– Supplemental Material –
SpoC: Spoofing Camera Fingerprints

Davide Cozzolino[1] Justus Thies[2] Andreas Rössler[2] Matthias Nießner[2] Luisa Verdoliva[1]
[1]University Federico II of Naples [2]Technical University of Munich

SpoC shows how to inject camera traces into synthetic images. Given a GAN generated image, we are able to insert the traces of a specific camera model into it, fooling at the same time the state-of-the-art camera model identifiers and GAN detectors. In this supplemental document, we report the details of the architectures used for Generator, Discriminator, and Embedder (Sec. 1). For reproducibility, we report the parameters of the used comparison methods used in the main paper (see Sec. 2). Finally, we analyze the scenario where we attack images by varying the JPEG compression level. In addition, we report the results obtained when we want to fool at the same time both a model classifier and a GAN detector (see Sec. 3).

1. Architectures

Generator Our generator is composed of seven convolutional layers with a fixed stride equal to one (see Fig.1). The number of feature channels increases through the network from 64 to 128 after the first three convolutional layers and is set to the image channel size of three in the last layer. We apply appropriate padding to keep the input image dimensions of 256×256. In order to guide our adversarial training, we apply spectral normalization for our five middle layers as described in [10]. We use ReLU as non-linearity for all layers besides the last. After the input has been passed through our convolutional layers, we use a residual connection to add it to our output and squash the final result back to image space using a Tanh non-linearity.

Discriminator As described in the main paper, the discriminator uses a fixed first layer to extract low level image features. This input is fed into a convolutional layer with a kernel size of three. Afterwards, we use four blocks of convolutional layers using a kernel size of three, spectral as well as mean-only batch normalization. The number of feature channels is 64 for all these layers and we use ReLU as non-linearity. The output is fed into a final convolutional layer with kernel size of three to reduce the number of features to one. We use no padding for all convolutional layers in our discriminator. The discriminator architecture is shown in Fig.2.

Embedder As shown in Fig.3, the main layer in our embedder is a residual block. This block has two branches. In one branch, a convolution with kernel size one is applied, while in the other branch, we make use of two convolutions using a kernel size of three together with a ReLU non-linearity. The outputs of the two branches are then summed up to obtain the final output tensor. We adopt spectral normalization for all convolutional layers. As described in the paper, the input image of the embedder is first passed through our fixed layer to extract low level image features. The output is passed through four residual blocks following

Figure 1: Generator architecture.

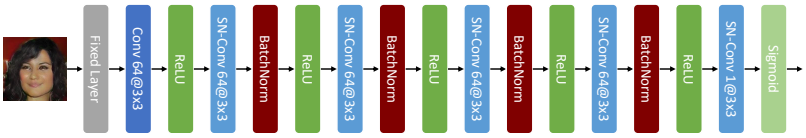

Figure 2: Discriminator architecture.

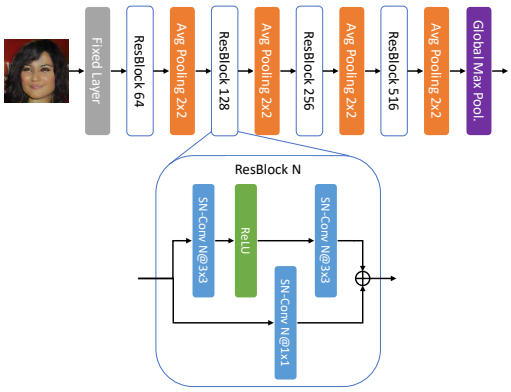

Figure 3: Embedder architecture.

each one by an average pooling of size two. The number of feature channels linearly increases from 64 to 512. The output is pooled to a single 512 dimensional tensor using a global max pooling.

2. Comparison with state-of-the-art

In the main paper we compare our proposal with four techniques that generate adversarial attacks. For all these techniques, we set the parameters in order to obtain a PSNR of about 31dB. In the following, we give more details about these techniques.

PGD (Projected Gradient Descent attack) [9]: PGD is an iterative attack method based on the evaluation of the gradient of the loss function w.r.t the input image. At each iteration, the image is modified with the projection of the gradient into the space of allowed perturbations. For this method, we use a number of iterations equal to 40 and an epsilon for each attack iteration equal to 1.25.

TI-MI-FGSM (Translation-Invariant Momentum Iterative Fast Gradient Sign Method) [4]: It is an iterative version of the Fast Gradient Sign Method with the use of a momentum term for the estimation of the gradient. Moreover, to improve the transferability of the attack, the gradient is computed considering a set of translated versions of the image. For this method, we use a number of iterations equal to 40, an epsilon for each attack iteration equal to 0.4 and, an overall epsilon equal to 8.

GAP (Generative Adversarial Perturbation) [11]: It is a method where a generator network is trained in order to obtain a perturbation able to fool the classifier with a constraint on the maximum allowed perturbation. The authors propose two variants, Universal Perturbation, and Image-dependent Perturbation. In the first case, the perturbation does not directly depend on the image to attack, while in the second case it depends on the image. We compare the proposal with Image-dependent Perturbation that is a less restrictive hypothesis and more coherent with our scenario. As proposed by the authors, for the architecture of the generator network, we use ResNet Generator that is defined in [5]. In our experiments, the epsilon of the constraint is set equal to 8.

Adv-Cam-Id [2]: The white-box attack proposed in [2] uses a generator network that provides a falsified version of the image. The generator network is trained using two losses, one is relative to the capability to fool the classifier and the other is the L1 distance between the original image and the falsified image. The generator architecture is composed of a first block, that emulates the color filter array, and seven convolutional layers. For our experiments, we use the hyperparameters suggested by the authors and stop the training when the PSNR is greater than or equal to 31dB.

3. Additional results

In this section we analyze robustness to compression of our approach. In Fig.4 we show the attack success rate by varying the JPEG compression quality. Camera model classifiers have been trained using images at differ-

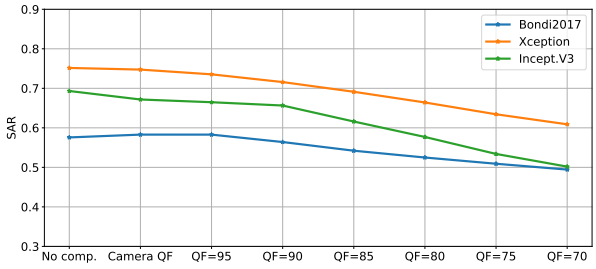

Figure 4: Attack success rate of our proposal by varying the compression level of the attacked images.

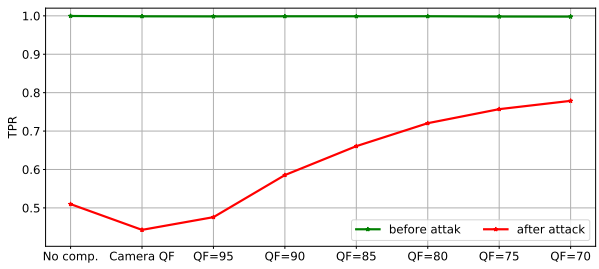

Figure 5: True Positive Rate of the network Xception by varying the compression level before and after the proposed attack.

		in training					out training				
		Xception	Spec	ResNet50	Patch Forensics	FFD	Xception	Spec	ResNet50	Patch Forensics	FFD
Model Clas.	Tuama2016	34.8	51.2	31.0	54.0	40.8	56.3	56.4	53.5	56.8	55.1
	Bondi2017	38.8	59.3	33.9	62.6	47.2	47.9	47.8	45.0	48.4	46.5
	Xception	43.8	66.3	37.9	71.1	54.6	73.6	73.4	69.8	74.2	72.5
	InceptionV3	40.6	63.1	38.8	67.7	51.0	65.3	65.3	61.8	66.0	63.5

Table 1: Successful Attack Rate (SAR) to fool the Model classifier and the GAN detector at the same time considering both images inside (StarGAN [3], CycleGAN [13], ProGAN [6], StyleGAN [7], and RelGAN [12]) and outside the training-set (bigGAN [1], and StyleGAN2 [8]).

ent JPEG compression levels, so as to improve their performance also on compressed data. In this analysis we exclude Tuama2016 because it achieves an accuracy below 50% in this scenario. Results are shown in Fig.4 and show that the attack is still effective even on compressed images, in particular the attack success rate still remains above 50% also for images compressed at JPEG quality in the range $[70 - 75]$, which is the quality level typically applied by a social networks when an image is uploaded.

We also carry out a similar analysis on GAN detectors. The trend by varying the JPEG compression is very much similar for all the GAN detectors, hence in Fig.5 we only report the behavior of Xception by varying the compression level. Performance of the detector are perfect before the attack and then reduce strongly after our attack, even on heavily compressed images.

Finally, we consider a scenario where the objective is to fool both the camera model classifier and the GAN detector at the same time. Results are shown in Tab.1 and confirm that our approach is able to obtain satisfying results, especially when attacking deeper networks for camera model identification and GAN detectors that were not trained on that specific GAN images.

References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018. 3

[2] Chen Chen, Xinwei Zhao, and Matthew C. Stamm. Generative adversarial attacks against deep-learning-based camera model identification. *IEEE Trans. Inf. Forensics Security, in press*, October 2019. 2

[3] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[4] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[5] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711, 2016. 2

[6] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *International Conference on Learning Representations*, 2018. 3

[7] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4401–4410, 2019. 3

[8] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, pages 8110–8119, 2020. 3

[9] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 2

[10] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 1

[11] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative Adversarial Perturbations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4422–4431, 2018. 2

[12] Po-Wei Wu, Yu-Jing Lin, Che-Han Chang, Edward Y. Chang, and Shih-Wei Liao. Relgan: Multi-domain image-to-image translation via relative attributes. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 3

[13] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 3