Pixel-domain adversarial examples against CNN-based manipulation detectors

B.Tondi

An attack method against CNN detectors, which minimizes the distortion in the pixel domain, is proposed. By focusing on CNN models developed for manipulation detection, our experiments show that, while the small perturbations introduced by existing methods tend to be cancelled out when the adversarial examples are rounded to pixels, thus making the attack ineffective, the proposed approach can generate pixel-domain adversarial images which succeed in inducing a wrong decision with very small distortions.

Introduction: Recent works have shown that Convolutional Neural Networks (CNNs) can provide a significant performance gain over model-based and standard machine-learning approaches also for multimedia forensic tasks, e.g., manipulation and forgery detection, and camera model identification. As a consequence, the vulnerability of deep learning to adversarial perturbations, highlighted by recent studies in the general context of pattern recognition [1], also affects CNN-based methods proposed in forensics [2]. Since in multimedia forensic applications the presence of a possibly informed attacker cannot be neglected [3], the analysis of such adversarial perturbations, and that of the possible countermeasures, is of primary importance.

As shown in the general literature of deep learning, the adversarial perturbations are often quasi-imperceptible, a very small perturbation of the inputs being sufficient to lead to an incorrect decision. This results in a poor robustness to image processing operations: in particular, the operation of rounding to integers is sometimes already sufficient to make an adversarial example ineffective. Passing to integer, however, is an essential step in practice, during forgery creation, to get an adversarial image that can be stored and transmitted.

In this letter we propose a gradient-inspired pixel domain attack against CNN detectors, which extends the one developed in [4] against SVMs. Differently from classical gradient-based attacks to CNN models, which perturb the image based on the gradient of the loss function w.r.t. the input, in the proposed attack, the gradient of the output score function is approximated with respect to the pixel values. The attack can be classifier as a *black box* attack since it does not need any knowledge of the model, but only that the network can be queried as an oracle and the output observed.

In the following, we first present the attack method, then we show its performance against state-of-the-art CNN-based manipulation detectors, distinguishing between pristine images (class 0) and processed images (class 1). The performance are compared to those achieved by existing methods for generating adversarial attacks, namely, the original box constrained L-BFGS by Szegedy et al. [1] (L-BFGS for short), the Fast Gradient Sign Method (FGSM) [5] and the Jacobian-based Saliency Map attack (JSMA) [6].

A gradient-inspired pixel domain attack to CNN models: Given an input image x, we let $z^l(x)$ denote the output score (logits) for class l, and $f^l(x)$ the value of $z^l(x)$ after the softmax layer, i.e., the softmax score. The objective of the attack is to find an image x^* which satisfies

$$\min_{x'} d(x, x') \quad s.t. \quad \arg\max_l f^l(x') \neq y, \tag{1}$$

where y is the ground truth label, or true class, $d()$ is a distortion measure, e.g. the L^1 distortion, and x' can only take integer values, in the range [0:255]. Then, the attacker searches for the minimal *feasible* perturbation such that the predicted class is wrong, where feasible means that, after the attack, the predicted image must be in the pixel domain. Solving (1) corresponds to search for the shortest path to the decision boundary. For binary classification problems, which is the case we focus on in this paper, this is equivalent to minimize $f^y(x')$ by means of the steepest descent method, until the predicted class is changed. Equivalently, by looking at the loss function $\mathcal{L}(f^y, y)$ (e.g., $\mathcal{L}(f^y, y) = -\log f^y$ in the case of cross-entropy loss) the attack has to maximize $\mathcal{L}(f^y, y)$ in order to solve (1). Note that such approach is effective also in the multi-class case, although suboptimum (in that case, we should maximize $f^l(x')$ for the class $l \neq y$ whose boundary is the closest to x).

To solve the optimization problem in (1), we extend to the case of CNN models the suboptimum iterative attack method proposed in [4] against binary SVM-based detectors. The approach presented in [4] works for grayscale (or one-channel) images only. In order to apply the attack to general networks trained on color images, we also generalize it to the case of 3-channels. The algorithm works as detailed in the following. At each iteration, first, it derives an approximation of the gradient of the output score function with respect to the pixel image x, then, the strength of the attack is adjusted by controlling the number of modified pixels. More specifically, with reference to the CNN softmax output f^y for the true class y, by letting σ be the increment applied to a pixel position, the approximated gradient in position (i, j, s) is given by

$$(\nabla f^y)_{ijs} = \frac{f^y(x + \delta_{ijs}) - f^y(x)}{\sigma}, \tag{2}$$

where δ_{ijs} is a 3-D matrix of the same size of x with all zeros except for the entry (i, j, s), where it takes the value σ. As in [4], we let $\sigma = 1$, which corresponds to the small possible increment on pixels. Obviously, due to the integer values constraint, the approximation of the real gradient can be very rough. The step size for the descent along the direction specified by ∇f^y, characterizing the strength of the attack, is determined by choosing the percentage k of to-be-modified pixels. If the decision boundary cannot be crossed, i.e., the predicted class cannot be changed, by modifying at most a prescribed fraction k_{max} of pixels, the modification is applied to the image with k_{max} and the process is iterated. Then, at iteration j, the image $x^{(j-1)}$ is modified as follows: $x^{(j)} = x^{(j-1)} - \lfloor a_k \cdot \nabla f^y \rceil$, where a_k is chosen in such a way that the k percentage of the pixels with the larger gradient intensity are modified ($\lfloor \rceil$ denotes the rounding operation). Controlling the number of modified pixels also permits to avoid that many neighbouring pixels are modified: in fact, since the effects that single-pixel changes have on the output are presumably not independent for pixels close to each other, the effect of a joint modification of many neighbouring pixels may be difficult to predict. The best k is searched iteratively and corresponds to the minimum fraction for which the boundary can be crossed (see [4] for more details regarding the search of k). Clearly, when the algorithm takes more than one iteration, k is set to the maximum value (k_{max}) for all but the last iteration of the algorithm.

We verified that, when applied to the softmax output of a CNN model, the estimation in (2) sometimes returns a zero matrix, for some trained networks (especially, for deeper models). This happens for large positive and negative values of $z^y(x)$, where the softmax has a rather flat output. To overcome this problem, instead of computing the softmax directly on the values z^y, we get the score function f^y used in (2) by applying the softmax to $z^l(x)/T$, with $l \in \{0, 1\}$, where T is a scaling parameter, known as temperature parameter of the softmax. The effect of considering $T > 1$ is only increasing the sensitivity of the softmax output to a small change in the input, the exact value of the output being not important in this phase, where we only search for an estimate of the descent direction.

Experimental analysis and results: We first briefly introduce the state-of-the-art approaches for generating adversarial examples that we considered in our tests. L-BFGS [1] looks for an approximately optimum solution of the problem of finding the minimal adversarial perturbation that makes the prediction changes. FGSM [5] obtains an adversarial perturbation in a computationally efficient way by computing the gradient of the output with respect to the input image and considering its sign multiplied by a strength ε. JSMA consists of a greedy iterative procedure which relies on forward propagation to compute, at each iteration, a saliency map, indicating the pixels that contribute most to the classification. The pixels are then modified based on this map. The procedure ends when the attacker succeeds or too many pixels are modified. While L-BFGS looks for the minimal perturbation necessary to cross the boundary, both FGSM and JSMA are suboptimal (but more efficient) approaches which try to induce a misclassification while keeping the distortion limited; in this way, they are expected to be more robust against integer rounding.

The Foolbox toolbox [7] is used to implement the above attack methods. For the FGSM attack, we considered the refined multi-step variant, in order to avoid that a highly suboptimum perturbation is generated (hence introducing a large distortion). The number of steps is set to 10 (default) while the best ε is searched in the range $[10^{-3} : 0.001 : 0.1]$. The JSMA is applied with the default maximum number of iterations 2000; the maximum number of times that a same pixel can be

modified is set to 3. Regarding the L-BFGS method, we set a margin ν on the final score, in order to force the attack to enter more inside the targeted region (so to increase the robustness to rounding); specifically, w.l.o.g., if 1 is the predicted class after a successful attack (then, 0 is the true class), and f_a^1 is the corresponding final score of the attacked image, then, the attack is run again and a decision in favor of 1 is made only when $f^1 \geq \min\{f_a^1 + \nu, 1\}$.

With regard to the proposed attack, the most computationally demanding part of the algorithm is the computation of the approximated gradient, hence the time needed for attacking one image depends on the number of iterations required. Typically, the number of iterations ranges from 1 to 3, and the image can be attacked in some seconds. The temperature T of the softmax during the gradient estimation is set to 100. As in [4], we set $k_{max} = 0.2$. We declare an attack failure if the image cannot be attacked in more than 10 iterations (the number of iterations being related to the maximum change per pixel).

The networks we considered are: a shallow one (3 convolutional layers), trained for median filtering detection (ME-D) and resizing detection (RES-D) [8] and a deeper one (9 convolutional layers) trained for contrast adjustment detection (CA-D) [9]. Both ME-D and RES-D were trained on grayscale patches of size 128×128, while CA-D was is trained on color patches of size $64 \times 64 \times 3$, as in [9]. The details of the network and the training information (e.g., architecture, optimization solver, batch size, etc...) can be found in [8] and [9]. The TensorFlow framework via the Keras API is used to train and test the CNNs, in a machine equipped with an Asus GeForce GTX1080TI - 9GB DDR5 gpu. We carried out our experiments on camera-native images from the RAISE dataset [10]. No compression is performed to them. The images were processed with the OpenCV library for Python to produce the manipulated images (class 1). The processing parameters are fixed as follows. The window size for the median filtering is set to 5×5, while the scaling factor for the resizing is set to 3. For the adjustment of the contrast, by following [9], the processing considered are a mixture, in equal percentage, of adaptive histogram equalization (with clip-limit parameter 5), gamma correction (both contrast expansion, with $\gamma = 1.5$, and reduction, with $\gamma = 0.7$) and histogram stretching (5% of saturation). The network accuracies in absence of attacks are 95% for ME-D, 97% for RES-D and 90% for CA-D.

Table 1 shows the performance of the various attacks against the three CNN-based detectors. For each method, the attack is carried out on 300 test images (150 per class) correctly classified by the network. Results are reported for the attacked images produced by each method (*Adv-I*) and for those obtained after rounding and truncation to pixels ($\lfloor$*Adv-I*$\rceil$). Obviously, there is no difference between $\lfloor$*Adv-I*$\rceil$ and *Adv-I* for the proposed method. In the table, *err* indicates the percentage of attacked images misclassified by the network, i.e., those for which the attack is successful; d^1 and d^∞ denote, respectively, the average (normalized) L^1 and L^∞ distance between original and attacked images; finally, *mod* indicates the average percentage of modified pixels in the attacked image. The averages are computed over the successfully attacked samples, i.e., only on the *err* percentage of images.

We see that, in all the cases, the proposed attack succeeds in inducing a decision error with a small distortion, both in terms of d^1 and, especially, of d^∞ (the maximum distortion is in fact the measure that the proposed method tends to minimize). The *err* percentage does not reach 100 due to the presence of a few images which require more than 10 iterations to be attacked. Also, we notice that in all the cases our attack is effective by modifying only a few percentage of the pixels.

The results in the table also confirm that the effectiveness of the existing attacks is significantly reduced after the integer rounding. As expected, L-BFGS (for $\nu = 0$) is the one that suffers more and the perturbation introduced by this method is cancelled out by the rounding more than 80% of the time in all the cases. Setting $\nu > 0$ permits to raise *err* after the rounding, at the price of a larger distortion. For each detector, the table reports the results when ν is set to the value ν^* for which the attack reaches the largest *err* after the rounding (above this value, the adversarial examples can not be found in some cases, then *err* decreases). We observe that, when a sufficiently high success rate can be reached after the rounding, as for the cases of ME-D and RES-D in particular, the distortion introduced is much larger than with our method. The table also shows that, as expected, FGSM gets a larger *err* with respect to L-BFGS on $\lfloor$*Adv-I*$\rceil$; however, the improvement in robustness is slight. By considering a stronger perturbation, which can be done by setting a larger ε parameter, the robustness of FGSM to rounding improves a bit, but at the cost of increased distortion. We see from the table that, for the case of CA-D, where *err* after rounding is around 50%, the average distortion with FGSM is already much larger than with the proposed method, both in terms of d^1 and d^∞. With regard to JSMA, we observe that this method is pretty robust to rounding, especially when it is run against the two shallow models ME-D and RES-D. In these cases, in fact, once the attacked images are rounded to integer, the success rate for the attack is 93.9% and 99% respectively, while it is 84% for CA-D. Regarding the distortion introduced by this attack, JSMA tends to keep d^1 very low at the expense of a large d^∞. Compared to the proposed method, JSMA can achieve a better d^1, however at the cost of a much larger d^∞.

Table 1: Performance of the attack methods against the three manipulation detectors.

		Adv-I				$\lfloor$Adv-I$\rceil$			
		err	d^1	d^∞	mod	err	d^1	d^∞	mod
ME-D	L-BFGS ($\nu = 0$)	100	0.19	1.34	98.77	18.67	0.32	2.93	99.9
	L-BFGS (ν^*)	100	0.33	2.42	99.0	98.7	0.23	2.43	99.0
	FGSM	100	0.42	0.48	98.7	38.7	0.85	1.01	98.9
	JSMA	94.7	0.07	7.37	97.4	93.9	0.08	7.70	97.3
	Pixel-based [prop]	98.7	0.16	1.43	13.0	98.7	0.16	1.43	13.0
RES-D	L-BFGS ($\nu = 0$)	100	0.14	1.10	98.2	5.0	0.29	3.50	99.7
	L-BFGS (ν^*)	100	0.32	2.60	98.8	98.6	0.22	2.60	98.8
	FGSM	100	0.35	0.37	99.2	35.0	0.88	1.01	99.9
	JSMA	99.3	0.06	7.50	97.2	99.3	0.06	7.40	97.1
	Pixel-based [prop]	99.0	0.12	1.20	12.0	99.0	0.12	1.20	12.0
CA-D	L-BFGS ($\nu = 0$)	100	0.10	1.99	95.2	17.3	0.12	3.60	9.5
	L-BFGS (ν^*)	100	0.18	3.45	96.5	71.0	0.14	3.91	10.5
	FGSM	89.4	0.78	1.60	95.3	49.6	1.31	2.83	54.5
	JSMA	92.3	0.06	7.10	92.1	84.0	0.06	7.44	3.0
	Pixel-based [prop]	99.0	0.13	1.80	15.3	99.0	0.13	1.80	15.3

Conclusion: By focusing on deep learning applications for forensics, this letter presents a method for generating adversarial perturbations against CNNs, by minimizing the distortion in the pixel domain. Future work will focus on the extension of the attack to other, more general, quantized domains, e.g. the JPEG domain.

Acknowledgment: This work has been partially supported by a research sponsored by DARPA and Air Force Research Laboratory (AFRL) under agreement number FA8750-16-2-0173.

B.Tondi (*UNISI, University of Siena, Siena, Italy*)

E-mail: benedettatondi@gmail.com

References

1 C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

2 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, 2018.

3 R. Böhme and M. Kirchner, "Counter-forensics: Attacking image forensics," in *Digital Image Forensics*, H. T. Sencar and N. Memon, Eds. Springer Berlin / Heidelberg, 2012.

4 Z. Chen, B. Tondi, X. Li, R. Ni, Y. Zhao, and M. Barni, "A gradient-based pixel-domain attack against SVM detection of global image manipulations," in *IEEE International Workshop on Information Forensics and Security*. Rennes, France: IEEE, 4-7 December 2017.

5 I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

6 N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *IEEE European Symp. Security & Privacy*, 2016, pp. 372–387.

7 J. Rauber, W. Brendel, and M. Bethge, "Foolbox v0. 8.0: A python toolbox to benchmark the robustness of machine learning models," *arXiv preprint arXiv:1707.04131*, 2017.

8 B. Bayar and M. C. Stamm, "A deep learning approach to universal image manipulation detection using a new convolutional layer," in *ACM Workshop on Info. Hiding & Multimedia Security*, 2016, pp. 5–10.

9 M. Barni, A. Costanzo, E. Nowroozi, and B. Tondi, "Cnn-based detection of generic contrast adjustment with jpeg post-processing," *to be presented to ICIP 2018. arXiv preprint arXiv::1805.11318*, 2018.

10 D.-T. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato, "Raise: A raw images dataset for digital image forensics," in *Proceedings of the 6th ACM Multimedia Systems Conference*, ser. MMSys '15, 2015, pp. 219–224.