In the same way, Q_λ converges to mechanism Q_∞ (25) because only the sign of $g(j)$ matters when $\lambda \to +\infty$.

Figure 3 illustrates the three quantization schemes in the domain $(g(j), \lceil u(j) \rceil)$. Note that there are pixels which are always quantized in the same way whatever the value of $\lambda \geq 0$. This is the case when one quantization value minimizes both the distortion and the classifier loss:

- for all indices where $g(j) < 0$ and $\lceil u(j) \rceil < 1/2$, $q(j)$ is always quantized to $\lceil u(j) \rceil - u(j)$,
- for all indices where $g(j) > 0$ and $\lceil u(j) \rceil > 1/2$, $q(j)$ is always quantized to $\lfloor u(j) \rfloor - u(j)$.

We denote by $\mathcal{J} \subset [n]$ the (complementary) subset of indices whose quantization depends on λ. It is defined as:

$$\mathcal{J} := \{j \in [n] : g(j) \neq 0,\ \mathsf{sign}(g(j)) \neq \mathsf{sign}(\lceil u(j) \rceil - 1/2)\}. \tag{29}$$

We assume that this subset is not empty. Sect. 5.2 empirically shows that $\mathcal{J}$ gathers around three fourths of the pixels.

[1] $\lceil x \rceil$ is defined as the unique integer s.t. $\lceil x \rceil - 1 < x \leq \lceil x \rceil$. $\lfloor x \rfloor$ is defined as the unique integer s.t. $\lfloor x \rfloor \leq x < \lfloor x \rfloor + 1$.

[2] A margin $m \leq 0$ can be also be enforced with $\mathsf{L}_Q(\mathbf{q}) < m < 0$.

4.4 Choice of Lagrange multiplier λ

We now look for the best value of $\lambda > 0$. When λ increases, the quantization trades the distortion against the classifier loss. The distortion is the lowest for $\lambda = 0$ and increases, whereas the classifier loss (at least its approximation (24)) is a decreasing function of λ. Therefore, if Q_∞ fails forging an adversarial image, so does Q_λ whatever the value of λ. Otherwise, it is worth looking for the best value $\lambda^\star$ giving the smallest distortion while succeeding to delude the classifier.

For all indices in $\mathcal{J}$, we define the following ratio:

$$r(j) := \frac{1 - 2\lceil u(j) \rceil}{g(j)} > 0, \tag{30}$$

so that the quantization rule (28) becomes $\mathrm{Q}_\lambda(x_a(j)) = x_o(j) + \lceil u(j) \rceil$ if $r(j) > \lambda$ and $\mathrm{Q}_\lambda(x_a(j)) = x_o(j) + \lfloor u(j) \rfloor$ otherwise. An index j in $\mathcal{J}$ sees $q(j)$ moving from one value to another when λ has increased up to the value $r(j)$. Note that it is useless to explore $\lambda \in (\max_j r(j), +\infty)$ in the sense that values of λ in this interval gives birth to the same results as Q_∞.

Therefore, we carry on a search of the minimal value of $\lambda^\star \in [0, \max_j r(j)]$ giving an adversarial image, *i.e.* $\mathsf{L}_Q(\mathbf{q}) \leq 0$. To do this, we rank the ratios $(r(j))_{j \in \mathcal{J}}$ by increasing order. Pixels ranked first offer a better trade-off: they yield a valuable loss decrease for a modest distortion increase. We perform a binary search on this sorted set, so that $\lambda^\star = r(j^\star)$ the smallest ratio giving an adversarial image. This has a complexity in $O(\log n)$ since $|\mathcal{J}| \leq n$.

4.5 Calls to the network

Solving problem (26) for a given λ a priori needs $O(2^n)$ calls to the network. By replacing the loss by its linear approximation (24), the complexity is reduced to $O(1)$ calls to get the gradient $\mathbf{g} := \nabla \mathsf{L}_Q(\mathbf{q})\big|_{\mathbf{0}}$. The gradient is computed by backpropagation and thanks to auto-differentiation, its complexity is roughly twice the complexity of one forward pass in the network.

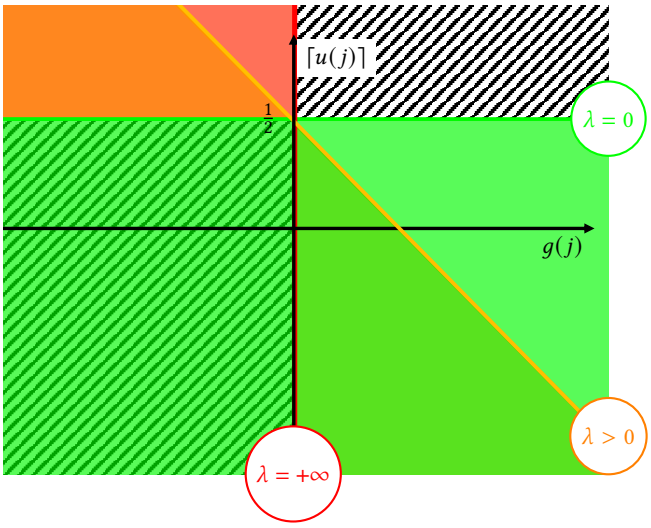

Figure 3: Quantizing with Q_λ in the domain $(g(j), \lceil u(j) \rceil)$. The colored regions show when $q(j)$ is quantized to $\lceil u(j) \rceil - u(j)$, *i.e.* $\mathrm{Q}_\lambda(x_a(j)) = x_o(j) + \lceil u(j) \rceil$, depending on λ. In the complementary half-plane, $q(j) = \lfloor u(j) \rfloor - u(j)$ and $\mathrm{Q}_\lambda(x_a(j)) = x_o(j) + \lfloor u(j) \rfloor$. In the hashed areas, quantization is always the same, independently of λ.

The above-mentioned binary search over λ can also resort to the approximated classifier loss. This avoids any call to the network. Nevertheless, this first order approximation is sometimes not accurate (see Fig. 4). An idea is to compensate this by a margin: The binary search ends with a value of λ for which the approximated loss (24) is below that margin. Setting the value of that margin s.t. the real loss is below zero with high probability is however difficult.

Another option is that the binary search uses the true classification loss (21) by calling the network to check whether $\mathsf{L}_Q(\mathbf{q}) \leq 0$. The complexity of the search now dominates the cost of the quantization: It scales as $O(\log n)$ calls to the forward pass of the network (since $|\mathcal{J}| \leq n$). In our simulation over ImageNet, the search ends within at most 18 calls. To summarize, the first order approximation (24) is used to quantize pixels (28) and to rank the pixels according to ratio (30), but the search of λ uses the true classification loss (21).

5 EXPERIMENTAL WORK

This section presents experimental investigations about the impact of the quantization, and then a benchmark of several attacks. We first present the experimental protocol.

5.1 Experimental Protocol

5.1.1 Dataset and Networks. Our experiments are based on the dataset of images used for the NeurIPS competition [7]. This is indeed a subsection of 1,000 images from ImageNet. We test several versions of the ResNet neural network [5]: the basic ResNet-18, the deeper ResNet-50, and ResNet-50R, its version robustified by adversarial retraining with PGD_2 [9].

Table 1 shows that ResNet-50 is more powerful than ResNet-18 enjoying a better accuracy with a higher confidence. On the contrary, adversarial retraining has notably spoiled the accuracy of this network.

5.1.2 Evaluation procedure. Comparing attacks is difficult because they have different purposes. FGSM (6) and PGD_2 (8) are constrained on the distortion: the main parameter is the allocated distortion budget. In the literature, these attacks are gauged by measuring the probability of success for a given distortion budget. In contrast, CW (9) and DDN (10) are forging an adversarial sample

Table 1: Accuracy and confidence of the image classifiers measured on the NeurIPS competition [7] dataset. Confidence is gauged as the mean of the estimated probability of the ground truth class as provided in the dataset, knowing that the prediction is correct.

	Accuracy	Confidence
ResNet-18 [5]	84.1%	0.79
ResNet-50 [5]	92.7%	0.88
ResNet-50R [9]	69.1%	0.60

almost surely (if the total number of iterations is large enough). These attacks are usually gauged by the average distortion.

For a fair comparison, we adopt the methodology of [17] comparing operating characteristics. This characteristic is the function relating a distortion measure $\bar{d}$ to the probability of success P_{suc}. Here, we measure the square root of the perturbation power (which is also the standard deviation in the pixel domain):

$$\mathrm{d}(\mathbf{x}_q, \mathbf{x}_o) := \|\mathbf{x}_q - \mathbf{x}_o\|/\sqrt{n}. \tag{31}$$

For example, FGSM gives $\mathrm{d}(\mathbf{x}_q, \mathbf{x}_o) = \epsilon$ since all pixels are modified by $\pm\epsilon$.

For CW and DDN (with or without quantization), the attack is mounted over the M images $\{\mathbf{x}_{o,m}\}_{m=1}^{M}$ of the dataset. Images for which the attack failed are discarded. We measure the distortion for the adversarial images and construct the characteristic $\bar{d} \to P_{suc}(\bar{d})$ with:

$$P_{suc}(\bar{d}) := M^{-1}\left|\{m \,:\, \mathrm{d}(\mathbf{x}_{q,m}, \mathbf{x}_{o,m}) < \bar{d}\}\right|. \tag{32}$$

For FGSM, IFGSM, and PGD_2 (with or without quantization), several parameter values are tested. For each image, we record the lowest distortion obtained for a success. This is what the attacker would do in a white-box scenario. For instance, with FGSM, we run a line search over ϵ in (6) for each image in order to get the smallest possible value that succeeds in deluding the classifier. Then, the operating characteristic is constructed as mentioned above.

5.2 Illustration of our approach

This section gives some illustrations of our quantization mechanism $\mathrm{Q}_{\lambda^\star}$. Table 2 first offers some statistics counting how many pixels are quantized s.t. it induces a loss decrease. This concerns two populations of indices:

- indices in $[n]\backslash\mathcal{J}$. Their quantization does not depend on $\lambda^\star$ because it decreases both the distortion and the approximated loss (24). Their couple $(g(j), \lceil u(j)\rceil)$ lies in the hashed regions depicted in Fig. 3.This roughly corresponds to one fourth of the pixels (see Table 2).
- indices in $\mathcal{J}$ whose ratio (30) $r(j)$ is lower than $\lambda^\star$. $\mathrm{Q}_{\lambda^\star}$ quantizes these pixels because they offer a more interesting loss decrease at a rather small distortion increase.

The global percentage reflects the robustness of the classifier. A more robust classifier implies more pixels quantized to reduce the loss at the expense of more distortion. We clearly see that ResNet-18 is less robust than ResNet-50 less than ResNet-50R. This global percentage also reflects the power of the attack: $\mathrm{PGD}_2$is more powerful than FGSM.

Table 2: Percentage of quantization contributing to a loss decrease. The first number is the percentage of quantization decreasing both loss and distortion (see hashed regions in Fig. 3), the second number depends on the value of $\lambda^\star$.

Attacks	FGSM	PGD_2
ResNet-18	29.5 + 3.8 = 33.3%	26.1 + 4.5 = 30.6%
ResNet-50	34.0 + 4.8 = 38.8%	26.5 + 3.7 = 30.2%
ResNet-50R	27.8 + 13.9 = 41.7%	21.3 + 16.7 = 38.0%

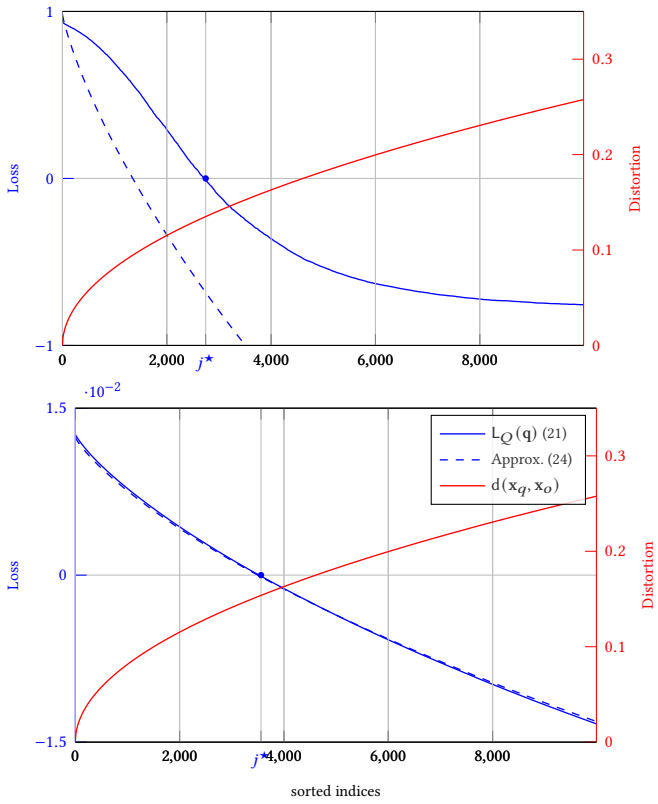

Figure 4: Trade-off between the classifier loss and the distortion as $\lambda = r(j)$ for increasing sorted index j. (Top) ResNet-50, (Bottom) ResNet-50R.

Figure 4 illustrates how we set the value of $\lambda^\star$ by showing how the classifier loss and the distortion evolves as λ increases for a given image. The x-axis represent the pixel indices of $\mathcal{J}$ once sorted by their ratios $\{r(j)\}$ in ascending order. Our approach finds the index $j^\star$ for which the loss cancels and which defines $\lambda^\star = r(j^\star)$. From the experiments we conducted, we have noticed that (24) is often a poor approximation of the true loss (21) of ResNet-18 and ResNet-50. This justifies the use of (21) in the line search for finding the value of $\lambda^\star$. The approximation (24) is then only used for ranking the pixels by the trade-off between distortion and classifier loss they individually provide. Yet, the approximation is much better for ResNet-50R. We suspect that this is due to the small norm gradient of the loss of this robust network.

5.3 Experimental investigations

In this section, the attacks are conducted with the "best efforts", in the sense that their complexity is not limited. The total number N of iterations is high, the step α is small, many ϵ values are tested (see Sect. 3.2). The goal is to forge for each image its adversarial counterpart offering the best quality with the purpose of revealing the intrinsic power each attack.

5.3.1 The nature of the quantization. We first study the impact of the rounding R (Sect. 3.3), the quantization Q_λ with $\lambda = 0$ (Sect. 4.1),

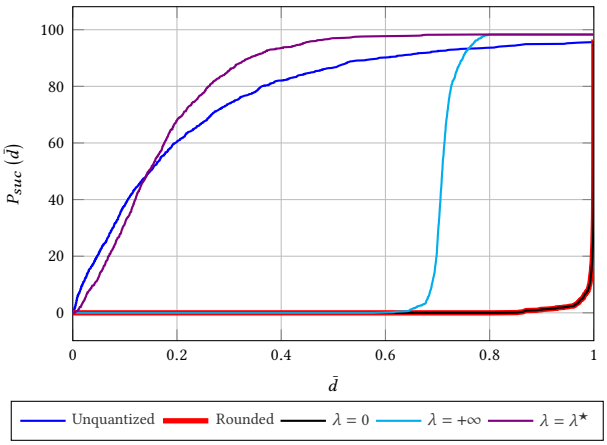

Figure 5: Operating characteristic of FGSM against classifier ResNet-18 for $\epsilon \in [0, 1]$.

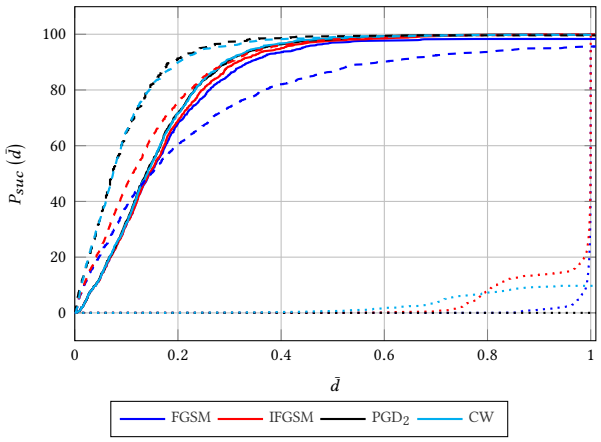

Figure 6: Operating characteristic for FGSM, IFGSM, PGD_2, and CW against ResNet-18. (dashed) without quantization, (dotted) with rounding, (plain) with our approach.

$\lambda = +\infty$ (Sect. 4.2), and the optimal strategy where the best value $\lambda^\star$ is found for each image (Sect. 4.4). The classifier is ResNet-18 and the attack is FGSM (6).

Figure 5 shows the operating characteristics when $\epsilon \in [0, 1]$. For $\epsilon < 1$, the pixel perturbation has an amplitude smaller than 1 and its quantization has the following two options: either $\{0, 1\}$ or $\{-1, 0\}$ depending on the sign of the perturbation. Therefore, for minimizing the distortion, $Q_0(\mathbf{x}_a) = \mathbf{x}_o$ systematically. For $\epsilon = 1$ exactly, the quantization has only one option (either $\{1\}$ or $\{-1\}$ depending on the sign of the perturbation). Then, $d(Q_0(\mathbf{x}_a), \mathbf{x}_o) = 1$ and it happens that almost all the images are adversarial. Therefore, the operating characteristic for Q_0 is almost an all-or-nothing function. $P_{suc}(\bar{d}) > 0$ for $\bar{d} \lesssim 1$ due to the border effect clipping the pixel values to the range $[0, 255]$.

The rounding to the nearest integer R has the same operating characteristic in Fig. 5 . For $\epsilon < 0.5$ the perturbation is rounded to 0 for all pixels. For $\epsilon > 0.5$ the perturbation is rounded to ± 1 and the result is the same as for $\epsilon = 1$ with Q_0.

The operating characteristic for Q_∞ is more interesting. Suppose that for the j-th pixel, $0 < u(j) = \epsilon < 1$. According to (6), this is due to the fact that the gradient of the loss at $\mathbf{x}_o$ has a negative component at index j. According to (28), Q_∞ quantizes this perturbation back to 0 if $g(j) > -1/\lambda$. Yet, that $g(j)$ is the j-th component of the gradient computed at $\mathbf{x}_a$. Hence for λ large enough, when these two gradients do not agree on the sign of the same component, that perturbation pixel is quantized to 0. We would expect this event to be seldom. However, since the operating characteristic is peaky around $\bar{d} = 0.7 \approx 1/\sqrt{2}$ and that all the other perturbation pixels are quantized to ± 1, it means that 50% of the pixels are quantized back to the original value.

From theses different results we can see that our approach $Q_{\lambda^\star}$ provides a huge improvement. Indeed, its operating characteristic is as good as the one of the unquantized FGSM. Quantization is especially more effective at middle range distortion: For $\bar{d} = 0.4$, the success rate is higher by 10 points. Its operating characteristic also converges to a higher level.

5.3.2 With or without quantization. Figure 6 compares the operating characteristics for the attacks FGSM, IFGSM, PGD_2, and CW with and without quantization (by rounding or by our approach) against ResNet-18.

Without quantization, we find back the hierarchy well-known in the literature: CW is better than PGD_2 better than IFGSM better than FGSM. Note that the difference between CW and PGD_2 is not tangible in this Fig. 6 (dashed plots) because we use a super-fine step for PGD_2. Rounding completely spoils these attacks because the distortion before quantizing is too low (dotted plots). This illustrates the theoretical study of Sect. 3.3. Our approach almost preserves the operating characteristics without rounding (plain curves). Indeed, our quantization mechanism improves FGSM but slightly degrades the other attacks. The hierarchy is preserved but the differences are now tiny.

5.3.3 Natural vs. robust network. Figure 7 shows operating characteristics against a deeper network (ResNet-50) and its robust version (ResNet-50R) fine-tuned by adversarial retraining with PGD_2.

Attacking a deeper network does not change much the performances except for the rudimentary FGSM: ϵ must be bigger than 1 to reach 90% in success rate. Note that our approach fixes this as $P_{suc}(\bar{d}) > 90\%$ for $\bar{d} = 0.5$. As for the other attacks, our approach preserves the operating characteristics.

Attacking the robust network is another story. For a given level of P_{suc}, the necessary distortion is much bigger. Note that the range of the x-axis in Fig. 7 is not the same as the one of Fig. 6. CW and PGD_2 are attacks more powerful than FGSM and IFGSM.

For FGSM, our approach produces an operating characteristic with 'leaps'. One leap corresponds to a range $(k, k+1]$ with $k \in \mathbb{N}$ for the parameter ϵ. This is due to the fact that our approach is constrained: For any value inside that range, it quantizes $|u(j)|$ to $\{k, k+1\}$. Indeed, for $\epsilon \in \mathbb{N}$, the quantization introduces no changes: $Q_\lambda(\mathbf{x}_a) = \mathbf{x}_a$. This is where the operating characteristic touches back the one without quantization. As for the other attacks, our approach improves or at least preserves the operating characteristic without quantization.

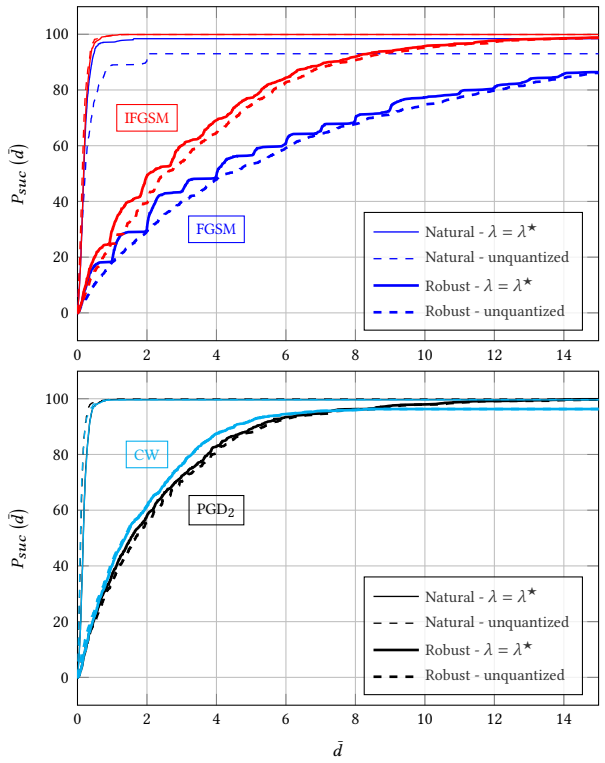

Figure 7: Operating characteristics against a natural network (ResNet-50) or a robust network (ResNet-50R) with and without quantization for the attacks (Top) FGSM and IFGSM, (Bottom) PGD_2 and CW.

Figure 8 displays some of the worst case examples with visible distortion against ResNet-50R and their equivalent on ResNet-50. For each image, the original is shown in the first column. For both network, the image on the left corresponds to the rounding that occurred when saving an adversarial sample $\mathbf{x}_a$ forged by PGD_2 in the 'png' format. This rounded image is no longer adversarial except for the zebra on ResNet-50R. The image on the right shows the result when quantizing PGD_2 with our approach.

When quantized with our method, the four images remain adversarial. On ResNet-50 our method increases the distortion in order to remain adversarial. On ResNet-50R our method actually slightly decreases the distortion and returns an adversarial image.

Although distortions are different in between the two images, they remain visually similar. On ResNet-50 the perturbation is imperceptible while it is very much is on ResNet-50R. This illustrates how the adversarial training defends the network against an attack.

5.4 Benchmark

This section now compares the attacks with our quantization to DDN (10), one of the rare attack 'natively' producing quantized digital images in the recent literature For a fair comparison w.r.t. complexity, the attacks are mounted against the robust network ResNet-50R but with a limited complexity (contrary to the previous study): they all compute 100 gradients. In other words, the total number of iterations is $N = 100$.

The setup is the following:

- FGSM (6). $N = 100 \times 1$: We test a hundred values ranging in $0.15 * \{1, 2, \ldots, 100\}$ for parameter ϵ.
- IFGSM (7). $N = 5 \times 20$: Parameter ϵ is set to $\bar{d}_{\max}/N$ with $N = 20$ and $\bar{d}_{\max}$ is given by a 5-step binary search over $[0, 15]$.
- PGD_2 (8). $N = 2 \times 5 \times 10$. For ResNet-50, $\alpha \in \{1, 5\}$. For ResNet-50R, $\alpha \in \{50, 100\}$. A 5-step binary search finds the correct ϵ to produce an adversarial example after 10 iterations.
- CW (9). $N = 5 \times 20$: 5 iterations for the outer loop, 20 for the ADAM inner loop, and a margin $m = 0$. The learning rate is 0.005 (resp. 0.01) and μ is initialized to 1000 (resp. 5000) for ResNet-50 (resp. ResNet-50R).
- DDN (10). $N = 2 \times 50$: For ResNet-50, $\alpha \in \{100, 500\}$. For ResNet-50R, $\alpha \in \{1000, 5000\}$. Each value of α is tested with $N = 50$ iterations and $\gamma = 0.05$.

Fig. 9 shows that CW and PGD_2 are now on par when under limited complexity (CW is slightly better at low distortion but PGD_2 is better when $\bar{d} \geq 4$ against ResNet-50R). DDN is not performing well as it is worse than CW, PGD_2, and even IFGSM for ResNet-50. The next section investigates on this difficulty.

6 INTEGRATION INSIDE AN ITERATIVE ATTACK

So far, our quantization mechanism is decoupled from the attack forging $\mathbf{x}_a$. This is in strong contrast with the quantized version of DDN in [11] where a rounding concludes each iteration. This section proposes a proof of concept on how to integrate our quantization mechanism inside an iterative attack like DDN.

The main message is that this integration must follow the spirit of the attack. In DDN, the iteration (10) is driven by a distortion budget parametrized by $\rho^{(i)}$. This budget is adaptively scheduled over the iterations in the following way: it is increased if the sample $\mathbf{x}_a^{(i-1)}$ is not yet adversarial, it is decreased if that sample is already adversarial. However, the rounding concluding the iteration may spoil this fine-tuning of the distortion as seen in Sect. 3.3.

Our approach is able to better handle this distortion scheduling. It amounts to change the setting of the Lagrangian parameter in (26). We will define it by $\delta^\star$, not to confuse with $\lambda^\star$. As illustrated in Fig. 4, the distortion from $\mathbf{x}_o$ is an increasing function of λ. Therefore, we set $\delta^\star$ as the value of λ which gives the scheduled distortion.

Once the pixels of $\mathcal{J}$ (29) are ranked according to their ratios $\{r(j)\}$ defined in (30), the line search finds the first (resp. last) index $k^\star$ s.t. $\delta^\star = r(k^\star)$ produces a distortion bigger (resp. smaller) than the targeted budget when the previous sample $\mathbf{x}_a^{(i-1)}$ is not yet (resp. is already) adversarial. This defines the quantization $Q_{\delta^\star}$ which minimizes the approximated classifier loss while fulfilling the scheduled distortion.

This has to be done for each iteration. However, since the stopping condition is defined via the distortion, the line search no longer calls the classification network. Usually, the complexity of an attack

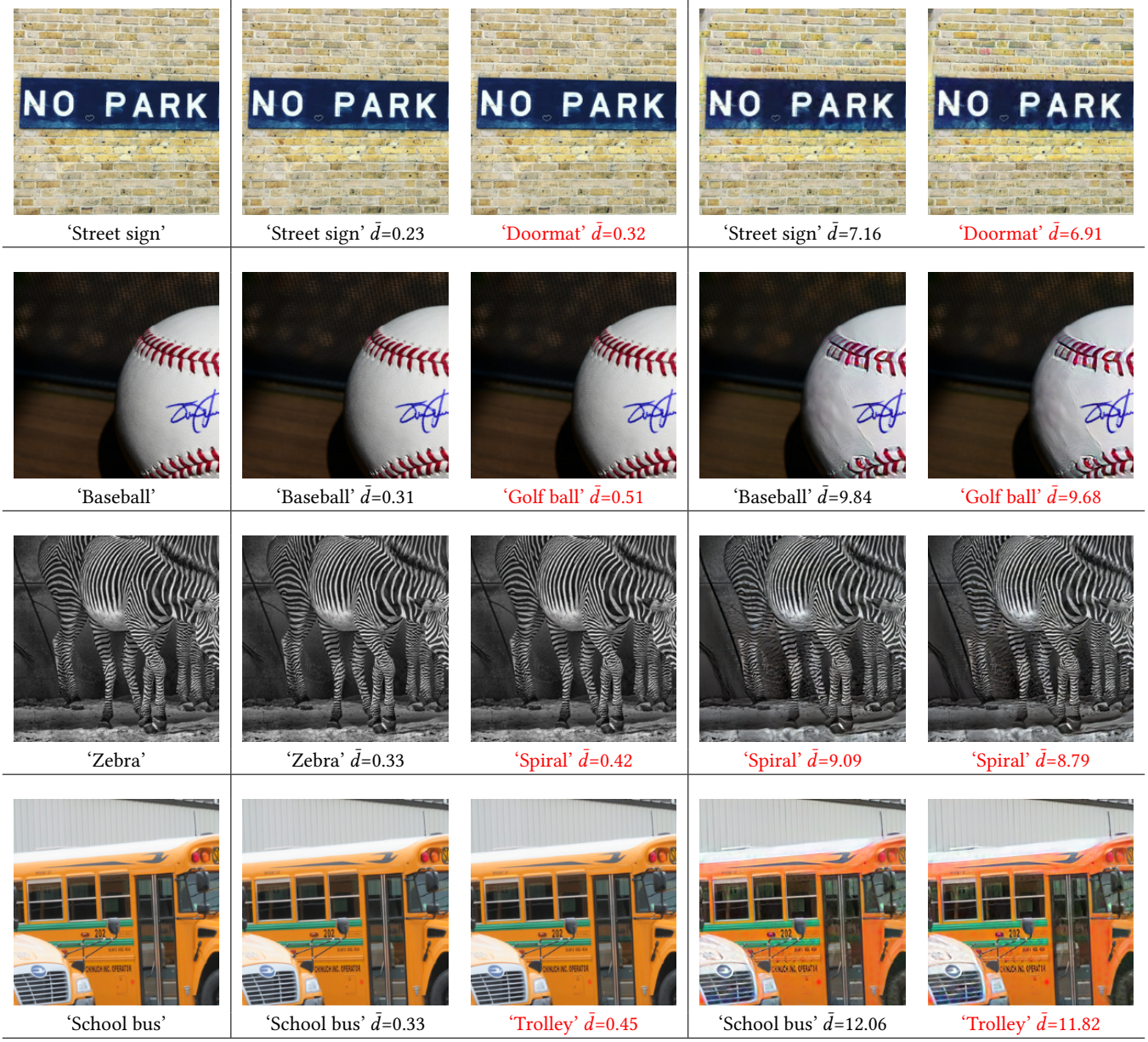

Figure 8: Examples of adversarial images against natural ResNet-50 and its robust version ResNet-50R. They are created with PGD_2 followed by a rounding (2nd and 4th columns) or our approach (3rd and 5th columns).

is measured by the number of calls to the classifier. Our integration inside DDN does not spoil its low complexity as it only consumes one gradient computation. This gradient information is used to compute ratios $\{r(j)\}$ defined in (30).

Fig. 10 shows operating characteristics of three versions of DDN:

- The original version of the quantized DDN which concludes each iteration by a rounding R,
- The original version of the unquantized DDN followed at the end by our quantization $Q_{\lambda^\star}$,
- Our integrated version of DDN which concludes each iteration by quantization $Q_{\delta^\star}$.

This comparison shows that the last two variants are better than the original scheme. This outlines that the quantification mechanism is of utmost importance. The last integrated version is the best. This

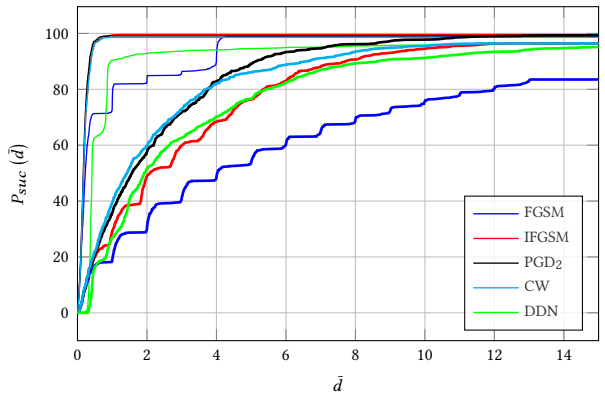

Figure 9: Benchmark of the attacks against natural (ResNet-50 - thin lines) and robust (ResNet-50R - thick lines) networks with limited complexity and quantization.

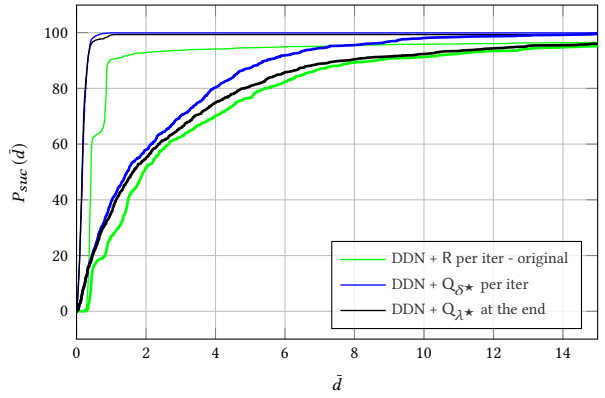

Figure 10: Operating characteristics of three DDN implementations against natural (ResNet50 - thin lines) and robust (ResNet-50R - thick lines) networks with limited complexity.

tends to prove that quantizing at each iteration is better than quantizing only at the end provided that the quantization mechanism is appropriate. This is not surprising: by integrating the quantization inside the iterative process, we allows the upcoming iterations to compensate for the drift due to the quantization.

7 CONCLUSION

This paper proposes a new quantification mechanism of adversarial samples. It has two main features: i) It is a post-processing independent of the attack, ii) Its complexity adds an extra cost of $O(\log n)$ calls to the network which is small compared to the complexity of the attack. Another point is that this mechanism can also be integrated inside an iterative attack like DDN.

Overall, thanks to our quantification, the integral constraint no longer spoil the operating characteristic of the attacks against natural and robust classifiers. The main difference is that the attacks CW, PGD$_2$, and DDN are more or less equally efficient under this constraint.

Figure 8 shows that the perturbation is clearly visible on some adversarial images against robust classifier. Our future work aims at taking into account a distortion metric better reflecting human perceptibility than the Euclidean distance.

8 ACKNOWLEDMENTS

This work has been funded in part by the French National Research Agency (ANR-18-ASTR-0009), ALASKA project: https://alaska.utt.fr, by the French ANR DEFALS program (ANR-16- DEFA-0003) and by the ANR chaire IAD SAIDA.

REFERENCES

[1] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *IEEE Symp. on Security and Privacy*.

[2] Yinpeng Dong, Hang Su, Baoyuan Wu, Zhifeng Li, Wei Liu, Tong Zhang, and Jun Zhu. 2019. Efficient Decision-Based Black-Box Adversarial Attacks on Face Recognition. In *The IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[3] A. Gersho and R.M. Gray. 1991. *Vector Quantization and Signal Compression*. Springer US.

[4] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. http://arxiv.org/abs/1412.6572

[5] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778. https://doi.org/10.1109/CVPR.2016.90

[6] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. In *ICLR*.

[7] Alexey Kurakin, Ian Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, Alan Yuille, Sangxia Huang, Yao Zhao, Yuzhe Zhao, Zhonglin Han, Junjiajia Long, Yerkebulan Berdibekov, Takuya Akiba, Seiya Tokui, and Motoki Abe. 2018. Adversarial Attacks and Defences Competition. arXiv:1804.00097 [cs.CV]

[8] Jie Li, Rongrong Ji, Hong Liu, Xiaopeng Hong, Yue Gao, and Qi Tian. 2019. Universal Perturbation Attack Against Image Retrieval. In *The IEEE International Conference on Computer Vision (ICCV)*.

[9] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. https://openreview.net/forum?id=rJzIBfZAb

[10] Anurag Ranjan, Joel Janai, Andreas Geiger, and Michael J. Black. 2019. Attacking Optical Flow. In *The IEEE International Conference on Computer Vision (ICCV)*.

[11] Jerome Rony, Luiz G. Hafemann, Luiz S. Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger. 2019. Decoupling Direction and Norm for Efficient Gradient-Based L2 Adversarial Attacks and Defenses. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[12] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations*. http://arxiv.org/abs/1312.6199

[13] Giorgos Tolias, Filip Radenovic, and Ondrej Chum. 2019. Targeted Mismatch Adversarial Attack: Query With a Flower to Retrieve the Tower. In *The IEEE International Conference on Computer Vision (ICCV)*.

[14] B Tondi. 2018. Pixel-domain adversarial examples against CNN-based manipulation detectors. *Electronics Letters* 54, 21 (2018), 1220–1222.

[15] Rey Reza Wiyatno and Anqi Xu. 2019. Physical Adversarial Textures That Fool Visual Object Tracking. In *The IEEE International Conf. on Computer Vision (ICCV)*.

[16] Yan Xu, Baoyuan Wu, Fumin Shen, Yanbo Fan, Yong Zhang, Heng Tao Shen, and Wei Liu. 2019. Exact Adversarial Attack to Image Captioning via Structured Output Learning With Latent Variables. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[17] Hanwei Zhang, Yannis Avrithis, Teddy Furon, and Laurent Amsaleg. 2019. Walking on the Edge: Fast, Low-Distortion Adversarial Examples. arXiv:1912.02153 [cs.CV]