Pixle: a fast and effective black-box attack based on rearranging pixels

Jary Pomponi, Simone Scardapane, and Aurelio Uncini

Department of Information Engineering, Electronics and Telecommunications (DIET) Sapienza University of Rome, Italy

Problem Formulation

Generating adversarial images that fool a neural network can be formalized as an optimization problem with constraints.
Let f be a classification function that takes as input an image sample, and produces a vector containing c containing the distribution of probability for each class.
Given an image x, and the associated class y, that is classified correctly by the function f, the goal of an attack is to change the prediction of the function, by producing an adversarial image $\overline{x}$ such that $f(\overline{x}) \neq y$. The image $\overline{x}$ must be similar to x, in such a way that the L_p norm of the difference is bounded by a constant ϵ:

$$\arg\max_i f_i(\overline{x}) \neq y \quad ||x - \overline{x}||_p \leq \epsilon \tag{1}$$

with $\epsilon \in \mathbb{N}_+$ in our black-box case. An attack can be untargeted, as described above, or targeted, by forcing the miss-classification of the adversarial image to a specific adversarial class $\overline{y} \neq y$: $f(\overline{x}) = \overline{y}$. The task of finding the perturbed image $\overline{x}$, associated to x, can be viewed as a minimization problem

$$\min_{\overline{x}} L(f(\overline{x}), y) \quad ||x - \overline{x}||_p \leq \epsilon\,, \tag{2}$$

whith L equals to $L(f(\overline{x}), y) = f_y(\overline{x})$ for untargeted attacks and $L(f(\overline{x}), \overline{y}) = 1 - f_{\overline{y}}(\overline{x})$ for targeted ones.

Pixle

Pixle is a black-box attack based on random search. Despite its simplicity, random search performs well in many situations and does not depend on gradient information associated to the objective function $L(\cdot, \cdot)$.
It attacks an image x, having class y, by moving the pixels in it. The algorithm performs R restarts, and T iterations per each restart. At each iteration, the algorithm randomly select a random source patch from the image and, for each pixel in it, it calculates a destination pixel using a function m. If the loss decreases, then the replacement is kept in the next iteration. At the end of each iteration the attack that reduced the loss more is permanently added to the adversarial image. Pixle is part of TorchAttacks framework (Kim 2020).

Pseudocode

Algorithm 1 The overall attacking procedure

Require: input image x with its associated label y. Maximum and minimum dimension for source patch. The number of restarts R and the iterations to perform for each restart step T. The mapping function m.

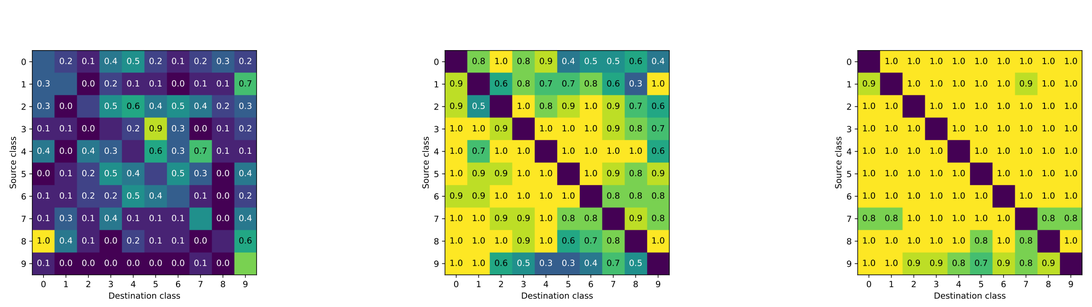
$\overline{x} \leftarrow x$
$l \leftarrow f_y(x)$
for $r = 0$ to R **do**
  $x^r \leftarrow \overline{x}$
  for $t = 0$ to T **do**
    Sample $p = (o_x, o_y, w_p, h_p)$.
    Calculate the set P
    $x^t \leftarrow \overline{x}$
    for $\forall (i, j) \in P$ **do**
      $(z, k) \leftarrow m(i, j)$
      $x^t_{z,k} \leftarrow x_{i,j}$
    end for
    if $L(f(x^t), y) < l$ **then**
      $l \leftarrow L(f(x^t), y)$
      $x^r \leftarrow x^t$
    end if
  end for
  $\overline{x} \leftarrow x^r$
end for
return $\overline{x}$

Results: success rate, iterations, and L_0 norm

Table 1: Results obtained when attacking multiple datasets trained on ResNet and VGG architectures. For each score, we show the mean and the variance, if present, calculated over all the images on that dataset. Best results for each pair dataset-architecture are highlighted in bold. We compared our proposal to OnePixle Su et al. 2019 and Scratch That Jere et al. 2019

Dataset	Model	Method	Success rate	Iterations	L_0 norm
CIFAR10	ResNet18	OnePixel	64.7	$5125_{\pm 799}$	5
		ScratchThat	99.7	1010	$38.3_{\pm 5.2}$
		Pixle (Proposed)	**100**	$119_{\pm 141}$	$26.8_{\pm 22.8}$
	VGG11	OnePixel	84.5	5100	5
		ScratchThat	99.3	1010	$27.64_{\pm 4.8}$
		Pixle (Proposed)	**100**	$80_{\pm 145}$	$20.1_{\pm 21.9}$
TinyImageNet	ResNet50	OnePixel	21.0	5100	5
		ScratchThat	69.7	1010	$49.1_{\pm 2.2}$
		Pixle (Proposed)	**99.6**	$310_{\pm 561}$	$59.0_{\pm 88.5}$
	VGG16	OnePixel	31.9	5100	5
		ScratchThat	76.6	1010	$48.6_{\pm 2.4}$
		Pixle (Proposed)	**100**	$87_{\pm 201}$	$21.5_{\pm 30.6}$
ImageNet	ResNet50	OnePixel	47.7	5100	5
		ScratchThat	82.6	$623_{\pm 321}$	$175.2_{\pm 25.7}$
		Pixle (Proposed)	**98.0**	$341_{\pm 426}$	$155.7_{\pm 184.2}$
	VGG16	OnePixel	31.9	5100	5
		ScratchThat	81.8	$753_{\pm 156}$	$143.0_{\pm 6.0}$
		Pixle (Proposed)	**99.0**	$519_{\pm 780}$	$98.5_{\pm 137.5}$

Results: targeted attack



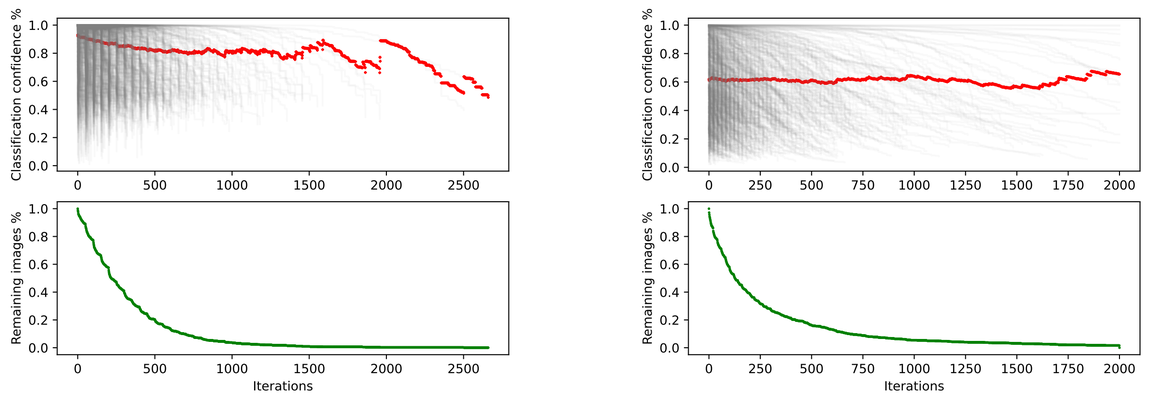
(a) The results obtained using OnePixel attack.

(b) The results obtained using Scratch That attack

(c) The results obtained using our proposal.

Figure 2: The images show the success rate on ResNet18 trained using CIFAR10. The results are calculated using 20 test images, classified correctly by the model, for each class. Each matrix contains the percentage of attacks that have been successfully completed.

Results: iterations and loss



(a) CIFAR10 results using VGG11.

(b) ImageNet results using ResNet50.

Figure 3: Each figure shows, on top, how the losses (the probability associated to the correct class) change during the iterations of our proposal, using the Restart-Iterative algorithm (the red dots are the average loss calculated on that iteration); while the bottom image shows how many images are left to attack after each iteration. Better viewed in colors.

References

Kim, H. (2020). "Torchattacks: A pytorch repository for adversarial attacks". In: *arXiv preprint arXiv:2010.01950*.

Jere, M., L. Rossi, B. Hitaj, G. Ciocarlie, G. Boracchi, and F. Koushanfar (2019). "Scratch that! An evolution-based adversarial attack against neural networks". In: *arXiv preprint arXiv:1912.02316*.

Su, J., D. V. Vargas, and K. Sakurai (2019). "One pixel attack for fooling deep neural